Utilising the concept of Human-as-a-Security-Sensor for detecting semantic social engineering attacks

Ryan John Heartfield

A thesis submitted in partial fulfilment of the requirements of the University of Greenwich for the Degree of Doctor of Philosophy

DOCTOR OF PHILOSOPHY

June 2017

Declaration

"I certify that the work contained in this thesis, or any part of it, has not been accepted in substance for any previous degree awarded to me, and is not concurrently being submitted for any degree other than that of Doctor of Philosophy being studied at the University of Greenwich. I also declare that this work is the result of my own investigations, except where otherwise identified by references and that the contents are not the outcome of any form of research misconduct."

SUPERVISOR: ....
. RYAN HEARTFIELD

SUPERVISOR: ....
. DR. GEORGE LOUKAS

SUPERVISOR: ....
. DR. DIANE GAN

Abstract

Social engineering is used as an umbrella term for a broad spectrum of computer exploitations that employ a variety of attack vectors and strategies to psychologically manipulate a user. Semantic attacks are the specific type of social engineering attacks that bypass technical defences by actively manipulating object characteristics, such as platform or system applications, to deceive rather than directly attack the user. Semantic social engineering attacks are a pervasive threat to computer and communication systems. By employing deception rather than by exploiting technical vulnerabilities, spear-phishing, obfuscated URLs, drive-by downloads, spoofed websites, scareware and other attacks are able to circumvent traditional technical security controls and target the user directly. In this thesis, we begin by defining the terminology of a semantic attack, introducing a historic time-line of attack incidents over the last 17 years to illustrate what is an existential relationship with the user-computer interface and it's ever expanding landscape. We then highlight the scale of the semantic attack threat by identifying different individual attacks and discussing recent statistics. Recognising the complexity in understanding the many facets that may form an attack, as well as the depth and breadth of the threat landscape, we construct a taxonomy of semantic attacks which encapsulates attack characteristics into a fixed, parametrised classification criteria that span all stages of a semantic attack. We then supplement the taxonomy of attacks with a survey of applicable defences and contrast the threat landscape and the associated mitigation techniques in a single comparative matrix; identifying the areas where further research can be particularly beneficial. Armed with this knowledge, we then explore the feasibility of predicting user susceptibility to deception-based attacks through attributes that can be measured, ethically, preferably in real-time and in an automated manner. We conduct two experiments, the first on 4333 users recruited on the Internet, allowing us to identify useful high-level features through association rule mining, and the second on a smaller group of 315 users, allowing us to study these features in more detail. In both experiments, participants were presented with attack and non-attack exhibits and were tested in terms of their ability to distinguish between the two. Using the data collected, we determine predictors of users' susceptibility to different deception vectors. With these, we have produced and evaluated a generalised model for training a dynamic system for proactive user security. Using the model as a baseline, we propose a technical framework that aims to utilise the concept of Human-as-a-Security-Sensor as a dynamic defence mechanism against semantic attacks. To test the viability of our framework and to demonstrate the concept of the Human-as-a-Security-Sensor in an empirical context, we employ the framework to develop a prototype Human-as-a-Security-Sensor platform called *Cogni-Sense*; evaluating its utility in a real-world experiment. Lastly, we conclude with a review of the problem space, summarising our novel contributions towards a dynamic, user-driven defence against semantic attacks and identify open problems in our work to discuss future plans and motivation for continuing the development of Human-as-a-Security-Sensor.

Acknowledgements

In all the theses that I have read, all agree that embarking on a PhD is a journey. None were wrong, and like all journeys, there are good times and not so good times. Along the way one experiences many emotions, inspiration, excitement, elation, as well as self-doubt, anxiety and despair where it feels as one is trying to fix an unfixable problem. For me, the journey has been all these things, but the single most important thing I have learned throughout this journey is that without the incredible and unwavering support of my supervisor, colleagues, friends and family, this PhD simply would not have been possible.

I am eternally grateful to my supervisor Dr. George Loukas. George is a remarkable human, possessing incredible insight which has helped me to illuminate ways forward when there seemed to be only dead-ends. He is a supervisor any PhD student would wish for - enthusiastic, motivating, providing support in times of need, as well as possessing a great sense of humour! If it was not for George seeing potential in me I would never have undertaken a PhD, and over the past four years his mentorship has guided me to develop a success as a computer science researcher that I never before thought possible. George, thank you for helping me unlock my potential.

I give heartfelt thanks Dr. Diane Gan for her continued support and guidance as my second supervisor and as my lecturer and mentor through my academic journey from undergraduate to PhD. Your continued enthusiasm and motivation has been both infectious and inspiring.

I would also like to thank my work colleagues Andy, Sean, Stuart and Zed for their continued support and input and enthusiasm when sharing ideas related to my research, it has helped me to think outside-the-box and improved the quality of this project. A big thank-you also goes to Riley, who helped me turn my unfriendly install process for *Cogni-Sense* into an automated, user-friendly script!

To my beautiful Carla, you have always believed in me, even when I have lost belief in myself. You will never know how much your love and support has driven me to make this PhD possible.

Dedication

For my Carla.

List of Author Publications

Manuscripts published

- 1. R. Heartfield and G. Loukas. *On the feasibility of automated semantic attacks in the cloud.* Computer and Information Sciences III. Springer London, 2013, pages 343-351.
- 2. R. Heartfield and G. Loukas. *A Taxonomy of Attacks and a Survey of Defence Mechanisms for Semantic Social Engineering Attacks.* ACM Computing Surveys (CSUR), 48(3), 2016.
- 3. R. Heartfield and G. Loukas. *Evaluating the reliability of users as human sensors of social media security threats.* Cyber Situational Awareness, Data Analytics And Assessment (CyberSA), 2016 International Conference On. 14 June 2016. IEEE.
- 4. R. Heartfield and G. Loukas. *Predicting the performance of users as human sensors of security threats in social media.* International Journal On Cyber Situational Awareness (IJCSA). 1(1). 2016
- 5. R. Heartfield and G. Loukas, and D. Gan, D. *You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks.* IEEE Access, 4, 2016, 6910-6928.
- 6. R. Heartfield and D. Gan. *Social Engineering in the Internet of Everything.* Cutter Business Technology Journal. 26(7). 2016, Pages 20-29

- 7. R. Heartfield and G. Loukas and D. Gan. *An eye for deception: A case study in utilising the Human-As-A-Security-Sensor paradigm to detect zero-day semantic social engineering attacks.* Software Engineering Research, Management and Applications (SERA 2017), 15th ACIS International Conference on. 7 June 2017. IEEE.
- 8. S. S. Rahman, R. Heartfield, W. Oliff, G. Loukas, A. Filippoupolitis. *Assessing the cyber-trustworthiness of human-as-a-sensor reports from mobile devices.* Software Engineering Research, Management and Applications (SERA 2017), 15th ACIS International Conference on. 7 June 2017. IEEE.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In information security, the user is often seen as the "weakest link" [3] because even the strongest technical protection systems can be bypassed if the attacker successfully manipulates the user into divulging a password, opening a malicious email attachment or visiting a compromised website. The term most often used for this process is *Social Engineering*, but this does not differentiate attacks that bypass the security of computer systems from those that can be observed in a non-technical arena, such as prize-winning letter scams [4] or physically impersonating an authority figure [5]. To differentiate from these, researchers have proposed the term *Semantic Attack* [6, 7, 8]. As semantics is the study of meaning and symbolisation, in the context of social engineering a semantic attack is one that manipulates the user-computer interface to deceive the user and ultimately breach the computer system's security. We have proposed the following definition [9]:

Semantic Attack. The manipulation of user-computer interfacing with the purpose to breach a computer system's information security through user deception.

On any system, the user interface is always vulnerable to abuse by authorised users, with or without their knowledge. Traditional deception-based attacks, such as phishing emails, spoofed websites and drive-by downloads, have shifted to new and emerging platforms in social media [10], cloud applications [11] and near field communications [12]. Furthermore, the advent of the Internet of Things [13] promises to compound the problem and extend to physical impact, exposing user interfaces of systems previously inaccessible to the standard user, let alone via a distributed application in the Internet [14]. The more effective such cyber-physical attacks prove [15], the more the deception attack surface continues to grow.

Semantic attacks have been characterised as a technical enigma for information technology [16]. That is because the user-computer interface is at the same time their only boundary and their primary window of opportunity. The expanse of the problem space and variation of attacks are extreme (from state-backed Advanced Persistent Threats employing multi-stage/platform attack vectors to script kiddies generating automated phishing emails), obscure (defining what constitutes a semantic attack through its component structure) and arbitrarily difficult to detect (as attack vectors primarily address the user rather than the technical system). They can be technically basic [7, 17, 18], highly complex [19, 20] or a combination of the two [11, 21]. Defence mechanisms have been proposed at scientific research level to target exploitations such as website and phishing attacks [22, 23, 24, 25, 26] as well as at commercial level [27, 28]. However solutions such as these have not addressed the wider semantic attack problem space, and mitigations tend to be specific to given exploits; static and disparate. Without the dynamic capabilities observed in other defence systems such as antivirus and firewalls, they are often overcome by attackers who subtly adjust their approach and render exploit-driven detection mechanisms no longer relevant.

1.1 Outline

We start in Chapter 1 by presenting a historic time-line of attack incidents to illustrate the evolution of semantic attacks over the last 17 years and discuss recent attacks statistics to highlight the scale of threat as it is today. In Chapter 2, we present a taxonomy of semantic attacks which introduces a generic classification criteria for simplifying the problem space and supplement the taxonomy with survey of applicable defences mechanisms. In Chapter 3, we explore the feasibility of predicting susceptibility to deception-based attacks by conducting two user studies, the first a large scale survey on 4333 users which informed a refined, smaller survey on

315 users, using the data collected to determine predictors of susceptibility and produce generalised model for training a dynamic system for proactive user security. In Chapter 4, we propose a technical framework based on the concept of "Human-as-a-Security-Sensor" (HaaSS), drawing together the generalisation of semantic attack classification, use susceptibility and the aim to develop a technical defence into a single architecture for detecting and mitigating semantic attacks. To demonstrate the real-world practicality of the framework, we develop a prototype HaaSS platform called *Cogni-Sense* which we evaluate in an empirical experiment with real users. In Chapter 5, we conclude with a summary of our contributions, discuss open problems in our work as well as the wider problem space, and provide key motivations and future research and development direction of the HaaSS concept as an essential defence mechanism against semantic attacks.

1.2 Characterising the threat of semantic attacks

Semantic social engineering attacks are a pervasive and persistent threat to computer and communication systems; existentially linked to the user-computer interface. The problem space is both paradoxically and implicitly related to this fact: as long as a user-computer interface exists, a computer or communication system is inherently at risk to malicious manipulation to deceive a user into eliciting an action that will compromise their own or their system's information security.

1.2.1 A brief historic overview of social engineering in computer systems

In computer systems, semantic social engineering attacks can be traced back to as early as 1989 when the *"AIDS Information Introductory Diskette"* Trojan [29] was sent to a mailing group in which Dr Joseph Popp, the Trojan's author, subscribed. To enter a computer system the attack exploited the use of a diskette pertaining to contain information about the AIDS virus which deceived the recipients of the diskette into inserting the media in their system. In fact, the diskette contained a malicious program known as a Cryptovirus [30] which ransomed users for money by encrypting their systems files. Another noteworthy semantic attack occurred only a year later in 1990, introducing a concept that is known today as “Scareware”. The malware, aptly named *Nightmare* [31], was distributed via diskettes called “Fish Disks” designed to share applications between Amiga computer systems of the time. On execution, every 5 minutes the screen was hijacked for four fifths of a second to display a full-screen image of skull with bullet wound and blood leaking out, at the same time playing a loud shriek on the Amiga’s audio channel. Whilst the malware posed no risk to user data, the concept of scaring/panicking a user would later be employed by many cyber-criminals to force users into opening malware and / or for paying for fraudulent services [32]. In 1995, five years later, new semantic attacks began to appear which were specifically designed to exploit users accessing resources over a new open network, called the Internet. Domain investor John Zuccarini introduced the concept of Typosquatting or Cybersquatting into the mainstream, where cyber criminals would purchase domain names that were similar to those of legitimate website. Users who mistype the domain name URL of a legitimate website would then be redirected to a malicious or fraudulent website. During the same year, service provider America Online (AOL) had experienced growing success with a popular instant messaging tool. Hackers soon realised that the application could be exploited and developed an attack tool that lead to the first use of the term “phishing”. *AOHell* [33], contained a “fisher” tool that enabled hackers to steal passwords and financial information through automated social engineering by generating instant messages to random AOL users with content such as: “Hi, this is AOL Customer Service. We’re running a security check and need to verify your account. Please enter your username and password to continue”.

Over the next decade, phishing attacks, in particular, had become widespread; with the period of January to December 2005 alone accounting for 173,063 reported phishing attacks, of which 49,774 were new phishing platforms [34]. However, it was five years earlier in the year 2000, when one of the highest profile email phishing attacks to have ever occurred was discovered. The infamous *ILOVEYOU* "worm"[1], contained a malicious visual basic script titled "LOVE-LETTER-FOR-YOU.txt.vbs" [37], initially spreading through corporate Philippine mailing lists and eventually affecting over 45 million computers systems world-wide; resulting in what was one of the world's most dangerous computer related disasters at the time. This attack was copycatted a year later in 2001 by the *Anna Kournikova* worm, using the same worm generating script [38]. The same year, the first known phishing attack against a financial institution was discovered, where E-Gold users were targeted with emails tricking them into entering their passwords into phishing websites [39]. Leading up to today, the exponential growth of the Internet, multimedia services and mobile platforms, have enabled semantic attacks to spread further into Android devices [40], peripheral hardware accelerated by direct memory access [41] (e.g., Thunderbolt and Firewire devices), file sharing networks [42], search engine optimisation engines [43], targeted, drive by malware on websites [44] and the landscape continues to expand. For example, the advent of online social networks and increase in online social media has introduced a paradigm shift in Internet communication; where platform functionality promotes openness and information sharing amongst users. This new, online social paradigm has enabled cyber criminals to take advantage of new concepts such as "friend" recommendations, user "posts" and sharing of media or apps that are replicated and automated with the network [10, 45, 46].

Even more concerning is the potential for semantic attacks to result in physical impact, outside of their traditional "cyber" domain. For example, in December 2014 damage was caused to a German steel mill furnace when hackers used targeted phishing emails as an initial entry vector to gain remote access to the steel mill office production network. Through spear phishing, the attackers captured key user credentials which provided remote network access enabling further technical penetration

[1]Note that here we use the term "worm" to refer to a malware with a semantic attack vector that exhibits automated, self-replicating behaviour, as stated by [35]: "To spread, worms either exploit a vulnerability on the target system or use some kind of social engineering to trick users into executing them". However, a more traditional definition of a computer worm, (such as a network worm) defines a worm as: "A malicious software that propagates automatically without human interaction, using a vulnerability that has not been patched or widely acknowledged at the point of an outbreak." [36].

of the product management software of the steel mill and ultimately control over the plant control systems with devastating consequences [47]. Another example of physical impact occurred when households in Ukraine suffered a blackout on 23rd December 2015 which was caused by an attack which brought down a portion of the national power grid. Again, the attackers initially used phishing emails to trick users at the electricity company into clicking on an attachment in an email, purportedly from the Prime Minister of Ukraine [48]. Here, the e-mail attachment contained a malicious macro which ran the notorious *BlackEnergy* malware, which is a Trojan horse software designed to launch distributed denial-of-service (DDoS) attacks and in this case deliver an exploitation called *KillDisk*; an exploitation that had the capability to disable or destroy critical software components on embedded systems (such as Industrial Control Systems). As with the German steel mill, here, the attack relied on a semantic attack vector as an initial entry point into the target system as a means to deploy further sophisticated technical exploitation against vulnerable embedded control systems. This is thought to be the first cyber-attack which brought down a national power grid leaving 80,000 homes without electricity. As integration between physical things and computer systems continues to evolve (e.g., Internet of Things) and the more effective such cyber-physical attacks prove [15], the more the semantic attack surface continues to grow.

In Figure 1.1, we provide a time-line containing an extract of high profile semantic attacks from the first publicised incident in 1989 to the most recent attacks of today; identifying the chronological emergence of different semantic attacks as well as the persistence of those into current times. The time-line shows that from 1989 to 2016, using a 5-6 year frequency, there has been a steady increase in the diversity (e.g., platform, device), size and impact of semantic attack exploitations across computer systems. To a large degree, the rise of semantic attacks is symptomatic of the explosion of personal computing and the exponential growth of the Internet over the last thirty years. However, importantly, we can see that irrespective of advancements in computing security over this time period there has been a persistent appearance of the same attacks such as email phishing and Trojan horse malware (which were identified as early as 1995), which continue to be highly successful today. Moreover, unlike in 1995 where the impact of phishing and Trojan malware was limited to traditional computer systems, phishing today has enabled attackers to execute attacks that result in physical impact.

Back in 1996, IT security expert Ira Winkler conducted a publicised study on the non-technical threats to computer systems [49], providing the first serious insight into the impact of semantic attacks. Two years later, IT security researcher David Harley produced a research paper that documented the many facets of technical and non-technical social engineering attacks and the risk posed to computer systems of the time [50]. However, in the decades that has followed the threats of semantic attacks today remain highly reminiscent of those originally identified by Winkler and Harley's research. In fact, the problem has remained the same, whilst the threat space has grown exponentially as computer technology has become a richer and more essential part of everyday life. Furthermore, a recent survey by Intel Security showed that even modern IT security experts, trained specifically to detect semantic attack such as phishing, were still found to have difficulty distinguishing between these types attacks and legitimate system activity [51].

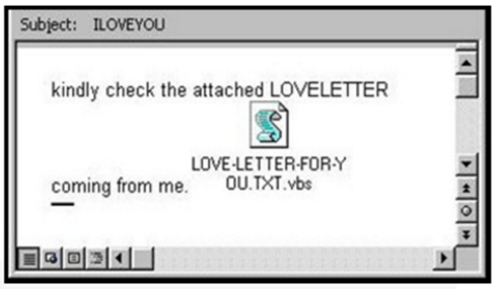
1.2.2 The extreme diversity of semantic attacks

As of today, there exists over 35 individually recognised types or variations (characterised by pseudonym) of semantic attack, existing within and cross-contaminating between a vast range of different platforms and systems. We summarise these types in Table 1.1 to highlight the number of different semantic attacks in the wild today and the extreme diversity in an attacker's arsenal when launching semantic attacks against computer systems.

HISTORY OF NOTABLE SEMANTIC ATTACKS

1989 - 1995

Trojan: “AIDS” Trojan
Scareware: Nightmare
Phishing (Instant Messaging): “AOHell”
Typosquating: Zuccarini’s Domains



Subject: ILOVEYOU

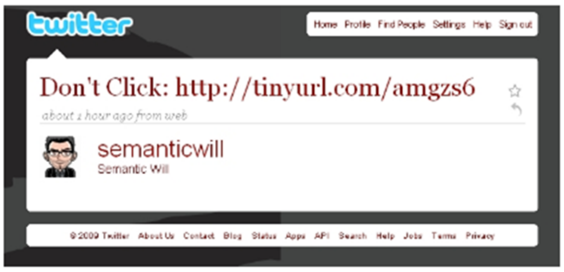
kindly check the attached LOVELETTER coming from me.

1995-2005

Phishing: E-Gold (Post 9-11 ID Check)
Phishing: ILOVEYOU worm
Phishing: Anna Kournikova worm
SMS Trojan: Mosquito (Symbian OS)
Vishing: VoIP credit card scam

2005-2011

Clickjacking: Twitter worm
Clickjacking: Facebook worm
Tabnabbing: Aza Raskin
Android Trojan: Tapsnake
SEO Poisoning: Ipad Announcement
DMA Attack: Inception (PCI, Firewire)
Phishing: Japanese earthquake
Malvertisement: New York Times banner feed
Torrent poisoning: Bitorrent



twitter
Home Profile Find People Settings Help Sign out

Don't Click: http://tinyurl.com/amgzs6

about 1 hour ago from web

semanticwill
Semantic Will

© 2009 Twitter About Us Contact Blog Status Apps API Search Help Jobs Terms Privacy

2011-2016

Phishing: RSA SecurID breach
Waterhole: The VOHO affair
Waterhole: Hidden Lynx Bit9
Phishing Tweet: Associated Press Twitter hijack
Phishing & Driveby Download: Game over Zeus
Phishing & File masquerading: Dridex
Phishing: LinkedIn (Professor Quisquater)
Scareware: Lizamoon (SQL injection attack)
Phishing & Trojan: Target third-party takedown
Android Spear phishing: Tibetan Activists
Phishing: Japanese earthquake
Phishing: IoT phishing attack (Proofpoint Report)
Facebook fake apps: Osama Bin laden video scam
SMS Android Worm: Selfmite
Phishing: Ubiquiti Networks 39.1 million IDs stolen
Ransom32: Ransomware-as-a-Service (RaaS)
Spear Phishing: Ukrainian Power Grid

Figure 1.1: Historic time-line summarising notable Semantic Social Engineering Attacks

Table 1.1: Different types of semantic attack observed in today's computer systems

Attack Pseudonym	**Description**
Spam	Irrelevant/unsolicited messages sent over the Internet to a large number of users, often containing advertising scams
Phishing	Attempt to obtain access to sensitive information by disguising as a trustworthy entity in an electronic communication
Spear phishing	Phishing attack designed to target a specific person/organisation/system
Pharming	Malicious code is installed on a personal computer or server, misdirecting users to fake web sites without knowledge or consent
Whaling	Phishing attack that targets high-profile end users such as corporate executives, politicians and celebrities.
Vishing	Making phone calls/leaving voice messages purporting to be from reputable companies in order to access sensitive user information
QRishing	Phishing attack using quick response (QR) codes
Blue Snarfing	Phishing attack enticing a user to install a malicious file allowing access to the users device via the bluetooth protocol
Smishing	Phishing attack on SMS
URL spoofing	A spoofed URL is where a website poses as another by e.g., copying domain name, sometimes exploiting bugs in web browsers
DriveyBy download	Implanting a malicious file on a vulnerable web platform
Waterhole	Targeted DriveBy download attack
File Masquerading	Disguising a malicious file to appear as a legitimate file type
Multimedia Masquerading	Disguising a malicious application appear as multimedia (e.g., video)
GUI Confusion	A mobile application confusing users by impersonating as another application (e.g., banking app) to obtain sensitive information
Adware	Software that automatically displays or downloads advertising material such as banners or pop-ups when a user is online.
SSL Spoofing	MitM attack that intercepts HTTPS web requests, redirecting the users to malicious and fake HTTPS website
Visual SSL spoofing	Process of using fake SSL verification logos or browser GUI components to visually deceive users into thinking they are on a secure website
Scareware	Malicious program tricking a user into buying/downloading unnecessary often malicious software, such as fake antivirus protection.
Rogueware	A standalone malware program that pretends to be a well-known program or a non-malicious one in order to steal sensitive data
Malvertisement	An online advertisement that incorporates or installs malware.
WiFi Evil Twin	A fraudulent WiFi access point that often spoofs other nearby access points that appears to be legitimate.
Rogue AP	Wi-Fi access point installed on a network but is not authorized for operation on that network and appears to be legitimate
Trojan Horse	Type of malware that is often disguised as legitimate software, such as a game that is actually a key-logger.

Self XSS	Operates by tricking users into copying and pasting malicious content into their browsers' web developer console.
Typosquating	A form of cybersquatting relying on mistakes such as typographical errors made by users when inputting an address into a browser.
RansomWare	A type of malicious software designed to block access to a computer system until a sum of money is paid, often using fear tactics.
Tabnabbing	A type of phishing where a website changes to impersonate popular websites
Sharebaiting	Enticing web content on social media that persuades users to share on their profile, often used to spread fake apps and phishing URLs
Click Jacking	Concealing hyperlinks beneath legitimate click-able content, thereby causing the user to perform actions of which they are unaware.
Like Jacking	Variation on clickjacking in which malicious coding is associated with a Facebook Like button.
Touch Jacking	Variation of clickjacking that applies to mobile devices where users touch the interfaces instead of using a mouse or keypad to click
Cursor Jacking	Variation of clickjacking where users are deceived by means of a custom cursor image, where the pointer is displayed with an offset.
Spamdexing	Manipulation of search engine indexes where a website repeats unrelated phrases to manipulate relevance or prominence
Torrent Poisoning	Intentionally sharing corrupt data and malware with misleading file names using the BitTorrent protocol
DNS Cache Poisoning	Process by which DNS server records are illegitimately modified to replace a website address with a different address.
Fake App	Variation of trojan horse, rogueware, scareware on mobile devices where a malicious app masquerades as a legitimate one
Fake Plugin	Malicious plugin to view a video typically spread by re-posting the fake video message to a victims profile page without permission
Madware	Aggressive advertising placement in mobile devices photo albums, calendar entries and notification bar

1.3 The scale of the threat today: recent semantic attack statistics

Semantic attacks statistics have been dominated by phishing incidents in recent years, due to their widespread use by cyber criminals and consistent success in breaching computer systems. A report by Trend Micro in 2012 identified that over 90% of targeted malware attacks discovered were initiated through spear-phishing attacks [52]. In 2014, *Social Engineer* reported that out of the 129 billion emails sent daily, 90% of these were spam and contained viruses. Moreover, clicking on email links accounted for 80% of reported phishing attacks and phishing itself represented 77% of all socially-based attacks [53].

In 2015 leading online statistics company Statista reported that phishing and deception-based attacks accounted for 62% of all cyber attacks experienced by companies world-wide [54], where 59% were reported by US companies alone [55]. Furthermore, the average number of days to resolve this type of attack for a US based company took 20 days [56], with damages of 12% and 16% for medium and enterprise companies total operating costs, respectively.

The Anti-Phishing Working Group (APWG) produce yearly statistics related to the current trends across a multitude of different phishing attacks that are reported from around the world to their online phishing repository. We have compiled data from the APWG phishing activity trends report archive [1] for years 2008 to 2016, showing in Figure 1.2 the cumulative number of phishing reports received by APWG over this period. With exception of 2008 to 2009, the linear increase in the number of phishing reports received by the AWPG from 2010 to 2013, then becomes exponential up to the current 2016 report (which at the time of writing did not yet include beyond September 2016 for the reporting year). These statistics only include reports made to the APWG and therefore are likely to only provide a snapshot of the actual number of phishing attacks that have been received during this time period.

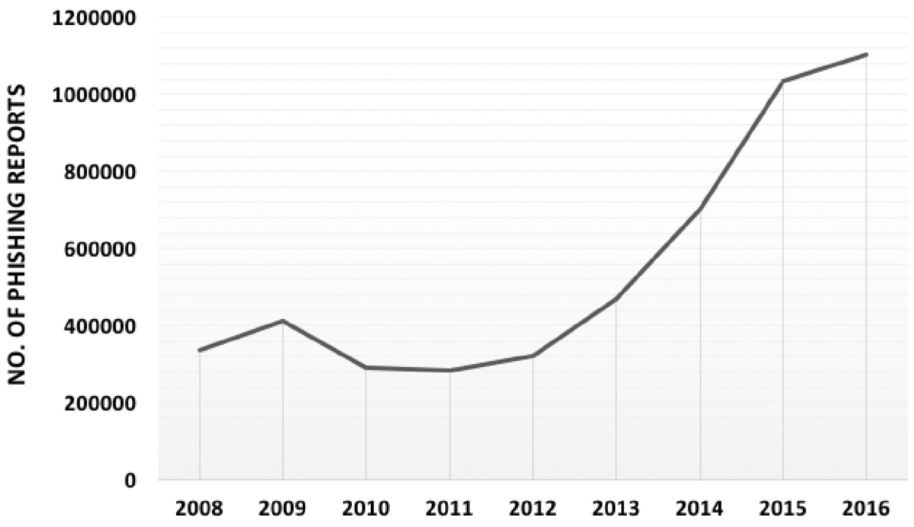


Figure 1.2: APWG Phishing report statistics for years 2008-2016 [1]

The Internet Security Threat Report, a yearly security study produced by Symantec, expands beyond traditional phishing statistics and organises semantic attacks (as well as other conventional malware and computer exploitations) amongst four categories: mobile and Internet of things (IoT), social media and spam, web threats and targeted attacks. Figures 1.3, 1.4, 1.5 and 1.6 summarise a number of semantic attacks and threats utilising deception techniques from years 2013 to 2015 [2]. For mobile platforms, over the course of three years approximately 5 million apps were categorised as malware; with some further 4 million grayware applications also categorised as madware (adware on mobiles). Amazingly, only 25% of mobile apps analysed during this time period were categorised as legitimate. In all cases, malware, grayware and madware require users to agree to install applications, granting permissions to the applications, irrespective of whether any further deception techniques are used (e.g., during app usage); which indicates low user awareness of mobile app vulnerabilities where users are likely to be deceived by a lack of perceived threat. Social media attacks were consistently shown to be propagated largely by users manually sharing posts and apps amongst friends and groups, instead of automated "free offerings" (e.g., surveys and malvertisements) that were dominant in 2013; further highlighting the vulnerability of users behaviour in online social network platforms. Spear phishing campaigns were also observed to have consistently increased over the last three years, whilst the number of recipients per campaign have decreased by an average of 25% each year which may indicate that attackers are developing more sophisticated methods for spear phishing which require less targets for successful exploitation; which also helps to lower detection by creating a smaller attack footprint. Whilst spear phishing attacks continue to target financial industries as cyber criminals aim to steal money for financial gain, recent attacks in 2015 have seen attackers switching some focus to energy and health-care sectors (Figure 1.6). These attacks demonstrate how threat actors are now beginning to target what were traditionally isolated environments, as semantic attack vectors are re-focused toward the Internet of Things.

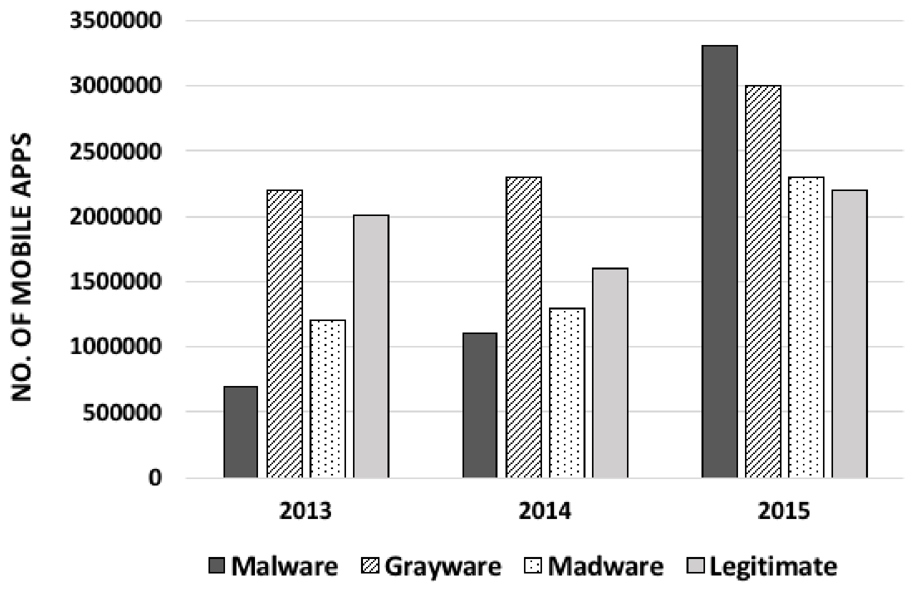


Figure 1.3: Classification of mobile apps analysed by Symantec during 2013-2016 (total apps analysed - 2013: 6.1m, 2014: 6.3m, 2015: 10.8m) [2]

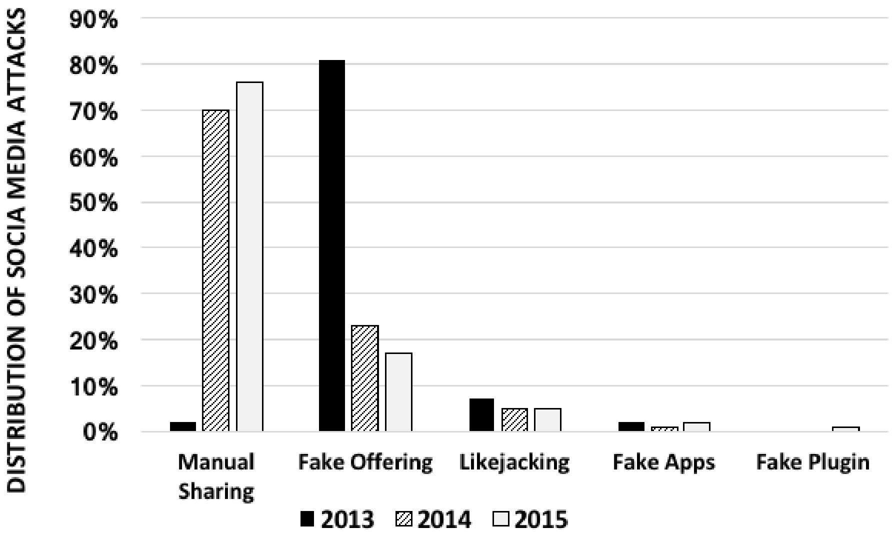


Figure 1.4: Distribution method of social networks and social media scams/attacks by percent

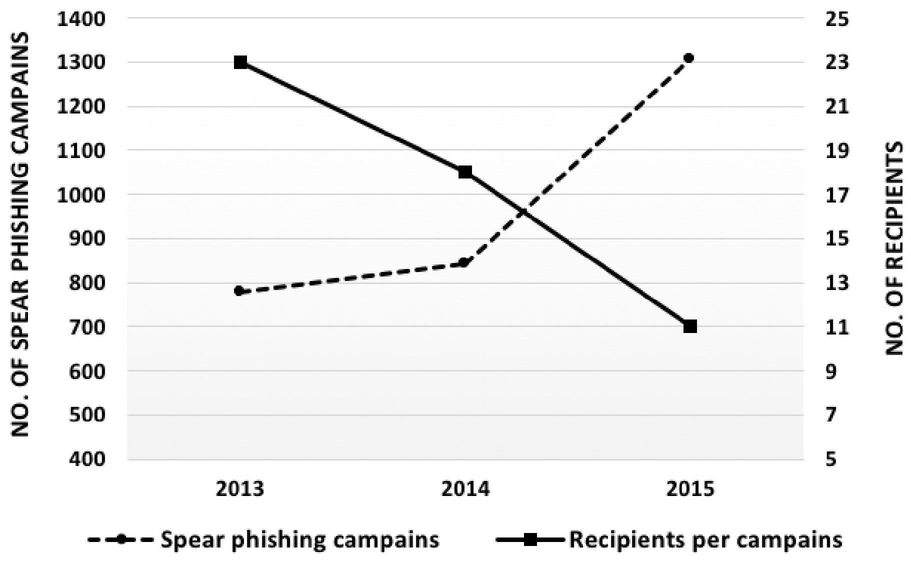


Figure 1.5: Number of spear phishing campaigns and average number of attack recipients per campaign [2]

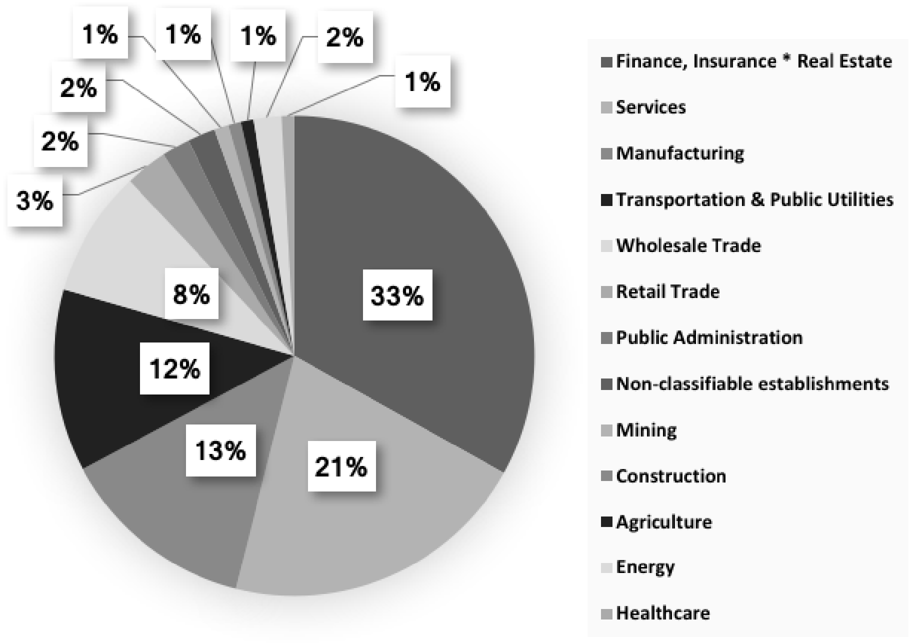


Figure 1.6: Top industries targeted by spear phishing attacks in 2015, ordered by majority percentage [2]

1.4 Research Aim

Semantic attacks pose a significant and sustained threat to computer information security. The attack landscape is vast, with attacks covering a multitude of disparate platforms. As a result, the problem space is difficult to define as it continues to grow. Consequently, there is an increasing need to identify defences that can operate across the wider attack space, rather than in a atomistic fashion addressing specific attack vectors, on specific platforms only. Such an approach to defence would help to simplify an arbitrary degree of complexity in respect to the overhead of managing multiple defences platforms, scaling such systems, their practicality for different types of users (e.g., individual home users, usability for non-experts etc.) and future applicability to new threats on emerging platforms.

This thesis's aim is to produce a defence system able to detect a wide range of semantic attacks, irrespective of target platform or attack vector.

Towards this aim, our principle objectives are as follows:

- To investigate whether semantic attacks in computer systems share key characteristics that can be utilised to develop approaches to defence that are not attack-specific.
- To investigate whether different users ability to detect semantic attacks in a human-as-a-security-sensor role can be predicted.
- To develop a technical defence system for automatic estimation of a users ability to detect an attack as a means to facilitate dynamic protection against semantic attacks.

In the next chapter, we start by exploring in-depth the existing attack landscape, evaluating the current literature related to attacks and identify existing taxonomies that have been developed to classify semantic attacks. We then propose a novel, systematic, parameterised taxonomy designed to identify and categorise key characteristics observed in semantic attacks that are independent of individual attack vectors. We then conduct a survey of defences and map their application to our taxonomy classification criteria in order to highlight the key areas of defence that provide a direction of travel towards the development of a novel, dynamic approach to defence.

Chapter 2

A taxonomy of attacks and survey of defence mechanisms

Semantic attacks are typically grouped by type of exploitation into specifically related attack families. For instance, phishing has become synonymous with fake websites and emails [17], whereby victims are targeted through a baiting mechanism [57]. In this example, the attack family is phishing, whereas phishing emails and phishing websites assume the exploitation type. Another example would be fake applications posing as legitimate pieces of software (scareware, rogueware [58] etc.), which can be associated with the Application Masquerading attack family. A majority view of currently observed semantic attacks is presented in Table 2.1, which combines terminologies commonly used by information security practitioners to identify semantic attack exploitations into related attack families. This static approach fails to identify common attributes of exploitations that are grouped under different attack families. For example, a phishing URL on a web page or email may share a similar method of obfuscation/deception with the one used in a bogus video URL on a social network site (SNS). So, a defence mechanism that would defeat the particular method would potentially be useful for both types of semantic attacks.

Attack-specific approaches to classification have historically led to development of computer security systems that rely on manually generated attack signatures and definitions [59], which can be circumvented in a number of ways (e.g. by zero-day vulnerabilities, obfuscation, polymorphism [60] etc.). Through taxonomic research and

Attack Family	**Exploit variant**
Phishing	Email, Website, URL, IM, Forums, SMS, IRC
File Masquarading	Document , System, Application files
Application Masquerading	Scareware, Rogueware, Ransomware, Trojan horse
Web pop-up	Media-plugin, Alert / Error message, bogus survey
Malvertisement	Infected ad, One-click fraud, Download button
Social Media / Networking	Friend injection, Fake-video link, Fake Game, Likejacking
Removable Media	USB, Flash / SD, Firewire, CD/DVD, PCI
Wireless	Rogue AP, RFID, bluetooth

Table 2.1: Examples of semantic attack exploit family categorisation

development, defence systems have employed techniques that analyse relationships between application behaviour and response (sandboxing [61, 62], dynamic anomaly based scanning [63] etc.). These systems have enabled dynamic and proactive response to security threats on multiple technical platforms, from mobile to desktop operating systems. Yet, this approach has not been replicated or realised for semantic attacks.

A taxonomy of the semantic attack problem space can help researchers evaluate the applicability and scope of proposed solutions for different current and future threats. Some first steps have already been taken in this direction, but still to a limited extent. For instance, [64] have classified attacks based solely on the primary attack vector, such as whether it is a website or email, while [65] have used a hierarchical classification structure that links specific attack vectors to the conditions of their delivery, specifically person-to-person or via online media. These approaches cannot identify attacks and their associated behaviour outside of the criteria specified and therefore may be unable to address future semantic attacks or ones that combine exploitations from more than one category.

More recently, [66] introduced a dynamic classification framework that addresses high-level attack behaviours involved in the creation of a semantic attack: persuasion, fabrication and data gathering. This approach provides a basis for classifying threats outside of the limited scope of previous taxonomies, but is limited to high-level theoretical dimensions of Social Engineering and it is not clear how it can be used at a technical level by developers of semantic attack defence technologies. On the other hand, the work of [67] is highly useful in that respect as it identifies the entities that comprise an attack irrespective of visual representation, but their focus is solely on social networking.

2.1 Taxonomy of Semantic Attacks

Here, the aim is to help researchers and engineers develop technical defence approaches for both current and future semantic attacks by addressing core semantic attack characteristics rather than particular implementations. That is because a defence system designed for a particular set of characteristics can be applicable across all attack types that share it, thus making it a more efficient choice from the perspective of technical development.

Note that current cybercrime operations may have many layers and an adversary may employ composite semantic attacks consisting of multiple phases of individual semantic attacks, in parallel or one after the other [68]. An example may be a spam email containing a URL to a spoofed website, where the user is forwarded to a drive-by download that results in infection with a scareware application. The taxonomy classification is designed to classify semantic attacks as individual singular components (e.g., the drive-by download) rather than the possible permutations of these components (spam $\rightarrow$ spoofed website $\rightarrow$ drive-by download $\rightarrow$ scareware). This is an important function of developing a generic taxonomy of attacks that is practically usable at a technical level because it simplifies the objective of a defence mechanism. Assuming that a composite attack requires all its individual component attacks to succeed, mechanisms that address any of the latter would protect against the composite attack too. In the example above, the threat of the scareware application delivered in this manner would be thwarted by any technical mechanism that would effectively block the spam email, spoofed website or drive-by-download attempt.

To be usable in the long-term, these characteristics need to be universally applicable and independent of the platform involved. In particular the platform consolidation driven by cloud computing, the Internet of Things and other recent computing paradigms is making the distinction between mobile, desktop and embedded system user experience less obvious than in the past. For example, in the near future, fake tyre pressure alerts shown on a car's dashboard [69] may be used to achieve deception in a manner not too dissimilar to current scareware pop-up alerts for mobile and desktop users. A defence approach focusing on the fake warning deception characteristics rather than the platform targeted would potentially be applicable for both.

These characteristics also need to span all stages of an attack. For this, we adopt the definition of the three distinct control stages of orchestration, exploitation and execution suggested by [70]. For each stage we pose the questions that we believe would matter the most to developers of technical protection mechanisms. In addition, we have ensured that for each category, the classifications are mutually exclusive.

Based on this constraint we do not by any means claim that the taxonomy is exhaustive, but we do evaluate its practical usefulness by providing for each category an example of how a developer can find it useful.

Control Stage 1: Orchestration

1. How is the target chosen?

   This can help identify the conditions for exposure to an attack. For instance, whether a user is vulnerable due to a specific attribute of theirs or chosen randomly makes a difference for the user and system features that a defence system may have to focus on.

2. How does the attack reach the target?

This may help identify the platforms that are involved in the attack, which in turn may help a developer choose which remote (hence involving a network) or local system to monitor and potentially where to place a defence mechanism.

3. Is the attack automated?

   The degree of automation in an attack can change fundamentally the response mechanism or the type of data that can be meaningful to collect about it. For instance, a fully automated attack may be possible to fingerprint based on patterns of previously observed behaviour, while a fully manual attack may focus on the attacker's behaviour instead.

Control Stage 2: Exploitation

1. Is it looks or behaviour that deceive the user?

   This can potentially help the developer of a defence mechanism to pinpoint mechanisms by which an attacker can deceive the user into a false expectation by manipulating visual and/or system behaviour aspects of a system.

2. Is the platform used in the deception only (ab)used or also programmatically modified?

   Identifying whether the deception occurs in code (embedded in the system or external), or by abusing intended user space functionality, can help shape the design of a defence system by narrowing down its scope.

Control Stage 3: Execution

1. Does the attack complete the deception in one step?

   An attack that relies on more than one step can be potentially detected more easily than a single-step one and before it completes by looking for traces of its initial steps. It may also be thwarted by preventing even one of the compromising actions that a user needs to be deceived into committing.

2. Does the deception persist?

   Contrary to one-off deception attempts, persistent ones may have a high chance of succeeding in their target but could also help a learning-based defence system to gradually identify its pattern of behaviour and block it.

Each category of answers that correspond to each question helps establish the sections and subsections of the taxonomy, as shown in Fig. 2.1.

Utilising a single-layer attack model for classification, the taxonomy is able to identify the composition of a semantic attack by using a parameter-based approach, organising the classification criteria in a linear fashion. Unlike previous taxonomy approaches, it captures multiple variables involved in the delivery and execution of a semantic attack by applying criteria that are independent of the attack vectors used. As a result, it can potentially represent both existing and future attacks.

To supplement the taxonomy we produce a survey of the current landscape of defence mechanisms, as well as a comparative matrix between the classification criteria and applicable defences, which illustrates areas of semantic attack security most in need of research.

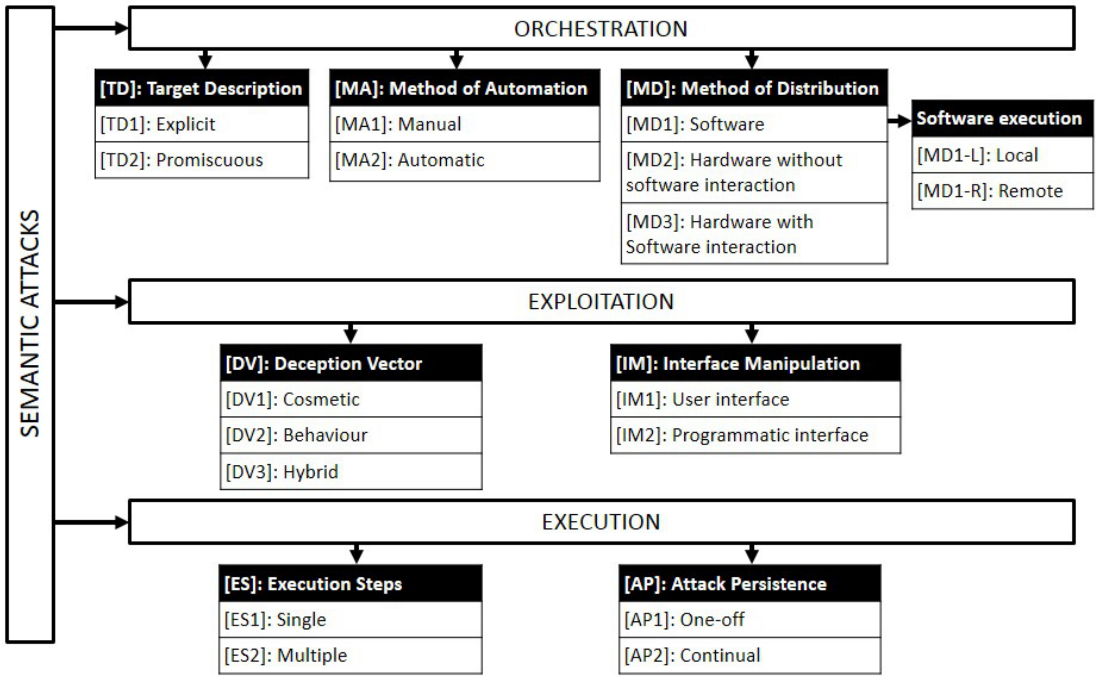


Figure 2.1: Taxonomy of Semantic Attack Mechanisms

2.1.1 Control Stage 1: Orchestration

The arrangement of targeting, distribution and automation in a semantic attack.

TD: Target Description

This refers to the targeting methodology applied in the attack orchestration process: whether the attacker has chosen to target an individual or group randomly (promiscuous targeting) or based on their identity or some other exclusive attribute of theirs (explicit targeting).

TD1: Explicit targeting

By explicit targeting, we refer to the practice of choosing a particular individual or group of users as the targets of an attack based on their specific identity or an exclusive attribute of theirs (company, role, location etc...). Typical examples here are spear-phishing and watering hole. Spear-phishing is the targeted version of phishing, where a carefully crafted phishing email is directed to a specific individual or organisation. This is known as whaling when the target is highly valuable (e.g., a senior executive) [71]. Watering hole is the targeted version of the drive-by-download attack, where a malicious script is implanted on the websites that a particular individual or community are known to visit [72]. In the same category of explicit targeting based on a common attribute we can include the practice of targeting victims located in a particular country, so as to intentionally cultivate botnets of better quality or to exploit that country's sociopolitical affairs [73]. For emphasis, let us clarify that we include here only attacks where the target is chosen explicitly for their identity or for an exclusive attribute of theirs, not merely as a second order of consequence. For example, a WiFi phishing attack in a coffee shop is generally not a targeted attack (see TD2: promiscuous targeting). The fact that people that are coffee drinkers are targeted is only a second order of consequence, as the attacker is highly unlikely to have explicitly chosen them because they are coffee drinkers. However, one could conceive a targeted instance of this attack (say a "WiFi spear-phishing" attack) launched specifically when a particular target individual is known to visit that coffee shop. This hypothetical attack would indeed be classified as explicitly targeted.

TD2: Promiscuous targeting

Promiscuous targeting is nondescript and typically focuses on maximum exposure. A typical example would be a large-scale phishing campaign launched by a botnet against any email address the attacker could find [74]. Semantic attack "worms" employing instant messaging and file-sharing platforms as automated replication mediums usually also aim for maximum exposure through promiscuous targeting [75]. Wifi phishing attacks [76, 77] are generally promiscuous in their user targeting. The attacker sets up a malicious access point to exploit any incoming client connection regardless of user identity or any other attribute.

MD: Method of Distribution

This refers to the mechanism by which an attack arrives at the target system: through a software interface (Software), executed on the target host (Software-Local) or on a distributed host environment (Software-Remote); or whether it is a physical device (Hardware) that interfaces with the target host via direct hardware access (Hardware without software interaction) or via the host operating system (Hardware with software interaction). Software interfaces are generally more common. Software-based attacks can be highly automated and easy to reuse and replicate with minimal supervision, while embedding a hardware device in a target environment can be manually intensive, high-risk, and difficult to target large number of users.

MD1: Software

This refers to semantic attacks that distribute a deception mechanism through a software medium, usually exploiting built-in functionality in the user interface of legitimate applications rather than technical flaws. They can be classified as local or remote.

MD1-L: Local

Local software attacks are distributed from within the target user's local operating system, as exemplified by attacks that use Trojan horse logic [78]. While a Trojan horse carrying a semantic attack payload may itself have been distributed remotely (e.g., unknowingly downloaded from a website), the semantic attack already resides within the target machine when presented to its target [58]. This behaviour is commonly observed in PDF file masquerading [79] and in scareware/rogueware attacks [76, 58].

MD1-R: Remote

Here, the semantic attack originates from a host environment other than the target user's system. It may involve web servers or distributed applications, such as cloud (for storage, server hosting or email) and peer-to-peer platforms (P2P networks, instant messaging, torrents). Remote distribution varies between different components in a remote platform. For example, a malicious URL may be presented via an email system on the Internet [18] or an instant messaging program [80, 81]. Malicious files can be served via synchronised cloud storage [11], malvertisements embedded in compromised web pages [82], automated bots in social network sites [83, 84] or a stolen online gaming avatar used to gather information on a computer gaming system [85].

MD2: Hardware without software interaction

This refers to attacks distributed to the target through a local hardware interface without interacting with the target host's operating system. Examples include hardware interfaces capable of direct memory access (DMA) or external hardware devices that passively intercept user data. In the case of DMA exploits, attackers plant common peripherals, such as storage devices, employing only hardware interfaces that are capable of creating a direct channel to a system's physical memory, such as firewire [86] or a PCI network interface card [87]. Attackers can also plant man-in-the-middle sniffing devices in the guise of Ethernet network outlet points, or hardware key loggers embedded in keyboards [88].

MD3: Hardware with software interaction

Most hardware-distributed semantic attacks rely on a software component in order to elicit compromising user actions. For example, [89] describe a series of attacks involving the USB interface, presented as a masqueraded hardware device that includes a software component, such as a malicious file triggering auto-run functionality in operating systems (namely Microsoft Windows). In the same category we can include exploitations that depend on WiFi, Bluetooth, as well as other less obvious interfaces for their distribution. An interesting example is near field communication (NFC) phishing. When attached to a legitimate surface (say a Red Cross donation appeal poster), a maliciously modified NFC tag interacts with the NFC software on users' smartphones and directs them to a malicious website rather than the Red Cross's. [12].

MA: Mode of Automation

Mode of automation refers to the degree of attacker supervision in the activation and administration of a semantic attack: by manually supervising every element of the attack throughout its lifetime (Manual), or by pre-configuring every aspect and having no control over the attack after it is launched (Automatic).

MA1: Manual

This requires intervention from the attacker to facilitate delivery of a semantic attack. The attacker explicitly triggers the deception mechanism by manually placing it in the target environment and, where applicable, explicitly responds to events when user interaction occurs. The attack procedure is flexible, because the attacker has full control over the process and may modify it during the attack. Examples could include an attacker manually sending a phishing email from a spoofed email account rather than an automated botnet, handing out infected DVDs outside a building, or physically intercepting legitimate media devices (e.g., through hardware or delivery supply chain) and embedding into them a malicious payload before they reach their target [90].

MA2: Automatic

Here, the semantic attack is delivered in an automated manner without requiring communication or intervention from the attacker. The attacker has pre-programmed the procedure for delivering the attack. Automatic mode of automation provides obfuscation of the attacker as the semantic attack operates without external influence, but also limits control and flexibility. For example, in the case of the Anna Kournikova "worm", the malicious email and file attachment contained the complete set of instructions for the attack [91]; once released the author had no control over its targeting or attack parameters. A more recent example is "Selfmite", a SMS malware for Android devices, which contains a hard-coded procedure to send text messages containing a link to an app. Once installed, it then replicates the message to the first 20 contacts on that device, effectively masquerading as a text from a known associate in an automated fashion [92]. In the same category we can include drive-by download attacks. Once an attacker embeds malicious code in a vulnerable website, the drive-by download runs automatically when a user visits the website [93, 94]. In general, full automation can help hide the origin of a semantic attack and reduce the effort required to replicate it, but also limits control over the process. Most automatic semantic attacks feature crude deception mechanisms that can be noticed by an experienced user.

2.1.2 Control Stage 2: Exploitation

The construction and application of attack vectors designed to bypass system information security.

DV: Deception Vector

The deception vector is a focal point of this taxonomy. It defines the mechanism by which the user is deceived into facilitating a security breach, and can be categorised as cosmetic, behaviour-based or a combination of the two.

DV1: Cosmetic

In the use of graphical user interfaces (GUIs), there is an implicit trust between GUI designer and user. The GUI designer trusts that the GUI will be used as intended, and the user trusts that each component is what it appears to be (e.g., in the sense that the GUI is used for the purpose of facilitating legitimate platform behaviour and not manipulated in a way that aims to deceive and exploit the host system or user). Cosmetic semantic attacks exploit this trust by manipulating the appearance of GUI components. For example, a file contains a name, type extension, icon and associated program. Filenames typically appear as readable, clear indicators of what a file does or contains, as is often required by the author. Therefore, official-looking and recognisable filenames command a level of trust. The attacks in [95] and [96] are prime examples where cosmetic deception techniques have proven particularly effective in P2P applications, either by matching filenames in a user's shared folder or by generating commonly searched filenames on the network. File extensions carry a presupposed risk associated to their behaviour. For example, ".exe" is a well known executable extension and thus threat to the user's system if the file is malicious. However, the same caution is not exercised for other executable types, such as ".bat", ".ini", ".lnk", or ".scr" [11]. Moreover, in Microsoft Windows, the file type and appropriate icon to be displayed are determined by extension. As Windows Explorer hides extensions by default and executable programs can be configured to display any icon [97], users can be deceived by an icon - extension mismatch if the filename is disguised, as in "document.txt.exe" displayed as "document.txt". In the same manner, emails contain sender addresses, subject titles, body of text containing images and attachments [18], and websites use a domain name URL and common page features, such as buttons and images, which can be manipulated to deceive a user. Attackers can generate phishing websites that assume the identity of their legitimate counterparts by copying the source code and images to appear completely identical [17]. In typosquatting attacks, simple but effective cosmetic deception is achieved by registering domain names that are similar to popular, legitimate websites, and are likely to be visited by users making spelling mistakes (e.g. twiter.com or facebok.com) [98].

Cosmetic deception is prevalent in attacks against web-based user interfaces, such as social media platforms where an attacker posts malicious links that visually imitate videos, enticing users to click on them to view the content [99]. In WiFi phishing, the attacker redirects users to an authentication website that looks like the login page for free Internet service [100]. The GUI components expected for such a page are included to add integrity and conformity to the exploit. [101] provide examples of techniques for developing fake mobile apps that appear like their legitimate counterparts, by duplicating login screen, including the logo, GUI layout and text. The study carried out by [102] has demonstrated visual logo spoofing, with manipulated "seal images", which fool the users but are otherwise seen as non-malicious content to a security system monitoring the web platform.

DV2: Behaviour

Here, deception is achieved by mimicking a legitimate system's behaviour rather than looks. Users are duped by supposed functionality convention against the standard approach used in well-known implementations. For example, in the generic rogue access point attack, the user is deceived by merely seeing it on the list of available WiFi networks [76]. Also, in URL phishing attacks on social network sites [80, 103], the deception relies solely on the accepted behaviour associated to the origin of the attack, as it is received from a user that is on the target's list of friends. In reality, the message is automatically generated by a malicious app installed on that friend's account.

DV3: Hybrid

Hybrid approaches combine aspects of both "looking like" (DV1) and "behaving like" (DV2) to create a convincing deception, as exemplified by phishing websites, which typically copy not only images and text, but also the actual code from the legitimate website [17]. [58] and [32] describe how scareware applications masquerade as antivirus programs by using similar logos, text and interfaces as their legitimate counterparts, complete with scanning engines and infection removal tools. [104] has reported an attack where a wrapper was attached to a legitimate mobile banking app. Once installed, it would load an in-app HTML page mimicking both the looks and the behaviour of the legitimate application's login form. On entering user credentials, an error would be displayed, requesting the user to install the application again. At that point, the legitimate version would indeed be installed and would function as expected from then on.

Other examples of hybrid deception can be seen in removable media attacks, such as USB flash drives. A USB storage device can be preloaded with a legitimate looking Microsoft word file that actually contains a malicious Visual Basic macro or zero-day exploit. Once the user has clicked on the file, Microsoft word opens as the user would expect, whilst also running the malicious code attached to the word document. A recent demonstration by [105] has shown how USB firmware controllers can be reprogrammed to spoof other devices to take control of a system, exfiltrate data or spy on the user. The exploit shows how a USB flash drive can assume different or multiple identities on a computer, such as a webcam, keyboard, as well as a USB flash drive, by manipulating USB interface identifiers used by operating system to interact with the USB devices functionality (e.g., processing key states sent by a USB keyboard peripheral). Here, the deception vector is both cosmetic and behavioural. The USB device appears and functions as the user would expect, but in addition it deceives the host system into thinking it is a different device and allows the execution of arbitrary code.

IM: Interface Manipulation

A deception vector can be exploited by only utilising maliciously the existing functionality of the target system (user interface) or by also programmatically modifying it (programmatic interface). For example, an attacker may direct users to an attack server by merely posting a hyperlink using a social network site's user interface or by programmatically injecting malicious JavaScript code into its HTML code.

IM1: User Interface

Here, we refer to attacks that are limited to abusing existing functionality provided by a user interface as the means of deceiving a user. This includes both hardware and, more commonly, software user interfaces. In malvertisement exploitations [82], attackers (ab)use the user functionality of an advertisement system on a website, by first posting a legitimate advertisement and then replacing it with a malicious version once it has gained popularity/trust on the host platform. In search-engine poisoning, attackers repeatedly place unrelated phrases in the body of text or URLs on a website, which when crawled by a search engine, return the attackers website in the results of a user search [106, 43]. In man-in-the-middle rogue access points, the attacker adjusts the user configurable service set identifier (SSID) to match that of a legitimate access point in the environment in which it has been installed [76].

IM2: Programmatic Interface

Here, we refer to attacks that are not limited to utilising existing functionality, but may also modify it, typically by exploiting vulnerabilities on the target system. This is prevalent in drive-by download attacks where the attacker injects malicious scripts (e.g., JavaScript) into vulnerable websites to redirect users to a malicious download/installation [107, 108].

A technique detailed by [109] involves compressing into a zip file an executable with special hexadecimal characters in the filename, so as to make it look like a Microsoft Word document when accessed with a particular popular zip archive viewer that has this technical flaw. Another example has been presented in [21] and [79], where the PDF language is manipulated to create a file that appears and behaves as a legitimate PDF file would, but when executed can launch a semantic attack requesting login credentials and redirecting the user to a malicious website. This is achieved by using certain special functions in the PDF language that once compiled run on file execution, such as embedding JavaScript which is automatically executed inside an unsanitised PDF object (which follows the PDF header as part of the PDF file format structure and is commonly used to declare a dictionary or document version labels).

2.1.3 Control Stage 3: Execution

Operational procedure during attack run-time

AP: Attack Persistence

Different semantic attacks implement varied levels of deception that are one-off or persist depending on the intentions of the attacker.

AP1: One-off

Here, once the user has triggered the attack payload, e.g. by performing an action that either grants access privileges or provides sensitive user data to the attacker, the semantic attack ceases any further action. Large-scale spamming is typical of this approach. Attackers mass mail a large address set aiming to capture user details through a spoofed website, which once gathered it redirects users to the legitimate site and disappear [18, 110]. Similarly, a drive-by-download attack completes on the download of the malware when the user accesses the infected website. The user remains vulnerable to the same infected website on further attempts to access it, but each time the drive-by-download attack completes on download.

AP2: Continual

In continual semantic attacks, a particular attack instance does not expire upon successful exploitation and the user continues to be exposed to its deception attempts. This may involve recurring and direct communication with the target by exploiting the messaging mechanisms provided by email, instant messaging, SNS, P2P networks, IRC forums and other Internet-based services [18, 80, 96]. A characteristic example would be scareware applications, which periodically display fake malware alerts for as long as they reside on the system [32]. Continual behaviour is common in semantic attacks that are designed to elicit financial gain from a target.

ES: Execution Steps

A single semantic attack may consist of one or more distinct steps.

ES1: Single-step

Single-step attacks require that the target user carries out only one action to facilitate exploitation (e.g. open a file, run a program, click on a web link or button). Once the initial action from the victim has been carried out, the attack payload is able to gain the necessary information or control for the attacker, at which stage the semantic attack is complete. Variations of the phishing attacks described in [18] rely on a single step. The less user interaction required the lower the risk of being detected. This is one of the reasons for the rise in drive-by malware, where the user's system is infected with a single click [108]. Other examples of single-step attacks can be identified in typical website and email-based phishing attacks [17], and more recently in malvertisements [82]. In the latter, the attackers place legitimate advertisements on trusted websites for as long as needed to gain good reputation and be trusted themselves. At that point, the attackers start placing advertisements that carry malware and infect all users visiting websites that display them.

ES2: Multi-step

In a multi-step attack, a user needs to be deceived more than once for the attack to achieve a meaningful result. Attacks involving multiple steps within the exploitation generally employ one step to gain access and an additional one to steal information or deliver malware. For example, in WiFi phishing, where the attacker generates a spoofed SSID of the local WiFi provider (e.g. a coffee shop's or hotel's), the user needs to first connect to this rogue access point (step 1), and then to provide his/her login credentials to the attacker (step 2) after he/she is redirected to a phishing login page for access to the Internet [100]. Similarly, in the usual typosquatting attack

[111, 98], the user must first navigate to the spoofed webpage with the mistyped domain name (step 1), and then click on a malformed download link (step 2). In [112], the phishing attack contains three separate steps for data gathering, payload distribution and deception.

Most semantic attacks that exhibit continual attack persistence (AP2) are multi-step. For instance, scareware [32] typically infects the target system via deception (step 1) and then periodically attempt further deceptions through fake alerts of malware detected in the system (a repetitive step 2).

2.1.4 Taxonomy Examples

Here, let us illustrate the use of the taxonomy with four detailed examples and a table summarising 30 different attacks that have been observed in the wild.

In a drive-by download attack (Figure 2.2), an attacker needs to first assess a website for vulnerabilities that can be exploited to embed malicious behaviour. Once a weakness is found, the attacker proceeds to insert malicious code into the website that will redirect a user into downloading malware. Attack automation is automatic (MA2) because it requires no intervention from the attacker after the malicious code has been embedded in the vulnerable website weaknesses. The software distribution is remotely presented to the user, on user navigation, via the infected website (MD1-R). Targeting is promiscuous (TD2) as the attacker has planted the exploitation to attack any user who visits the website (but there is also a targeted - TD1 - version of this attack: watering hole). The deception vector can be classified as hybrid because the website itself is legitimate and thus appears and behaves as expected (DV3). The attacker has programmatically manipulated the application platform (IM2) to embed the attack payload. The attack requires only one step of deception, which is that the user performs an action, such as clicking on a button, for the exploitation to occur (ES1). After this, the attack completes (AP1), but, of course, the user can be infected again by performing the same action again.

The usual implementation of WiFi Evil Twin phishing (Figure 2.3) requires two steps of deception (ES2). The attacker configures a rogue access point to advertise

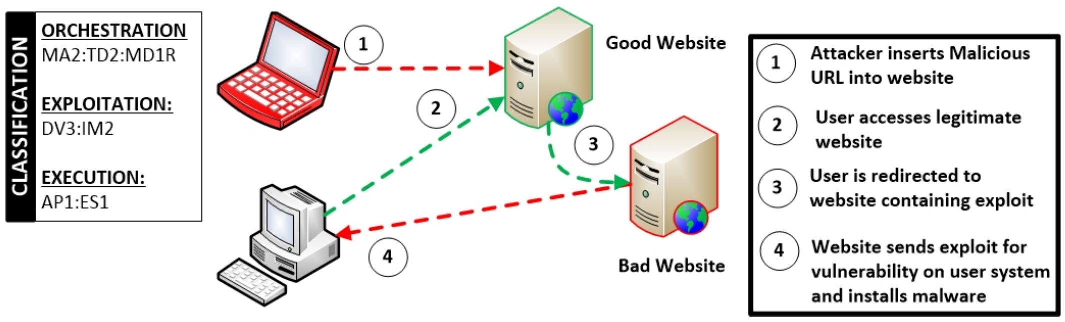

CLASSIFICATION
ORCHESTRATION
MA2:TD2:MD1R
EXPLOITATION:
DV3:IM2
EXECUTION:
AP1:ES1
Good Website
Bad Website
1
2
3
4
1 Attacker inserts Malicious URL into website
2 User accesses legitimate website
3 User is redirected to website containing exploit
4 Website sends exploit for vulnerability on user system and installs malware

Figure 2.2: Taxonomic classification of Drive-By Download

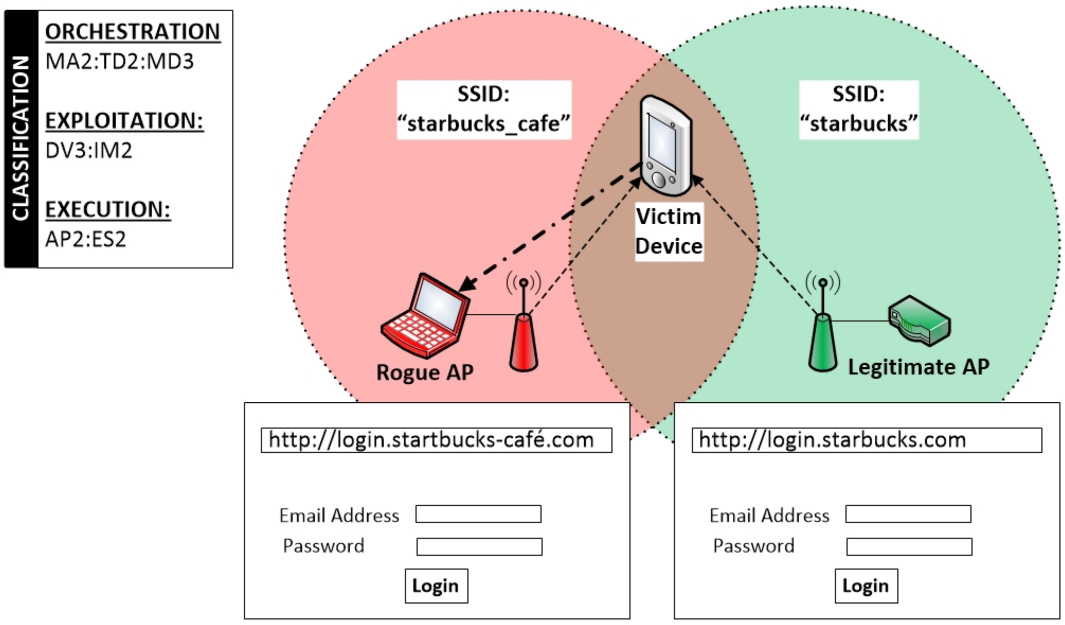


Figure 2.3: Taxonomic classification of WiFi Evil Twin Phishing

an SSID that is identical to a legitimate access point's in the target environment, such as a coffee shop or shopping centre etc., and also programmatically introduces a spoofed captive portal (IM2), which is automatically presented (MA2) to any user who connects to the Internet through it. The rogue access point both looks (identical SSID) and behaves in a manner that mimics the legitimate one (DV3). The rogue access point is distributed via a wireless hardware interface waiting for users to connect to the spoofed SSID. As software interaction is necessary to connect to the wireless network, this is classified as MD3. In the attack's usual implementation, targeting is promiscuous (TD2), directed to any user within range. The deception persists at least for as long as the user is connected (AP2).

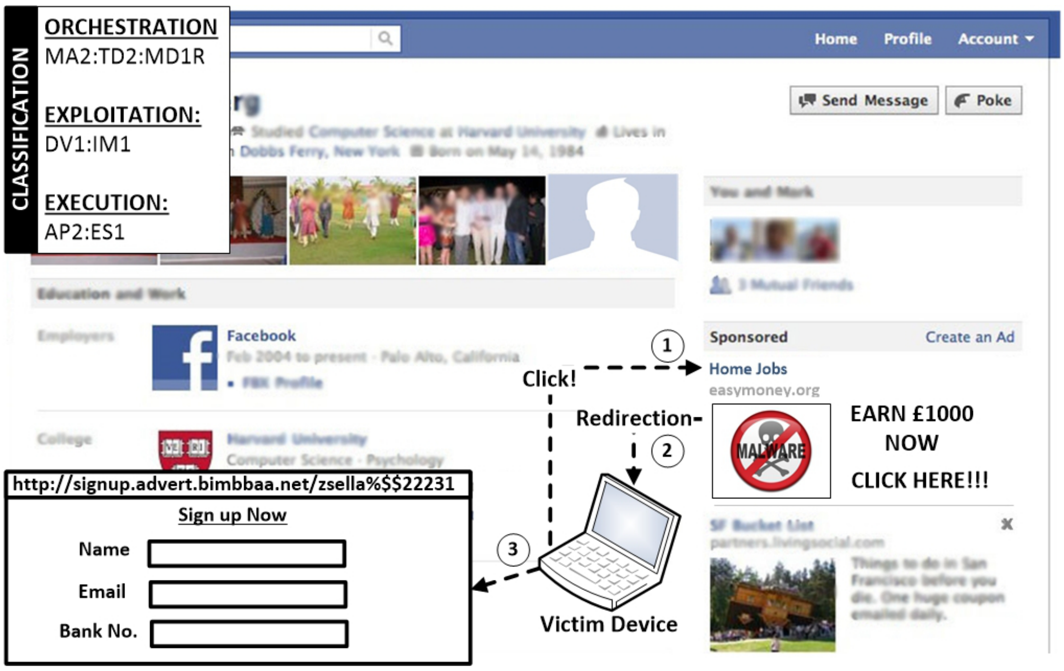


Figure 2.4: Taxonomic classification of SNS Malvertisement

Malvertisements in social media (Figure 2.4) are embedded by attackers after having previously posted a legitimate advertisement that gained in popularity, through user click volumes. Once that advertisement has gained enough reputation, the attacker replaces it with a malicious one, which automatically redirects users to an attack website (MA2). The attack targeting is usually promiscuous (TD2). Attack distribution occurs via the social media website and is forwarded automatically across user communities through in-built recommender and reputation processes on the social media site (MD1-R). The malvertisement deception vector uses cosmetic obfuscation through the social media system own carrier browsing system, which creates the appearance of its legitimacy as an advertisement by placing it in the advertisement pane, displaying a graphic and applying recommendation labeling to users (DV1, IM1). The attack persistence is continual (AP2), as after it deceives a user once (ES1), the malvertisement continues appearing in the advertisement sections of their social media profile.

In fake mobile applications (Figure 2.5), the attacker uploads a spoofed app that masquerades as a legitimate program in a mobile software marketplace (e.g. Google

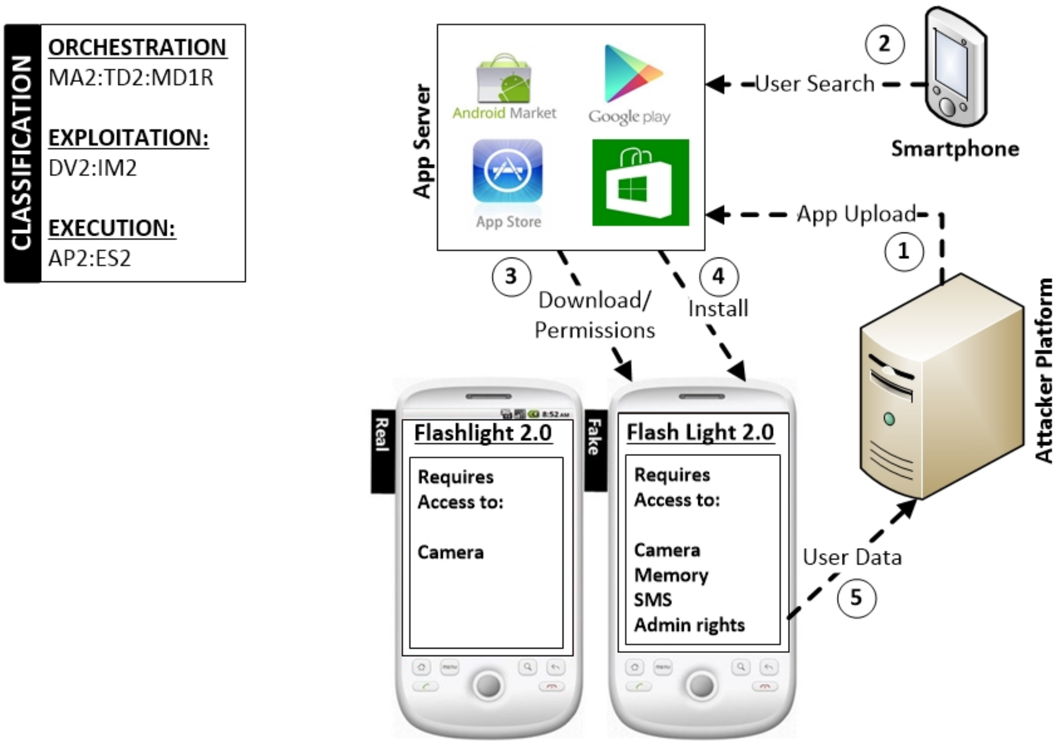


Figure 2.5: Taxonomic classification of Fake Mobile App

Play, Android, Windows Store, Apple AppStore etc.). The first step is to deceive a user who has searched for related keywords to download it. The second step is to deceive the user into accepting the request for granting excessive system privileges (ES2). Typically, attackers spoof popular types of applications with the purpose of gaining maximum exposure (TD2) before the app is identified as malware and removed. Depending on the implementation, deception can be based on looks, behaviour or both. In the example of Figure 2.5, a flashlight app downloaded from an app market features non-standard GUI and poor graphics, yet the user may still consider it legitimate if it operates the camera flash as expected (DV2), while in reality, it may be a Trojan horse stealing the user's private data in the background. The whole process is automated (MA2), and attack persistence is continual because the application resides permanently in operation on the mobile device until removed (AP2). The delivery of the deception requires programmatic manipulation (IM2) and distribution from the remote app server (MD1-R).

Having followed a similar analysis for a broad range of semantic attacks that have been discovered or are emerging, we have compiled Table 2.2 to summarise semantic attacks based on the taxonomy and observe common characteristics.

Typical Semantic Attacks in the wild	**Orch.**	**Exp.**	**Exe.**
Bluetooth Snarfing Attack: This is targeted at any users that leave their mobile devices open to accepting Bluetooth transfer as a default setting. The attacker sends to the target device a malware file with a filename that looks like it corresponds to a popular application (e.g. twitter, facebook, a banking app etc.). When opened by the user, it starts exfiltrating data back to the attacker system [113, 114].	MA1, TD2, MD3	DV1, IM1	AP1, ES1
Cryptovirus/Cryptotrojan/Cryptoworm: An attacker spreads a malicious application through email, infected programs (using a wrapper approach) and compromised websites. If the user is deceived into allowing the application to infect the user's machine (e.g., by opening the email attachment), it starts encrypting user data, such as files or the hard drive itself. At that point, it attempts a second deception, demanding payment from the user to release the private key that decrypts the data typically assuming the appearance of an official-looking letter from a government agency [115].	MA2, TD2, MD1-L	DV1, IM2	AP2, ES1
Drive-By Download: An attacker exploits vulnerabilities in a website's code structure (typically JavaScript) and proceeds to embed a malicious payload. The attacker then sends a link to this website, typically containing the website and full URL path to the exploited code, to potential targets (e.g. via phishing email, social media post etc). The machines of the users are infected when they open the link to the website [93, 94, 107, 108]. The deception takes advantage of the normal-looking cosmetics and behaviour of the affected website.	MA2, TD2, MD1-R	DV3, IM2	AP1, ES1
Fake Mobile App: The attacker uploads a spoofed app that masquerades as a legitimate program in a mobile software marketplace. Upon installation, the application requests full user or admin/root access to the device, such as access to volatile/non volatile memory, sending texts, making calls, or downloading other software. When the fake application runs, it typically exhibits no related functionality to the spoofed legitimate counterpart [101] (see Figure 2.5).	MA2, TD2, MD1-R	DV2, IM2	AP2, ES2
Forum phishing - manual: An attacker manually posts a message within a conversation thread on a web forum. This can consist of multiple messages posted in the context of the thread, with some being legitimate but others containing malicious links or images. Typically, the attacker must bypass CAPTCHA controls by manually entering a verification code designed to prevent web bots posting messages.	MA1, TD2, MD1-R	DV2, IM1	AP1, ES1
HTTPS Man-in-the-Middle Adware: An attacker intercepts a user's HTTPS requests and places a spoofed certificate within the root certificate store on the user's system. As a result, when a user send a HTTPS request the attacker redirects them to a phishing website that is incorrectly identified as legitimate through the spoofed certificate. A vulnerability facilitating this attack has recently been discovered to have been inadvertently introduced to Lenovo laptops through poorly written adware [116, 117].	MA2, TD2, MD1-L	DV3, IM2	AP2, ES1
Instant Message phishing - automated: Similar to email phishing, instant messaging phishing includes obfuscated malicious URLs or attachments. The URL redirects the user to a malicious website and the attachment typically infects the instant messaging application or installs malware on the user's machine. In the case of the former, the attack configures the user's instant messaging account to resend the phishing messaging containing the attachment or URL to the user's contact list [118, 80].	MA2, TD2, MD1-R	DV2, IM1	AP1, ES1

Malicious web pop-up: A malicious pop-up is generated through a browser or website application window that activates on user action, such as navigation, clicking or hovering over buttons, pictures and URLs. The window content typically takes on various approaches to deceive users, such as suggesting that they have won an online lottery and need to enter their bank details to collect the prize, or displaying a fake antivirus scan that identifies bogus infections on the user machine and advises download of a file to remove the infection [16].	MA2, TD2, MD1-R	DV1, IM2	AP1, ES1
Malvertisement: Malvertisements are embedded by attackers after initially posting a legitimate advertisement that gains in popularity on a website. Once the legitimate advertisement has gained popularity and is trusted by the hosting website, the attacker replaces the advertisement with a malicious one. Once clicked, a user is redirected to a drive-by-download or bogus questionnaire. The malvertisement is distributed through the legitimate advertisement system on the website without needing to programmatically manipulate it [82].	MA2, TD2, MD1-R	DV1, IM1	AP2, ES1
Multimedia Masquerading: In multimedia masquerading, an attacker embeds a malicious link in the form of a video or audio player, often displaying the video's first frame and title to strengthen the deception. Once clicked, the fake media redirects the user into installing a software application to play the purported media content or to a website requiring the user to input sensitive data, join a premium subscription service or complete a /questionnaire. Often the attack will propagate through social media platforms where the user, on completing a plugin installation to run the media content, gives the malicious adware permissions to re-post on the user's behalf. The fake media is then propagated to friends [119].	MA2, TD2, MD1-R	DV1, IM2	AP1, ES1
NFC Phishing: In NFC attacks, the exploitation is embedded in a modified NFC tag. If the user is deceived into scanning it, the user is presented with a request to download an application, which is malicious. The deception relies on the fact that NFC is a relatively new technology, which does not have a reputation of being used maliciously and the interface generally resides in a publicly accessible place governed by a trusted hosting authority, such as airports, shopping centres and schools [12].	MA2, TD2, MD3	DV3, IM2	AP1, ES2
P2P Malware: A very simple attack, whereby the attacker shares malicious files over a P2P application, giving them commonly searched file names. A user downloads the malware matching their search criteria and infects their machine by opening the file. Obfuscation is achieved by manipulation of filename, extension and possibly the icon displayed by the proprietary browsing system of the P2P application. Replication occurs by the malware sharing itself through the user's P2P application [95, 120].	MA2, TD2, MD1-R	DV1, IM1	AP1, ES1
PDF File Masquerading: A Portable document format file is manipulated by using special functions in the PDF language that when compiled execute malicious functions, such as requesting access rights and login credentials or redirecting users to an attack website. The appearance and handling behaviour, as represented by the operating system, appear legitimate [21, 79].	MA2, TD2, MD1-L	DV3, IM2	AP1, ES1
Peripheral Masquerading - USB: In USB masquerading, an attacker configures a USB compatible device, such as a flash drive, mouse or keyboard, to contain a malicious payload that executes on insertion in a computer system. Typically, this is achieved using the autorun features in the operating system, but zero-day attacks have also exploited vulnerabilities in the way an operating system handles files on USB devices. Importantly, the malicious peripheral functions as intended, so as to avoid raising suspicions while infecting the target system in the background [121]. In its usual implementation, this is a highly targeted attack (TD1), but a supply chain non-targeted attack (TD2) is also conceivable).	MA1, TD1, MD3	DV3, IM2	AP2, ES2

Peripheral Masquerading - Firewire: A hardware deception attack where the deception mechanism is presented as a physical peripheral utilising the firewire interface. This can be in the form of a memory stick or external media reader. Once the user physically connects the hardware device via the firewire interface to the target host, the malware in the device exploits firewire's direct memory access capability to steal user data or inject malicious data into the system [86].	MA1, TD1, MD2	DV3, IM2	AP2, ES1
Phishing Website: An attacker spoofs a legitimate website for the purposes of stealing user information or duping them into installing malware, usually by duplicating both appearance and behaviour.	MA2, TD2, MD1-R	DV3, IM2	AP1, ES1
Ransomware: Ransomware is malware, similar to cryptovirus (without encrypting files), typically delivered through a Trojan horse, that blocks access to a system by taking control of a browser of the operating system itself until the user pays a sum of money to unblock the machine. The ransomware attempts to convince victims that they are the subject of a criminal inquiry, from a law enforcement authority (using official looking logos and pictures), for ownership of illegal software or pornography. The user is then advised how to unblock the computer by following instructions for paying a fine electronically [122]. The attack is continual for as long as it takes the user to pay the money demanded by the software, after which the access restrictions are lifted.	MA2, TD2, MD1-L	DV1, IM2	AP2, ES2
Rogueware: Similar to a scareware application, rogueware is a standalone application, downloaded from a malicious website or as a wrapper/adware alongside a legitimate file. It masquerades as a well-known program, (such as a media player or Internet connection optimiser, which when executed attempts to steal the user's confidential data [58]. The rogueware continually deceives the user by running as a application on the user host system until removed.	MA2, TD2, MD1-L	DV3, IM2	AP2, ES2
Rogue Access Point: An attacker installs a wireless network access point in public or private environment. Its aim is to harvest connecting users and perform man-in-the-middle attacks through network data sniffing [76].	MA1, TD2, MD3	DV2, IM1	AP2, ES1
Scareware: A malicious computer program designed to convince a user that their system is infected with malware, pressuring the user into buying and downloading further (fake) antivirus applications (which are typically malware, spyware, adware etc.). Commonly, the protection software periodically displays wwarnings for infections and demands payment for licensing in order to remove them.	MA2, TD2, MD1-L	DV3, IM2	AP2, ES2
Search Engine Poisoning (Spamdexing): In search engine poisoning, a search engine is manipulated, so as to include a malicious website among the top-ranked results for given search terms. The users trust that since it appears high on the list of a trusted search engine, that website must be popular and very relevant to their keyword search, hence not suspicious. However, the attacker can force search engines to associate keywords with websites (that are malicious) in several ways, such as flooding websites that accept simple user generation content with specially crafted URLs [106, 43].	MA2, TD2, MD1-R	DV2, IM1	AP1, ES1
SMS Worm (Selfmite): In the ”Selfmite” SMS worm an attacker configures a text message to contain a web link to a recommended app that once downloaded by a user proceeds to send the same text, including the obfuscated link, to the first 20 contacts in the user's phonebook; thus spreading effectively by originating from a trusted source [123].	MA2, TD2, MD1-L	DV1, IM2	AP1, ES2

Spam Phishing Email (Botnet-generated): An email with a spoofed sender address and an enticing subject title and message requesting the user to download an attachment (which contains malware) or click on a malformed URL. Large-scale phishing attacks are typically distributed via botnets, which crawl the Internet for email addresses and are configured by botmasters to send template-based emails with varied malicious content. The attacker uses official-sounding language and logos to create a cosmetically appealing and authentic communication. Occasionally, spam phishing emails contain payloads for compromising the user's system and adding it to the botnet [18, 124].	MA2, TD2, MD1-R	DV1, IM1	AP1, ES1
Spear-phishing Email: Identical in format to a spam phishing email, however highly targeted toward a specific user, organisation or demographic, whereby the email subject, body of text and sender address are designed to take advantage of an existing relationship with the target. For example, the sender address may appear to originate from a friend and there may be an attachment appearing to be a photo they would be likely to receive from that friend. Spear-phishing is typically a manual process, as the attacker needs to be meticulous in the email construction to create a communication that is highly-specific to the intended target [124].	MA1, TD1, MD1-R	DV1, IM1	AP1, ES1
Spear-phishing Email - APT: This is an advanced form of spear-phishing employed by advanced persistent threats (APTs). Highly targeted emails are first sent to harvest login credentials for Microsoft Outlook accounts using dummy User Access Control prompts. A compromised account of one employee, customer or partner is then used to send spear-phishing emails to other company insiders. The attacker periodically injects a malicious message containing previously exchanged Microsoft Office documents that embed hidden malicious macros into an ongoing email discussion among multiple people (to improve the exploit success factor). E-mails are sent from the accounts of people the target knows adding to on going discussions already taking place. The attacker will blind carbon copy (bcc) other recipients to further obfuscate the malicious e-mail [125].	MA1, TD1, MD1-R	DV3, IM1	AP1, ES2
Tabnabbing After a user unknowingly navigates to a malicious website, embedded malware on the latter detects when the page has lost its focus and attempts to deceive the user into thinking they had left a Gmail (or other popular website) tab open. For this, the attacker needs to display the Gmail favicon (short for "favourite icon") and a convincing title, such as Gmail: Email from Google, as well as a spoofed Gmail login page by adjusting Javascript code within the website. The spoofed gmail login page is used to steal the login credentials before redirecting the user to the real Gmail website. As it is most likely that the user had never really logged out of Gmail, the direction makes it appear as if the login were successful and, thus not suspicious [126].	MA2, TD2, MD1-R	DV1, IM2	AP1, ES1
Typosquatting (also known as Cybersquatting): User mistypes a domain name (e.g, "twiter.com" instead of "twitter.com") into a browser and lands on a phishing website, which has copied the appearance of the legitimate, intended website. Typically the website appears identical, but does not have the same functionality. An example would be a social media website with a login prompt. Once the user enters their login details into the phishing website, they are redirected to the legitimate website after their personal data have been stolen [111, 98].	MA2, TD2, MD1-R	DV1, IM1	AP1, ES2
Visual SSL Spoofing: In SSL spoofing, the attacker configures a website to imitate specific parts of the browser interface, which a user would expect to display SSL/TLS connectivity status. By using official-looking logos or address bars with the typical padlock icon demonstrating encrypted communication, users can be fooled into believing they are on a secure website and entering sensitive information which the user will steal [127].	MA2, TD2, MD1-R	DV1, IM1	AP1, ES1
Watering hole: A highly targeted version of the drive-by-download attack, whereby an attacker embeds malware on websites that are visited by a specific individual or group of people. Eventually, the user/group will be exposed to the exploitation in one of the websites they regularly visit and trust. Here, the attacker is persistent in the attempt to exploit the chosen individual/group by infecting multiple websites they are known to visit [72].	MA2, TD1, MD1-R	DV3, IM2	AP1, ES1

WiFi Evil Twin: Similar to a rogue access point, on top of the rogue AP the attacker spoofs a nearby SSID of another WiFi service whilst also programmatically duplicating any web-captive portal (typically an authentication page) that exists on the legitimate access point in the environment. Once a user connects to the spoofed access point, the attacker intercepts all traffic, syphoning off credentials, files and sensitive information [100, 101, 128, 77]. An advanced version of this attack uses the snoopy platform that can be embedded within a UAV drone [129].	MA1, TD2, MD3	DV3, IM2	AP2, ES2

Table 2.2: Taxonomic Classification of Semantic Attacks

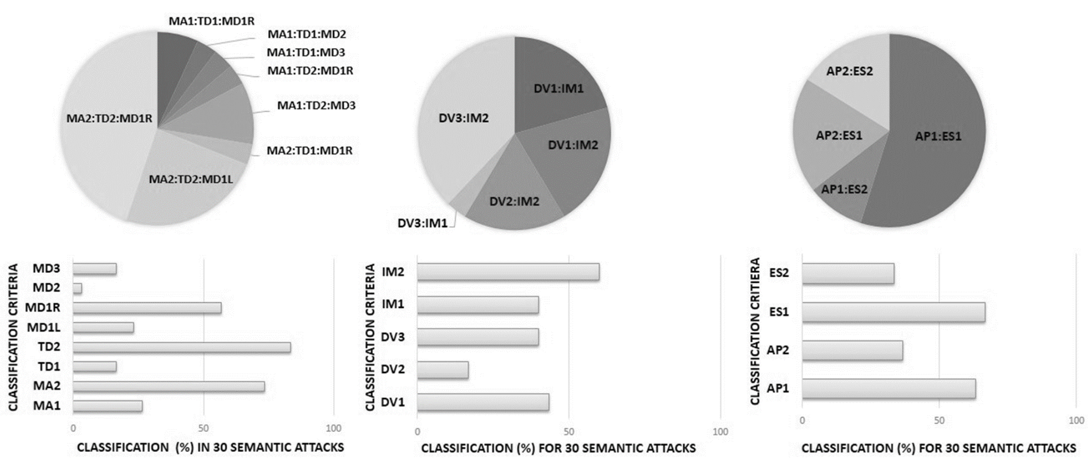


Figure 2.6: Taxonomic statistics of Table 2.2

The taxonomic classification of Table 2.2 can help identify the characteristics that are shared between different attacks (Figure 2.6). For example, it is evident that in terms of orchestration most attacks exhibit promiscuous targeting (TD2), software-based remote distribution (MD1-R) and automation (MA2). A defence approach that would thwart this orchestration pattern would be applicable to almost half of the semantic attack types listed here. In terms of exploitation, the vast majority rely on cosmetic deception alone (DV1) or in combination with behavioural deception (DV3). Approaches that would detect or prevent mismatches between the appearance and function of GUI components on websites and applications would reduce considerably the semantic attack surface.

The classification also shows the greatest strength of semantic attacks. Most require no more that one deception step (ES1), which means that a single error in judgement on behalf of the targeted user is sufficient. This is exemplified by spear-phishing attacks used to gain a foothold in target systems with otherwise strong technical security measures, from financial institutions and nuclear facilities [15], to the computers of high-value individuals. Even a prominent cryptographer with several computer security patents in his name can be deceived into clicking the "accept" button on a spoofed LinkedIn invitation from a non-existent employee of the European Patent Office [130].

2.2 Survey of Semantic Attack Defence Mechanisms

The impulsive and unpredictable behaviour of human users has been widely characterised as one of the most significant weaknesses in modern computer systems [131]. While security controls against technical exploitations have improved considerably in recent years, the same successes have proven difficult to replicate in user security.

Training and experience can certainly help but are neither easy to acquire nor sufficient [132, 133]. The landscape of semantic attacks is wide and continuously expanding. While user targeting has been historically limited to emails and websites, the advent of the Internet of Things [131] is expanding the scope of exploitations to even include smart home appliances [134]. Current systems have limited capability to compare a spoofed file icon with the associated extension or to efficiently highlight the execution similarities of different file extensions to users in a pro-active fashion. Similarly, users rarely inspect their computer environment before performing actions that may compromise their system [7, 17, 16]. The following survey describes an overview of mechanisms that can protect users against semantic attacks.

2.2.1 Organisational

2.2.1.1 Policy and Process Control

Policy and process provide hierarchical control through management frameworks as opposed to technical systems. Designed at lowering exposure to semantic attacks, well maintained policy and organisational procedures help to mitigate and significantly lower the risk of a potential exploit occurring, without relying on the technical capabilities of users [132, 133, 135].

Policy and process control is defined around the business and user environment, as appropriate to their specific needs, for organisational and personal use. However, security frameworks introduced to address semantic attacks have been added as bolt-ons to the wider technical threat, rather than embedded into the governance and strategic development of policy and process control. [132] reiterates this approach as a major vulnerability, insisting policy should be intrinsically built with people management and therefore embedded in the core of any information security framework. The prevalence of semantic attacks has resulted in a higher degree of applicability within policies that are integrated in internationally standardised frameworks ([136, 137, 138]). Dedicated governmental guidelines have also been proposed ([139, 140, 141, 142]) as well as detailed methodologies and investigations in guide books and research ([143, 144]). The book published by [143] offers a comprehensive collation of strategies for implementing secure policy and process for IT governance; combining standards and best practice guidelines for integrating information security across organisational domains i.e. user roles and responsibilities to embedded data security tools. [144] have developed a user centric, holistic framework specifically targeting phishing attacks. The framework includes a user, "Human Firewall", layer in the information security framework, enabling cross communication between defence dimensions (the user, organisational policy and technical controls) by combining mechanisms such as user awareness and training with policy and linked technology tools.

Current frameworks focus on preemption through layered control mechanisms, as opposed to being proactive and dynamic to change. Policy and procedures need to be flexible to unknown and unforeseen attacks and therefore appropriate to the changing threat landscape. Fixed guidelines can quickly become out-of-date as new attack methods are constantly being developed. Furthermore, as a personal tool, policy and procedure can be ineffectual. Rules and regulations designed to enforce and maintain security are too large in scope, almost entirely irrelevant to the personal user and their specific use of computer systems and the internet.

2.2.1.2 Awareness Training

As user deception is the primary attack vector, susceptibility to semantic attacks can be somewhat reduced with user awareness training. Education is a core component of the defence-in-depth model and in the instance of semantic attacks relates to areas where technical security mechanisms have proven to be inadequate. Interactive training maximises learning and awareness, such as bite-size quizzes, tests or games [26, 145, 146, 147], which when applied periodically provide education on the characteristics of different attack types. A large proportion of research into awareness training has focused on the approach of content, the methodology of its delivery and how data gathered from testing and formal application can be used to shape security policy and user training programmes [148, 149]. [150], [151] and more recently [152] have conducted research to measure the effectiveness of awareness systems built into browser security systems, including security toolbars, domain highlighting for suspected phishing and HTTPS/SSL/TLS connection indicators. Their results showed that none of these methods were effective because users would not understand or pay attention to warnings. Similarly, the more recent work of [153] has shown that security images meant to provide visual confirmation of website authenticity in Internet banking have also proven unreliable, as 74% of the study's participants entered their password after the security images were removed.

An analysis of user awareness indicators in a role-playing survey carried out in [154] demonstrated that the attitude of users against phishing attacks correlated highly with certain demographic characteristics. For example, the age group 18 to 25 was shown to be more susceptible to phishing than other age groups. Furthermore, once educational material was provided against a demographic, users were 40% less likely to enter information into an attack website. Interestingly, this was shown to be counter-productive in a few instances where users avoided legitimate websites too.

A study carried out in [155] suggested using the familiar concept of live “capture the flag” competitions also for measuring and teaching “cyber situational awareness” to users faced with security threats, including semantic attacks. Their competition involved 900 students spread across 16 countries. While undoubtedly interesting and useful, there was no mechanism for measuring which of the students’ skills and to what extent were improved by participating in the competition.

Other research has approached awareness training by exploiting the interactivity observed in gaming and its educational impact and value, with development and integration of technology trends such as Social Networking. In [156] and [157] Social Networking Sites are used as a platform to both deliver awareness applications to maximise exposure, exploitable functionality such as automated propagation through friend activity notifications and viral trends. Furthermore SNS were able to provide the means to measure effectiveness of applications in a social media environment by monitoring the response of subsequent user activity such as public/private access to profiles. Other developments have taken awareness efforts further through web-based/portal training systems, even by introducing proxy based training such as ”awareness middleware” for access to the internet [158]. The middleware in this sense is a proxy training portal, responsible for brokering a connection between the user and internet access. Only after users successfully complete a training quiz/test for improving security education and risk awareness will access to remote resources be permitted. [159] and more recently [160] have developed training platforms to create awareness and better understanding of the dynamics of semantic threats, by utilising virtualisation as a collaborative and interactive training platform.

A study conducted by [161] has used neurological scanning to determine how “habitation” (an automated response based on a reoccurring theme) occurs when users are presented with common, static security warnings, such as HTTPS certificate warnings. The research analyses brain behaviour through functional magnetic resonance imaging, where users were shown to exhibit a drop in visual processing sections of the brain after multiple exposures to a similar security warning. As a solution, the researchers suggested security warnings that are polymorphic, constantly changing appearance and behaviour each time they are generated (e.g., different colour, text, location, size, etc.). This was shown to reduce automatic responses and influence higher cognitive responses.

A recent study by [162] identified the neurological indicators associated with user response when exposed to semantic attacks. They showed that users exhibit notable brain activity associated to decision-making, problem solving, attention and comprehension, when security scenarios arise. The study was able to show how individual personality traits can influence decision making results, which can be used to improve training and awareness programmes. For example, training users to respond assertively to malware warnings may prove ineffectual outside a controlled environment for users that exhibit highly impulsive behaviour.

Novel suggestions for creation of a user risk-detection tools [163, 164], can be used to build and define user awareness and develop training policies. The experiment conducted in [165] has demonstrated how participating organisations can identify the level of user security awareness, through targeted phishing attacks. Whilst this type of experiment can yield very useful results, they can also be disruptive to an organisation's daily work and may have to be abandoned early (as was the case in the particular study). Furthermore, results need to be followed up through continually improved awareness programs and efficient user risk measurement. These tasks are manually intensive and their long term benefit may not be obvious to organisations. Awareness training approaches certainly have an impact, but the extent of this impact is uncertain, as there is no reliable approach for measuring the lasting effect. Primary evaluation is traditionally based on supervised testing in specific case studies [166, 167]. The extent to which different awareness training strategies employed by organisations impacts their quarterly or yearly breach statistics is not measured.

2.2.2 Technical

2.2.2.1 Sandboxing Mechanisms

Sandboxing is the process of creating an isolated computer environment, typically through virtualisation, to test untrusted operations such as those observed in unverified and untested code or programs. It has been effectively implemented as a security solution in various fields of computing, from specific code platforms [168] to browser systems [169], and in the field of smartphone security to improve defence against malicious software [61]. Proprietary applications, such as Adobe Acrobat X, have also implemented their own sandbox engine for enhanced security [170].

A sandboxing environment can check and analyse characteristics associated with an attack by categorising behaviour for anomaly detection [62]. Further sandboxing methodologies can provide an assured integrity checking mechanism [171, 172]; this approach relies heavily on the accuracy of behavioural analysis algorithms for detection and introduces the issue of increased resource demand. In the context of semantic attacks, sandboxing offers limited control, especially as it handles the execution of code, rather than user interfacing. A technical attack using visual deception methods instead of technical mechanics to deceive users would need to be understood by the sandbox, but this is a very complex task requiring sampling of graphical interpretation. Web browsers [173, 174] have implemented sandbox technologies to prevent websites from manipulating the browsers' own visual and behavioural properties, such as the URL address bar, browser tabs and security indicators. However, these mechanisms are still unable to prevent the execution of spoofing attacks, such as those creating a fake address bar [18, 175] using code that appears legitimate to the sandbox, but is visually deceiving the user. Instead, the limited research conducted in the application of sandboxing for semantic attacks focuses on its value as a learning tool. [176] utilised a sandbox environment to gather data used in enhancing user security training.

The Windows and Linux operating systems implement a number of sandbox mechanisms to prevent unauthorised changes being made to a system; acting as a good line of defence even when a user may have unknowingly been deceived by a semantic attack. One such example is the secure-attention-sequence or secure-attention-key as it is better known, which is a secure method of preventing applications spoofing an operating system login process. The login system is sandboxed from potential threats by using the operating system kernel to suspend all running processes before it is initiated. The user access control (UAC) system is another Windows sandbox mechanism that uses the user interface privilege isolation (UIPI) to prevent applications from unauthorised privilege escalation during execution. This is achieved by isolating processes marked with a lower integrity level (e.g. unsigned, untrusted third party application) from sending messages to higher integrity level processes (e.g. to request an action that might require root system access), until granted by an authorised user in the UAC prompt or valid certificate from an approved code-signing authority [177]. This approach can certainly limit the impact of some semantic attacks, but is specific to the operating system and cannot observe requests made to a remote system that it has no control over.

A system titled "BLADE" (block all drive-by exploits) developed by [178], provides a sandboxing mechanism to mitigate drive-by download exploitations. BLADE introduces a browser-independent operating system kernel extension, which enforces the rule that executable files must originate from an explicit user action providing consent. In the case of a drive-by download, where there is normally no explicit user action involved, the system redirects to a secure sandboxed location on the user system disk where execution is prevented. This approach is agnostic of the mechanism for deception or obfuscation and whether it is a zero day threat, as it responds only when a download occurs without specific user consent. In testing, BLADE was able to block all drive-by download attacks against a sample of over 1900 active malicious URLs with a minimal effect on system performance. To date, there is no further information regarding its practical implementation and evaluation.

[179] and [180] offer commercial sandboxing environments that integrate into existing operating systems. These applications install an abstraction layer for handling

the execution of programs and files (user initiated or otherwise). Whilst not exclusively aimed at providing protection against semantic attacks, they offer a last line of defence where a malicious application may have found its way on a user system. Requests made by all executing programs are analysed in real-time and categorised as either a threat or legitimate, and are then handled appropriately.

The open source project "Qubes OS" [181] has developed an operating system that utilises sandboxing to logically separate different applications on a system from infecting another, should one particular element become compromised. This is achieved by separating different applications into virtual machines that are isolated by security domain, e.g. work/personal. The system also integrates graphical properties, such as red app window frames for untrusted and green for trusted, to visually identify applications in different sandboxes. Whilst this functionality does not necessarily block a semantic attack from occurring in the sandbox itself, it can help inform users whether they should trust the behaviour or appearance of application based on the specific sandbox's security domain.

Recently, [182] have developed a tool for the android operating system designed to prevent malicious apps executing cosmetic (DV1), behavioural (DV2) or both cosmetic and behavioural deception (DV3). This can include spoofing an applications appearance or generating misleading visual cues on top of legitimate applications by capturing and analysing application programme interface (API) calls to the android graphical user interface. The researchers employ a method call "static code analysis which interrogates application bytecode, based on a system proposed by [183], during run-time to scan for requests made to specific Android API's which can enable the execution of visually deceptive components into an android systems graphical interface. The method can be applied for both preemptive and proactive protection of the android interface by vetting applications before release or by detecting this behaviour on the host device during operation. Whilst the approach was shown to be successful at identifying applications that use these APIs to execute cosmetic deception, the researchers also found that there was a significant rate of false positives. In response, they supplement the sandbox mechanism with a visual reputation system to confirm the identity of applications which use these APIs legitimately and those that do not. Crucially, in an evaluation involving 308 users, this approach was shown to significantly improve a users ability to detect malicious applications.

2.2.2.2 Authorisation, Authentication and Accounting (AAA)

Authentication, authorization, and accounting (AAA) is a framework for intelligently controlling access to computer resources, enforcing policies, auditing usage, and providing the information necessary to bill for services. It is usually implemented in controlled environments, especially where there is a varied user landscape posing a risk to the control and protection of data. The framework provides controls for access to resources (Authentication), organisational policy enforcement (Authorisation) and auditing of resource usage i.e. login times, session length, devices accessed (Accounting). Its use aims to ensure that organisations have a detailed level of assurance and control over who has access to a system, based on data on names, roles, skill sets etc. In large deployments, such as organisation domains and networks, AAA can be effectively managed through supervision, ensuring extended identification and delegated access is in place to run executables, access shared folders etc. However, in a home user environment, AAA is less practical because the central supervising authority remains with the user [184]. The capabilities of the individual require correct usage of "least user rights" and administrator privileges where appropriate to protect the system and secure data.

Many technical, autonomous, session based protocols such as Kerberos [185] have been successfully combined with secure centralised authentication and authorisation platforms [186]. Such solutions have proven effective at managing AAA assurances, but governing configuration rules are statically implemented and lack interactivity and dynamic visual interfacing with users once access to a system has been established. On the other hand, security platforms such as Microsoft's User Access Control not only indicate to users that an action may require more privileged rights but it actually displays when a program is trying to gain these rights to run and it asks the users to identify themselves beforehand [184]. One can then make an informed decision.

However, the system still relies on the user's due diligence in correctly selecting use access control options. Advancement in identity management, such as Microsoft's Forefront Identity Manager [187], have developed comprehensive AAA security that has a wide range of application effect at preventing semantic attacks. Yet, although they provide flexible control over users and access to resources, Identity management systems are not dynamic in their mitigation mechanisms. With regards to the growing threat of SNS-based semantic attacks, [188] have identified a range of AAA flavoured mechanisms that can provide some protection against automated Sybil attacks (an attack wherein a reputation system is subverted by forging identities in peer-to-peer networks) [19] and other current threats. Portable identities, with mutual verification and authentication between open systems (e.g. Facebook, Twitter, LinkedIn), limit the ability to abuse SNS functionality. This is suggested as the main challenge to address as it maintains usability without introducing difficult to use, user facing security controls. This area still lacks working examples and complex semantic attack SNS exploitations remain a major problem. For AAA frameworks to be able to address semantic attacks effectively, there needs to be a shift towards more pro-active measures, rather than relying almost entirely on sound management and individual user security awareness against system generated warnings.

2.2.2.3 Monitoring

In this context, by monitoring we refer specifically to the observation of computer system behaviour, generated by user/programmable actions, via collection, aggregation and analysis mechanisms. Monitoring is a key security mechanism for semantic attacks as new exploitations can be classified through exposure and effective security controls introduced through understanding of compromising user actions. Historic monitoring has required heavy supervision and forensic investigation, whereas new developments allow semantic attacks to be identified and addressed through use of Honeypots [189, 190]. These virtualised environments allow security practitioners to analyse exploitations, classify attack characteristics and identify the user behaviour that can lead to compromise. Research conducted in statistical monitoring focuses on developing semantic attack masquerade aversion by modelling user search behaviour. For example, [191] have introduced a novel method of monitoring and machine learning in order to identify anomalous user behaviour on a given system. The work conducted by [192] proposes adding dynamic adaptation to crime scripts (a sequence of actions observed in a semantic attack), which can model user/attacker behaviour in a given scenario and can be used in simulation, monitoring data output for collation in order to simplify the process of enumerating attack procedures. A promising piece of recent research in this area by [193] has developed malicious website detection by monitoring the redirection path taken to reach a web destination, instead of analysing physical website features.

[194] have introduced "SURF", a browser extension designed to detect search engine poisoning by monitoring "search-then-visits" user sessions and classifying redirections into legitimate or malicious. The particular system was able to achieve 99% accuracy in the researchers' empirical real-world evaluation.

[195] have topologically mapped dedicated hosts on malicious web infrastructures, involved in orchestrating redirection paths to malicious websites. The research used a PageRank algorithm to monitor and capture attack hosts responsible for traffic brokering and exchange for malicious activities. This study focuses on the management overlay that cybercriminals implement to conduct and control large-scale malware distribution. As part of the study to identify redirection chains, the researchers used a system called "WarningBird" [196], which is a URL detection system for Twitter, designed to protect against conditional redirection, where the attacker redirects web crawlers to a legitimate landing site, but a user to the attack location. In combination with a real-time classifier, WarningBird exhibits a high degree of accuracy against large samples of Tweets.

2.2.2.4 Integrity checking

User and application/data integrity is difficult to assure without proof or analysis, especially if the data and subsequent behaviour characteristics originate from an external network, colleague or friend. Integrity checking provides the user with a visual response and technical assurance as to whether the file, website, or data should be trusted [27, 28]. Systems such as TripWire implement File Integrity Monitoring that continuously monitors the integrity of a data e.g. a file a based on its properties and behaviour, and further work has shown how this defense mechanism can become dynamic [197]. A study carried out by [198] evaluates methods for file type classification, for the purposes of identifying and detection file masquerading attacks. The research evaluates file type detection algorithms such as FHT (File header and trailer), MDA (Multi-Discriminant analysis) and CFD (Compound file detection) in order to identify the true file type. Using this approach the researchers are able to classify fragments of a file or embedded files to determine a file type; thus providing a key mitigation mechanism for common file masquerading attacks. Research conducted in [199] shows how using simple methods to determine integrity, by checking and comparing visual similarity-based phishing can provide accurate and effective results. The researchers demonstrate that by analysing the similarity between visual components in a website, without prior knowledge that it is an attack platform, they are able to automatically determine specific templating of websites that have been spoofed by a large degree of phishes sites.

Recent studies have proposed solutions for detecting the integrity of URLs (using lexical tokens, network and web-based features) [200] through optimisation algorithms (e.g., Ant Bee Colony algorithm) and image detection techniques to mitigate rogueware [201, 202]. Integrity tools have also been developed to provide protection against WiFi "Evil Twin" access points, by analysing key components related to the appearance SSID-mac address mappings as well as hashing and cryptographic algorithms being used [203]. Future research addressing semantic attacks can consider digital signatures or certificates that originate from a central authentication system to prove legitimacy of data presented in a file or application (website, email, utility programs), where the data is uploaded, analysed and approved, so that where data are missing from this central authority or using unknown certificates, there is reduced functionality until it is properly verified for secure operation in its respective environment (domain, shared platform/desktop, folder, cloud storage etc). Social media platforms have been plagued by “Sybil” attacks (where in the context of a social network / media site a reputation system is subverted by creating multiple fake identities such as user profiles), [204] document and evaluate the evolution of defences that have been developed to mitigate this growing threat. The study shows trends in defence mechanisms, that are quickly becoming mathematically complex in nature and sometimes proven to be ineffectual in practice e.g. when applied against a real-world primitive attack. The researchers suggest a defense in depth approach is necessary, such as implementing region-based community detection and social graph topologies to identify sybil attacks where sybil regions (forged identities/communities) are connected to honest nodes (legitimate user profiles). They also indicate that social network operators should implement machine learning algorithms, user profiling tools and user activity monitoring in combination to help identify attacks.

2.2.2.5 Machine Learning

Research has demonstrated how malware detection through machine learning can be dynamic, where suitable algorithms such as k-nearest neighbours, decision tree learning, support vector machines, bayesian and neural networks, can be applied to profile files against known and potential exploitations and distinguish between legitimate and illegitimate data [205, 206, 207]. Machine learning algorithms have been successfully applied to detect malicious emails, using anomaly classification techniques that demonstrate the potential and capability for further application in other related semantic attack areas [208]. Research conducted to tackle the threats of social engineering specifically has seen the employment of neural networks to predict the legitimacy/illegitimacy of phone calls. Using a fabricated dataset, the approach showed positive accuracy in prediction responses, and as such the classification methodology may be suitable across a wider range of semantic attack types. However, dependable datasets generated from real-world data may be essential, as well as the ability to formally integrate algorithm decision processes into suitable access-control systems [209]. An effort to build credible classifiers for teaching machine learning algorithms has been proposed in [190], where bespoke honeypots provide features that can indicate and help filter out social network spammers.

The application of machine learning for defence against semantic attacks can be traced back to the training of support vector machines using binary features for detecting early spam emails [210]. A significant amount of research utilising machine learning algorithms has focused on building efficient detection mechanisms for web and email phishing attacks, with work by [211] and [212] testing a range of clustering and filtering techniques to classify data for accurate prediction of attack detection. Similarly, the research conducted by [213] highlights the difference between legitimate and illegitimate URLS and proposes a regression filter for constructing classifiers specifically for URL phishing. In a related piece of work, [214] developed a system to detect phishing emails based on the component features of the email, such as body of text, sender address or embedded images, using combinations of machine learning for classification and class modelling mechanisms for filtering. [215] have continued this research by demonstrating the effectiveness and functionality of a range of filtering approaches that, if combined, can provide effective detection for this attack vector.

Advancements in the application of Ontological Semantics Technology have seen developments that demonstrate applicability to semantic attacks too, employing an artificial intelligence approach with an expression based dataset as the prediction mechanisms for building inferences about textual data [216]. This type of detection mechanism indicates useful application against phishing attacks observed in emails and websites. Whilst this technology is not strictly classified as machine learning in regards to the algorithms that are used, there is a learning process coordinating decisions on a piece of data, in this case text-based language, to make predictive inferences about malicious or non-malicious communication. More recent research by [217] has introduced CANTINA+, which is a framework for website phishing detection using machine learning algorithms over a range of systems including search engines and third party services. This research brings together multiple systems, such as search engines, third party services with machine learning mechanisms to accurately classify web pages.

In [108], machine learning and emulation techniques have been used to detect anamolous javascript behaviour against established behaviour profiles to prevent drive by downloads. The researches apply 10 different features of an attack process to categorise activities involved in drive-by downloads, emulating website behaviours that match these features in combination with an anamoly detection classifier; implemented in a publicly available tool named JSAND. The researchers presents results of JSAND's application on a large-scale dataset of Javascript code. The study is able to demonstrate through the use of an emulator, detection of malicious code when compared with anamolous features built into a learned classifier (machine learning) model. Further research aims to build on the JSAND system to provide a browser extension that provides pro-active detecton and mitigation of drive-by download attacks.

Machine learning techniques have been applied to detect phishing attacks in Twitter. In [218], a URL phishing detector for Twitter focuses on tweet content, length, hashtags, mentions, age of account, number of tweets combined with URL features (shortened text, non standard domain prefix and suffix structure). Using these variables within machine learning classification techniques (Naive Bayes, Decision Tree and Random Forest algorithms), detecting phishing techniques with an accuracy of 92.52 %. The system has been developed for practical implementation, where the researchers have deployed the system as a Chrome browser extension, working in real-time to classify whether a tweet is a phishing attack or a legitimate communication from a twitter account. Furthermore, the study has been tested on a user community to evaluate its usability and practicality in application; with the researchers claiming its relative simplicity and ease of use. The work represents research with a product outcome that is usable to the internet community against a known vulnerability of a popular social media platform. Other URL detection mechanisms have been demonstrated by [219], the study introduces a system that provides real-time URL spam filtering. The system proposed, Monarch, performs real-time filtering of spam, phishing and malware URLs that are submitted to web services through the use of linear classification with the combination of iterative parameter mixing and subgradient L1-regularisation (which is a tuning parameter used in statistical modelling to prevent over-fitting and improve generalisation in probabilistic models); calculating an impressive overall accuracy of 91% and 0.87% false positive margin.

[220] have recently developed a system for classifying and blocking spear phishing emails from compromised email accounts by training a support vector machine based system with a behavioural model based on a user's email habits. The system collects and profiles behavioural features associated to writing (e.g., character/word frequency, punctuation, stylus), composition and sending (e.g., time/date, URL characterises, email chain content) and interaction (e.g., email contacts). These features are used to create an email behaviour profile linked to the user, which is continually updated whenever a user sends a new email. Testing showed that the bigger the email history of a user the lower the rate of false positives, dropping below 0.05% for 7000 emails and over.

Machine learning algorithms provide an architecture for profiling semantic attacks in a pre-emptive fashion. By utilising characterisation variables with behavioural input datasets (usually gathered via monitoring), precise predictions and indicators can be established as to the significance of a file's or a user's behavioural effect on a system. Whilst machine learning tools have been built, extensively tested and evaluated in research, application has heavily focused on tackling phishing attacks, and most specifically the cosmetic aspects (DV1), covering only a fraction of the wider semantic attack problem space. There are not many actual implementations within deployed and popular email services in the commercial space, and consequently there is a lack of empirical evaluations in real user-facing environments.

2.3 Attack Vs. Defence Matrix

The Attack Vs. Defence matrix shown in table 2.3 provides a comparative platform for mapping current defence mechanisms to the taxonomic criteria of semantic attacks. Its benefits are two-fold. It provides researchers with an indication of where the current state of research stands (the majority view) and how to effectively categorise this research (the taxonomy classification criteria). It also highlights attack vectors that have seen underinvestment and may point towards areas where new research may yield useful results, potentially achieving wider coverage by combining approaches from multiple areas into a single, autonomous solution. For clarity we match each of the survey literature to classification criteria where it has had the most impact or for which is directly or most relevant, and not criteria with an association by second or third order of consequence.

Table 2.3: Attack Vs. Defence Matrix

Attack Vs. Defence		Organisational		Technical				
		Policy/Process	Awareness	Sandbox	AAA	Monitoring	Integrity	Machine Learning
Orchestration	TD1	Mitnick2001[132] Colwill2009[135] CPNI2013[140] Hinson2008[133]	Bakhshi2009[165]		Ruskov2014[192]			Raskin2010[216] Stringhini2015[220]
	TD2	CPNI2013[140] Sheng2010[154] Hinson2008[133]						Drucker1999[210]
	MD1-L	CPNI2013[140] GOVUK2015[139]		Sandboxie2014[179] BufferZone2014[180] Peterson2002[171] Blasing2010[61] Greamo2011[62] Xiao2013[170]	Salem2011[191] Ruskov2014[192]			Singhal2012[206]
	MD1-R	CPNI2013[140] GOVUK2015[139] Frauenstein2013[144] Janson2011[142]	Labuschagne2011[156] Sheng2010[154] Arachilage2012[146]	Lu2010[178] Barth2008[169] Blasing2010[61]	Boshmaf2012[188]	Lee2012[196] Stringhini2013[193] Li2013[195] Lu2011[194] Ruskov2014[192]	Webroot2013[28] FCS2009[27] Bhardwaj2014[200] Dietrich2013[202]	Stringhini2015[220] Basnet2008[212] Aggarwal2012[218] Cova2010[108] Xiang2011[217] Bergholz2010[215] Garera2007[213] Fette2007[211] Drucker1999[210] Dong-Her2004[208] Lee2010[190]
	MD2		Boileau2006[86] Duflot2011[87]					
	MD3	GOVUK2015[139]			EvilAP2015[203]		EvilAP2015[203]	
	MA1	Mitnick2001[132] Colwill2009[135]		Salem2011[191]				
	MA2			Lu2010[178]	Schaff2013[163] Boshmaf2012[188]	Lee2012[196] Li2013[195] Lu2011[194] Stringhini2013[193]	Webroot2013[28]	Aggarwal2012[218]

Continued on next page

Table 2.3 – continued from previous page

Attack Vs.		Organisational		Technical				
Defence		Policy/Process	Awareness	Sandbox	AAA	Monitoring	Integrity	Machine Learning
Exploitation	DV1	Calder2010[136] Sheng2010[154] CPNI2013[140]	Bakhshi2009[165] Gulenko2013[157] Labuschagne2011[156] Kirlappos2012[149] Arachilage2012[146] Kumaraguru2007[25] Kumaraguru2009[26] Lin2011[152] Wu2006[150] Lee2015[153] Sheng2007[147]	Bianchi2015[182]	Lu2011[194] EvilAP2015[203]		Seirfert2013[201] Webroot2013[28] FCS2009[27] Hara2009[199] Dietrich2013[202] EvilAP2015[203]	Xiang2011[217] Bergholz2010[215] Bergholz2008[214] Garera2007[213] Basnet2008[212] Fette2007[211] Lee2010[189] Dong-Her2004[208]
	DV2	CPNI2013[140]	Gulenko2013[157] Schechter2007[151] Anderson2013[161] Neupane2014[162]	Bianchi2015[182]	Salem2011[191]		Alvisi2013[204]	Singhal2012[206] Thomas2011[219] Cova2010[108] Raskin2010[216] Drucker1999[210] Lee2010[190]
	DV3		Gulenko2013[157] Lin2011[152] Lee2015[153] Anderson2013[161] Neupane2014[162] Sheng2007[147] Wu2006[150]	Bianchi2015[182]	Lee2010[189]		FCS2009[27] Bhardwaj2014[200] Dietrich2013[202]	Aggarwal2012[218] Stringhini2015[220]
	IM1	Calder2010[136]	Bakhshi2009[165] Arachilage2012[146] Kumaraguru2007[25] Kumaraguru2009[26] Konak2012[160]		Salem2011[191] Lee2010[189]		FCS2009[27]	Dong-Her2004[208] Basnet2008[212] Aggarwal2012[218] Xiang2011[217] Bergholz2010[215] Garera2007[213] Fette2007[211] Drucker1999[210] Lee2010[190]

Continued on next page

Table 2.3 – continued from previous page

Attack Vs. Defence		Organisational		Technical				
		Policy/Process	Awareness	Sandbox	AAA	Monitoring	Integrity	Machine Learning
	IM2		Konak2012[160]	Sandboxie2014[179] QubesOS2015[181] Microsoft2007[177] BufferZone2014[180] Bianchi2015[182] Barth2008[169]		Lee2012[196] Li2013[195] Lu2011[194] Stringhini2013[193]	Abdullah2011[197] Dhanalakshmi2010[198] FCS2009[27] Bhardwaj2014[200] Dietrich2013[202] EvilAP2015[203]	Chandrasekaran2006[205] Singhal2012[206] Thomas2011[219] Basnet2008[212] Aggarwal2012[218] Cova2010[108] Xiang2011[217] Bergholz2010[215] Garera2007[213] Fette2007[211]
Execution	AP1			Sandboxie2014[179] BufferZone2014[180] QubesOS2015[181] Blasing2010[61] Xiao2013[170] Barth2008[169] Lu2010[178]	Neuman1994[185] Desmond2008[186] Motiee2010[184] Turner2010[187]			
	AP2			Sandboxie2014[179] BufferZone2014[180] QubesOS2015[181] Blasing2010[61] Xiao2013[170] Barth2008[169] Lu2010[178] Bianchi2015[182]				
	ES1			Sandboxie2014[179] BufferZone2014[180] Blasing2010[61] Xiao2013[170] Barth2008[169] Lu2010[178]	Neuman1994[185] Desmond2008[186] Motiee2010[184] Turner2010[187]			Stringhini2015[220]
	ES2	Frauenstein2013[144]			Neuman1994[185] Desmond2008[186] Turner2010[187]			

We can observe that target description is the classification category that is one of the least covered across the surveyed defence mechanisms. The usual approaches relate to policy and process, which are typical in the corporate arena, but somewhat surprisingly, we have identified only three related technical approaches, none of which in monitoring or integrity. This highlights an area where further research could be beneficial. For example, semantic attacks that are automated (MA2), remotely distributed (MD1-R) and promiscuously targeted (TD2) would generate the same pattern and possibly noticeable volumes of network traffic, which may be detectable by a network monitoring system. Alternatively, an attack that is manual (MA1) and targeted (TD1) may be more easily identified by an integrity mechanism, for example by authenticating an addressable feature.

Another observation that can be made is that there are several areas that are not adequately addressed by policy and process controls. Examples include the hardware distribution techniques that feature no software interaction (MD2), as well as attacks that persist and may target the user periodically (AP2). From the matrix, it is also evident that awareness and machine learning have the widest and largely overlapping coverage. Yet, we have not identified a single example of research where the two areas have been combined, potentially by incorporating a machine learning component in an awareness system, or by feeding awareness data for the training of a machine learning system.

Interestingly, awareness efforts have primarily focused on exploitation factors, and particularly in understanding the impact of education against the plethora of deception vectors that are used in attacks. However, most have only addressed this through scrutiny of user interfaces (IM1) and not of programmatic interface manipulation (IM2), perhaps because awareness methodologies are typically designed for non-technical audiences. Also interesting is the application of sandbox technologies. They are highly appropriate for the software-based locally distributed (MD1-L), one-off (ES1) attacks, which are generally difficult to defend against, but have seen a relative underinvestment in application against deception techniques (DV1-3). However, as recent research has shown [182], sandboxing can be effective in identifying semantic attacks that implement both behavioural and cosmetic deception. Therefore, sandboxing mechanisms are likely to offer a viable focus of future research against deception techniques.

A distinctive observation of the matrix relates to defences that incorporate user awareness, sandbox and machine learning techniques. In combination, these three defence categories apply to almost all of the classification criteria. Note that we previously demonstrated how research in each classification category will be to a large extent applicable across several attacks that share a characteristic. Therefore, it is conceivable that a defence system combining these features would potentially provide protection against a very wide range of semantic attacks. To the best of our knowledge, a defence mechanism designed to mitigate a multitude of semantic attacks integrating all of these features does not exist yet.

2.4 Discussion: Grand Challenges in Semantic Attack Research

Semantic attack research is fundamentally multidisciplinary. Different aspects of it may require an understanding of personality traits, human-computer interaction, computer and network monitoring, malware dynamics, and access control. In this context, we have identified the following grand challenges.

2.4.1 An authoritative semantic attack dataset

The lack of publicly available, up to date, relevant and reliable datasets is a pervasive issue in computer security research. In semantic attacks, the human factor makes the acquisition of datasets even more difficult. Researchers often conduct lab-based experiments, which allow for a controlled environment where several socio-technical metrics can be evaluated, but their realism is arguably limited. The alternative, which is to run "social experiments" at the scale required for collecting usable data is problematic. Even if the semantic attacks conducted are innocuous, their potential impact on the users is poorly understood and difficult to monitor. Crucially, debriefing is ethically vital for any deception-based research, but conducting effective debriefing in large-scale social experiments is not straightforward.

The problem is aggravated by the wide range and dynamic nature of deception mechanisms employed. By the time a research group studies a new attack, designs a large-scale experiment, recruits participants, conducts it, analyses and publishes the results, the particular attack may already be gradually abandoned in the wild in favour of newer ones. We propose that this can be somewhat alleviated by focusing not directly to specific attack types, but to the taxonomic characteristics shared between multiple attacks. An authoritative dataset that would be useful in this manner does not exist.

2.4.2 Predicting susceptibility to semantic attacks in real-time

A challenge related to the acquisition of an authoritative dataset is the identification of measurable semantic attack susceptibility indicators that are practically usable by technical defence developers. For instance, an interesting real-world study by [221] has identified conscientiousness as a personality trait that is highly correlated to a user's susceptibility to spear-phishing attacks. This might be helpful to social scientists and designers of awareness programs, but is of little use to a technical protection system's developer, because it is unclear how conscientiousness can be measured by a software system in real-time or automatically. For example, behavioural science studies designed to validate the impact of conscientiousness on susceptibility to semantic attacks have been shown to be contradictory [222, 221], measured using survey and questionnaire instruments which are validated on different base-lines and analysed off-line against qualitative measures. What may be measurable, such as typing speed, time to click, typing errors, mouse movement, number of browser tabs open, time logged in, may not necessarily be relevant. A significant challenge is to identify automatically measurable features that do indeed correlate with susceptibility to semantic attacks.

2.4.3 Patching the user

An important strength of technical security measures is the ability to stay up to date. Following an antivirus update, which is automatically and rapidly delivered, a computer is protected effectively against several new and evolving threats. There is no equivalent concept for semantic attacks. A software application or embedded system may have a serious security flaw, but after this is discovered, the vendor can issue a patch that addresses it. Again, there is no equivalent concept for semantic attacks. The user, which is the entity of the system that is targeted, may take months or years to learn how to recognise and protect from new deception techniques.

The challenge here is to develop a mechanism by which a user's vulnerability to a new semantic attack vector can be reduced in a timely manner. This may involve elements of both dynamic access control and training. For example, a corporate user's access rights that are not core to their job role may be modified dynamically according to the indicators identified in Section 2.4.2. An access restriction may then be lifted once the user successfully completes a brief e-learning module or a practical assessment on the latest semantic attack methodologies. In general, complying with the core security principle of user least-privilege should ensure that any user in an organisation does not have access to system rights which are outside of their core role. However, this also requires a mature role-based or attribute-based access control platform and policy to enforce securely; which is often a prohibitive management overhead as experts are needed to install and support such systems. Therefore, in smaller companies a wider range of users may (by default) have elevated system rights beyond the requirements of their core role, whereby dynamic rights assignment based on susceptibility indicators and security training (shown to address these indicators) may provide a more flexible and cost effective means to balance access control security. Of course, this highlights a further challenge, which is how to measure the effectiveness of user training or indeed of any measure designed to protect users and organisations from semantic attacks.

2.5 Conclusion

Technical countermeasures have traditionally struggled to address a good range of semantic attack types, especially the new and more dynamic ones. There is little doubt that policy control, user education through training and awareness campaigns, and other user management countermeasures can be effective, but to what extent is difficult to tell without a formal framework. In-line with our initial research aim in section 1.4, it is important to have an overall view and understanding of the semantic attack threat landscape, where the taxonomy provided here is a step in tackling this challenge. It introduces a structured baseline for classifying semantic attacks by breaking them down into their components and thus allowing the researcher to identify countermeasures that are applicable to a range of different attacks that share a subset of their characteristics. Therefore, referring back to our initial research aim in section 1.4, the taxonomy proposed identifies a set of key characteristics observed and shared across different semantic attacks, across disparate platforms. Furthermore, by complementing this taxonomy with a survey of defence measures, and highlighting their suitability against the taxonomy's categories, we have identified where existing defence techniques apply to classification criteria that are not attack-specific.

Crucially, and as a result, the "Attack Vs. Defence" matrix in 2.3 uncovers "User Awareness", "Sandbox" and "Machine Learning" as 3 defence techniques shown to address the full spectrum of the taxonomy classification criteria. In doing so, we have identified a path towards a potential defence framework that would integrate each of these techniques in order to develop a defence system that can operate holistically across the semantic attack threat space. Specifically, we propose that key user attributes related to User Awareness can be measured and supplied to a Machine Learning mechanism in order to predict levels of awareness by using these attributes as predictor features. Here, the user and their system form the basis of a human semantic attack sandbox, where user feedback through exposure to semantic attack deception vectors and the collection of awareness attributes can be utilised to directly influence the actions of a protection system. In such an environment, accurately and reliably predicting (e.g., Machine Learning) users' ability to spot semantic attacks (e.g., User Awareness), facilitates the development of a technical system that would utilise detection telemetry (e.g., semantic attack sandbox) to provide heuristic "user-driven" protection against semantic attacks.

In the next chapter, and towards this goal, a large-scale experiment is presented aiming to measure users' ability to detect semantic attacks; designing attacks based on deception vectors instead of attack specific concepts, as defined in the taxonomy classification. Using the data collected we develop a statistical machine-learning model with the aim to predict user susceptibility based on participant attributes related to user awareness concepts. We expand on the existing literature by reviewing current work in this space, leading on to the modelling process where we evaluate user attributes that are practically useful in a technical system, can be collected automatically and are reliable in their prediction of user susceptibility to semantic attacks.

Chapter 3

Predicting Susceptibility to Semantic Social Engineering Attacks

Efforts towards technical defence against semantic attacks have led to the development of solutions that are typically specific in design. This can be attributed to the sheer complexity required to translate what is essentially human deception into code, as well as attempting to combine this into a solution that spans disparate platforms. One example is phishing emails, where filtering and classification software have proven to be highly successful [223, 217, 28]. However, these defence mechanisms are built to function on email systems only, unable to prevent conceptually very similar phishing attacks in instant messaging, social media and other platforms. Similarly, automated tools developed to block drive-by downloads via web browsers have been shown to be highly effective in mitigating the threat [107, 178], yet the same tools cannot prevent a drive-by attack in removable media.

Alternative approaches to technical solutions have focused on managing users themselves, rather than the computer interface. For example, creation of policy and process for user compliance [224] has helped to define specific rules which enforce secure system use, but these are almost never applicable to the private user of a computer system and the Internet. Furthermore, compliance guidelines are usually static in nature and therefore can quickly become out-of-date when new attack methods appear. User education and awareness training have been evaluated extensively and in practice have been shown to improve user responses to specific attack scenarios [225, 146], but it is difficult to automate this process and even more difficult to measure its lasting effect. Moreover, training material tends to be limited to known exploitations and requires regular updates to include new attack vectors. Systems generating visual warnings or security indicators have also been implemented, presented to users in real-time by indicating a possible attack or whether a potential threat exists, but research has shown that in practice users often do not pay attention to them or do not understand them [226].

A comprehensive survey by Khonji et al. [227] evaluating the state of phishing detection provides a valuable insight into potential future defences. The researchers have highlighted the application of machine learning techniques as a promising approach to defence, producing accurate attack classifiers and effective defences against zero-day threats. Measuring the effectiveness of user training has also been suggested, where research towards a hybrid user/software solution is indicated as a potential multi-layered approach to protection.

Given the limitations of defences designed for specific attacks and platforms, it is attractive to look also towards the feasibility of predicting a user's susceptibility to different semantic attacks in order to augment technical systems with user-driven defence. For example, user susceptibility profiles can be used to support systems that are dynamic, by training predictors with user data collected in real-time or over a period of time, and allowing dynamic allocation of access rights dependent on a detected user profile. Furthermore, they could support the development of context-based user awareness systems, where training material would be tailored to users depending on their susceptibility to different deception vectors. User susceptibility profiles can also provide useful measurement criteria for predicting the performance of human sensors of semantic attacks, indicating whether a user report of a suspected attack is accurate (and worth investigating); sharing analogies to the learning and prediction capabilities employed in sandbox antivirus defences for categorising and identifying different malware families [228]. Towards this vision, we have conducted two experiments with the participants being asked to tell whether particular exhibits show an attack or not. We have collected data regarding both the users and their performance in detecting attacks that employ different deception vectors [9] and have developed two prediction models. The first experiment helped identify high level predictors that can be measured ethically, automatically and in real-time, whilst being applicable across the wider Internet population; we define this study as stage 1. The second experiment helped build upon the initial predictor features by further dissecting each into a series of sub-features used to predict susceptibility against new attacks using a smaller population; we define this as stage 2.

3.1 Predicting Susceptibility

3.1.1 Related Work

In computer security, it is usually computer systems, networks, applications and data that are monitored to be able to detect and mitigate threats. Researchers have also attempted to monitor and profile unauthorised users [229, 230] or witting insiders performing unauthorised actions [231]. However, semantic social engineering attacks target authorised users and lure them into performing an authorised (albeit compromising) action. Recent research in this area has focused on demographic attributes and psychological indicators as methods for predicting user susceptibility. For example, in the field of behavioural science, research has explored the impact of personality traits [232], influencing and persuasion techniques [233] as measurement criteria for predicting susceptibility to semantic attacks. A study carried out in [222] has reported that female participants exhibiting neurotic behaviour (e.g., tendency for anxiety, obsessive compulsive disorder) were more likely to respond to phishing emails than female and male participants that did not. More recently, the same researchers have conducted a spear-phishing field-experiment, where the tendency for conscientiousness (e.g., tendency to be efficient, organised and take obligations to requests seriously) reported a high correlation to phishing susceptibility [221].

Research in [234] has reported openness, positive behaviour (e.g., use of language) and high levels of conversationalist activity as predictors of vulnerability to an online social network bot. In [235], researchers have conducted a survey and field experiment of phishing attacks which found that participants who demonstrated higher degrees of normative, affective and continuance commitment, obedience to authority and trust, to be more susceptible to phishing. Similar results were also reported in a recent study in [236], where submissiveness and trust predicted higher susceptibility to phishing emails. Crucially, these personality traits were also found to perform consistently as predictors of susceptibility amongst participants from different geographical locations, in this case Australia and Saudi Arabia.

Demographic research has considered Internet usage and behaviour as prediction criteria of susceptibility to semantic attacks. In particular, it has been reported that users who have knowledge of or take guidance from visual cues (security indicators, source, design, language, etc.) on technology platforms are often good predictors of susceptibility. For example, [237], [238, 239] and [240] have all reported a lower degree of susceptibility to phishing attacks in emails and websites when the participants are aware of security indicators and visual components. However, in many cases participants did not understand what the security indicators meant and the varying severity of their message. In fact, in [241] and [242], it has been reported that the effect of habitation to the visual cues and especially security warnings increases susceptibility to attacks. Where studies have included general demographic elements such as age and gender, a number of studies have reported that female participants were found to be more susceptible to phishing attacks than male participants [154, 243, 124, 221]. In [225], users were measured demographically as to whether they have had training on the phishing email training system *PhishGuru*, where the number of training sessions taken by users are used as input features to identify the lasting effect of the training.

Technical prediction systems have been previously proposed in [244] and more recently [245]. The first describes a system which would present users through a series of information security related questions within a web pop-up. Then, the system uses a series of weighted decision algorithms to quantify the user's degree of susceptibility based on the responses to the questions, and accordingly displays a visual indicator of susceptibility to the user as a form of awareness mechanism. No security enforcing functions are implemented. To date, there is no further information regarding its practical implementation and evaluation. The latter, and more recent study empowers user to report whether an email is a suspected phishing attack. Based on prior knowledge and in-line warnings, correct reporting conversely highlights predictor features for phishing susceptibility. More recently, a commercial phishing company *PhishMe* reported results for their phishing simulator application, collating the results for 3.5 million simulated employee phishing emails, over the course of one year across 23 industries around the world. A key observation showed that repeat phishing simulations lowered susceptibility for consecutive attacks by 35% for the first successful phishing simulation to 1% by the fourth phishing simulation.

Table 3.1 provides an overview of the literature associated to susceptibility research in semantic social engineering attacks. In the "Technical measurement" set of columns, we have identified for each study whether the predictors of susceptibility can be realistically measured by a technical system in real-time, automatically and ethically. By ethical, we refer to aspects of diversity and inclusion related to protected personal characteristics [246], and we extend this to also include personality traits, where decision making based on assessment of personality types are argued to be a form of discrimination [247].

The available literature for predicting user susceptibility to semantic social engineering attacks is not as mature as other areas of computer security. Most related studies have been constrained by small sample sizes and predictors that are difficult to generalise across a multitude of semantic attacks. To some extent, this is due to the fact that most researchers focus only on phishing attacks, which is only one section of the problem space [9]. Specialised training systems have been shown to work well [225], as well as technical models combining demographic and behavioural attributes [242], but they are application-specific and do not consider other deception vectors that might be employed in semantic attacks. Therefore, it is difficult with the results produced from current studies to generalise across a wide range of attack

Table 3.1: Related research in the field of semantic social engineering attacks

Pub.	Attack Type*	Methodology	Participants	Practicality of predictors: Ethical	Real-time	Automatic	Key findings for reducing [-] or increasing [+] susceptibility
[237]	E	Exhibit survey	179	✓		✓	[-] Knowledge of technical and visual cues
[238]	E	Interview, role-play	20	✓		✓	[-] Familiarity with specific attacks, [-] visual cues
[240]	E/W	Exhibit test	17	✓			[-] User guided by technical and visual cues
[239]	W	Role-play	232	✓		✓	[-] Knowledge of attacks, technical and visual cues
[147]	W	Online Game	42	✓		✓	[-] Trained on an online phishing website training system
[235]	E	Survey, simulated phishing	588				[+] Normative, affective, continuance, obedience, commitment, trust
[241]	E/W	Survey, role-play	70	✓			[-] Interactive browser warnings, [+] platform habitation
[225]	E	Simulated phishing	515	✓		✓	[-] Trained on an embedded phishing email training system
[154]	E	Role-play	1,001				[+] Ages 18-24, [+] Gender (Female, attr. less technical training)
[243]	E	Analysis, survey, interview	224				[+] Gender (Female), [-] non-exposure to deceptive visual cues
[248]	E	Survey, simulated phishing	446	✓		✓	[-] Comp. self-efficacy, web experience, security knowledge, suspicion
[249]	E	Survey	64			✓	[-] Tailored susceptibility message
[250]	E	Simulated phishing	94	✓		✓	[-] Trained on phishing email training system, attack experience
[242]	E	Survey, simulated phishing	321	✓			[-] Knowledge+attention+cues+elaboration [+] load triggered habitation
[251]	B	Social media experiment	500	✓**	✓**	✓	[+] High platform activity, openness, positive behaviour (e.g., language)
[234]	E	Simulated phishing	10,917				[-] Gender (Female), [-] Age (18-21),
[252]	E	Survey, Training experiment	321	✓		✓	[-] Attention to deception indicators, attack knowledge
[253]	E	Test assessment	210	✓		✓	[-] Computer security self-efficacy and proven attack knowledge
[222]	E	Survey, simulated phishing	100				[+] Neurotic personality trait
[254]	E, W	Simulated phishing	105				[-] Investment of cognitive effort
[255]	E	Simulated phishing	2,624				[+] Cialdini's influence techniques + self/non-self determination
[236]	E	Survey	296				[+] Submissiveness, trust, [-] Visual cues & important privacy data
[256]	W	Simulated phishing	251	✓		✓	[-] Trained on android-based phishing game [-] Self-assessment
[221]	E	Simulated phishing attack	40				[+] Gender (Female), conscientiousness, self-efficacy
[257]	W	Simulated phishing	21		✓	✓	[-] Gaze time on browser chrome elements
[258]	E	Simulated phishing	54				[-] Distrust, calmness, attention, attack exp. [+] Extroversion, Anxiety
[259]	W	Simulated phishing	173	✓		✓	[-] Knowledge combined with familiarity and HTTPS indicators
[260]	E	Simulated phishing	600	✓			[+] Attention allocation, dispositional optimism, organisation familiarity
[261]	E	Simulated phishing	3.5 million***	✓	✓	✓	[-] Repeated attack experience via phishing email simulator
[262]	E	Simulated phishing	16	✓	✓	✓	[-] Embedded URL phishing detector and advisor
Exp.1	**E/W/W1/S/A/Q**	**Exhibit test**	**4,333**	✓	✓**	✓	**[-] Sec. training, platform freq., familiarity, self-efficacy**
Exp.2	**E/W/A/T/S1**	**Exhibit test**	**315**	✓	✓**	✓	**[-] Sec. training, platform freq., dur., familiarity, self-efficacy**

*** E = Phishing emails/IMs, W = phishing websites, B = Online social network bot, S = search engine poisoning, W1 = WiFi evil twin, A = Fake app**
Q= QRishing, T= Typosquating, S1 = Scareware
**** Not all features can be accurately measured in real-time**
***** Aggregated statistical results from customers of PhishMe Simulator**

types and it is unclear which of the research results could be realistically integrated into a technical system for defence. To overcome these limitations and as our aim is focused on facilitating the development of technical defence systems, we only select predictor terms that can be collected and measured in real-time, automatically and ethically. We argue that in order to predict susceptibility to a wide range of semantic attacks, the mechanism for measuring susceptibility should be naive to low-level and attack-specific parameters (e.g., sender source and body of text within an email, URL composition in website post, etc.).

3.2 Indicators of susceptibility

To identify practical indicators, we start with five high-level concepts associated to user knowledge, experience and behaviour:

3.2.1 Auditable Indicators

Auditable susceptibility indicators are user profile features that can be collected in real-time, ethically and in a fully-automated manner and without user bias, as they focus on the measurement of factual user activity rather than speculation.

Security Training (S)

Refers to the individual's type of computer security training. Prediction of susceptibility by computer security training has been shown to produce accurate results [225, 261], but the approach can be limited by the specialised system delivering the training or the specific training curriculum; especially where these system tend to focus on one type of attack such as phishing emails. For example, it is likely that a user who is self-trained will cover a wider range of material relevant to their technology profile than an employee who has only received work-based training on systems the organisation uses. Moreover, the long-term benefit of training its effects on detection and associated skill fade is not clear.

Frequency (FR) of access

Refers to the frequency with which the individual accesses the type of platform. A user who accesses a specific type of platform (e.g., social media websites) very frequently may be more aware of the kind of attacks that occur within that type of platform, based on their experience with conventional platform behaviour and appearance [248]. Furthermore, frequently accessing specific platform providers (e.g., with Google+ social network site) may extend to other similar platforms (e.g., Facebook, Twitter), where behavioural and cosmetic attributes on the platform are shared or follow user experience and interface design conventions.

Duration (DR) of access

Refers to the duration for which the individual accesses the type of platform. Similarly to FR, a user who uses a specific type of platform for long periods may be more aware of the kind of attacks that occur within and on what components for that type of platform [238, 257]. However, it is also possible that the longer the duration the higher risk of platform habitation which may or may not have an adverse effect [251].

3.2.2 Self-Efficacy Indicators

Self-efficacy attributes, influences and measurements have been studied extensively across a multitude of disciplines, particularly within the fields of sociology and psychology. In computer science, studies related to computer security tend to focus on measuring the impact in reducing vulnerability when users engage with specific security training systems or participate in simulated semantic attacks experiments, and primarily from a behavioural sciences standpoint. By comparison, a relatively small amount of attention has been given to self-efficacy within computer security research aiming to actively develop practical technical defences. For instance, a study by Rhee et al [263] in 2009 agreed that self-efficacy in information security did have substantial explanatory power regarding individuals information security practice behaviour, but their results are mostly theoretical in nature and remain unclear as to how to apply such findings in a technical system. A more recent study in 2014 by Arachchilage and S. Love [264], found that conceptual and procedural knowledge positively effected self-efficacy and as a result enhanced computer users phishing threat avoidance motivation and behaviour. However, again, the results are represented by a largely theoretical model construct that is not evaluated within an empirical phishing experiment, and without clear indication of how to measure the self-efficacy criteria in a real system.

Measuring self-efficacy has lead to a number of proposed specific self-efficacy scales that have been evaluated as valid and accurate instruments for use across a range of self-efficacy research concepts [265, 266], however these scales often do not fit well with specific problems in computer science, as per the two prior examples that have focused on measuring self-efficacy against information security threats. Therefore, most existing studies in the field of computer security develop arbitrary scales that suit the requirements and confines of the specific research problem at the time. In our work we select basic self-efficacy criteria that is both simple to use, but also practical for use within a technical computer system. At the time of writing and to the best of our knowledge a unified and agreed self-efficacy measurement scale for computer literacy and computer security awareness does not exist.

From a technical standpoint, self-efficacy features can be collected in real-time if an existing record can be referenced prior to a security incident. Depending on the last time a user updated their self-efficacy records (e.g., for security awareness or platform familiarity), external auditable features can also potentially provide a means to explain the users historic self-efficacy values and then employed to generate a predicted temporary efficacy report value until the users updates it, or if the existing self-efficacy measurement is inappropriate. Based on this approach, self-efficacy can also be automatic, but in most cases for active and representative user input will require manually updating. Self-efficacy features are also ethically compliant from a user privacy perspective, however if a user were to abuse their ability to perform self-assessment, technical means such as test-based validation can be used to adjust features values supplied by users.

Computer Literacy (CL) and Security Awareness (SA)

Refers to the user's self-efficacy in respect to their computer literacy and computer security awareness. User self-efficacy in computer literacy and computer security awareness has been reported as an important predictor in numerous studies seeking to identify what influences reduced susceptibility to phishing emails [248, 253, 221, 256]. Overall, self-efficacy was found to accurately represent a user's expectation of their ability to use a computer system competently and securely. However, self-efficacy implicitly harbours a degree of bias depending on the user's honesty and the accuracy and practicality of the measurement scale used. While we count it here as practical, we assume that in actual application, both CL and SA would need to be validated against evidence (e.g., with some form of testing, certification, etc.).

Familiarity (FA)

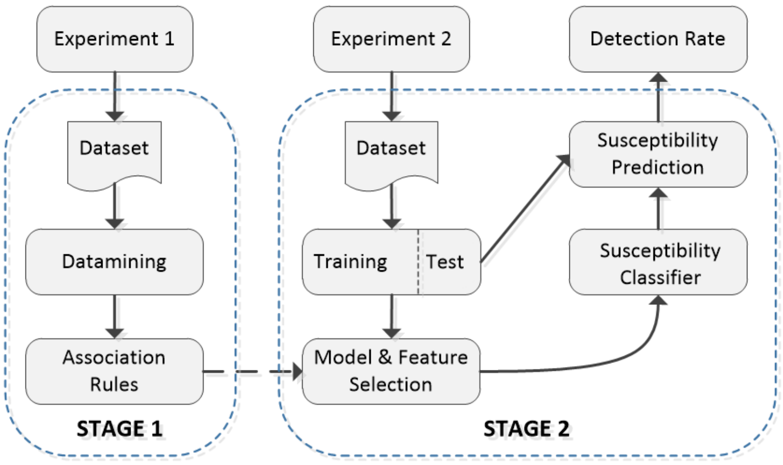
Refers to the familiarity the individual has with a given platform. Familiarity is a key enabler of distinguishing between what visually looks normal and what is normal behaviour. For example, in [238, 239], the researchers have identified familiarity with specific attacks and visual cues as key predictors of susceptibility, both of which describe how a user identifies what is normal visually or behaviourally on a system and what is not. Similar findings were also reported in [240] and more recently in [259], with knowledge of visual cues being attributed to familiarity with the type of platform used. In this context, platform habitation [241] is a factor that can increase susceptibility to semantic attacks, facilitated in part by platform familiarity. At the same time, without familiarity a user may be unaware how a system should normally look and behave, and consequently may fail to detect an attack or may see threats where they do not exist.

In section 3.6.2 we evaluate the viability of utilising self-efficacy indicators for computer literacy, security awareness and platform familiarity measurement by conducting correlation and modelling analysis; which also serves to identify whether such features can be accurately validated by other unbiased user attributes (e.g., Auditable indicators).

3.3 Methodology

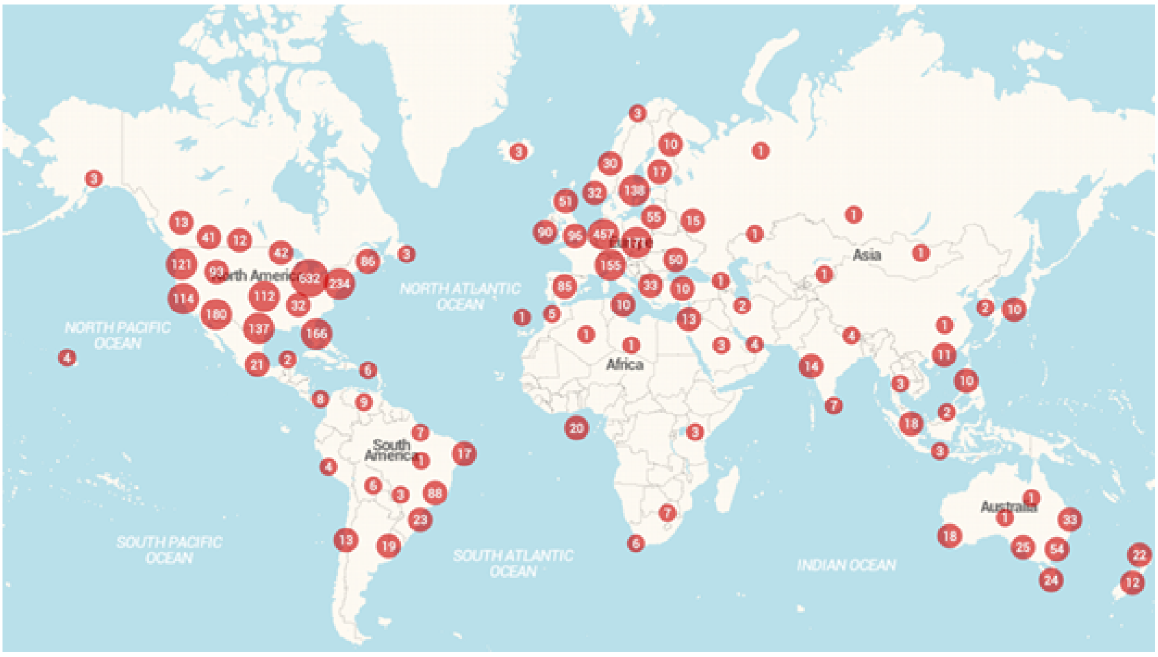
Figure 3.1 summarises our two-stage experimental approach. In stage 1, we have conducted a large scale experiment, where we applied data mining techniques to try and identify whether relationships and associations exist between the different indicators of susceptibility described in Section 3.2. In stage 2, we have utilised the results from the stage 1 analysis to apply a greater degree of granularity (and measurability) to each of the indicators highlighted through the data mining process. These refined predictor features were then employed in a second experiment in order to determine practical predictors of susceptibility and develop a model to form a susceptibility classifier.

Figure 3.1: Experiment approach and methodology



Both experiments were designed to be quantitative in nature in order to generate numerical data that could be transformed into usable statistics. Some qualitative data was captured in experiment 2, where users were asked to explain in free-text for each exhibit why they had classified it as an attack or non-attack; this data was used to eliminate sample "noise", such as participants who guessed or marked all exhibits as attacks (or non-attacks). Furthermore, attack exhibits were randomised so that

Figure 3.2: Number of survey participants by geographical location for Experiment 1

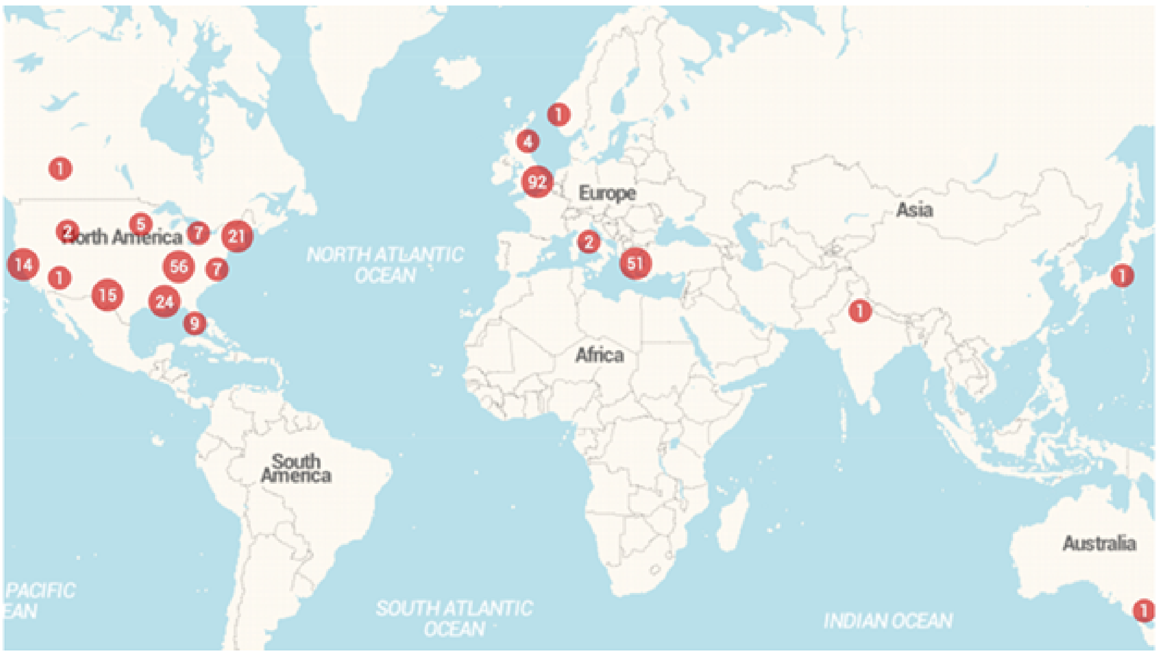


participants could not guess the order between attack and non-attack exhibits in the susceptibility test.

Both experiments were implemented in the online survey platform *Qualtrics* and consisted of a short survey that collected demographic and platform behaviour data, followed by an exhibit-based susceptibility test. In total, after sample cleaning and pre-processing, experiment 1 consisted of 4,333 participant responses, and experiment 2 consisted of 315 participant responses. Both experiments provided participants with a study brief prior to commencing the survey, so as to ensure they understood how to proceed with answering the survey and exhibit test questions.

The research was approved by our institution's research ethics committee and participants were informed of the purpose of the study prior to providing online consent and confirmation of being over 18 years of age. Furthermore, all data were anonymised and participants were also given the opportunity to opt out of the study analysis after completing the test; participants who opted out had their responses removed from the study.

Figure 3.3: Number of survey participants by geographical location for Experiment 2



3.3.1 Recruitment

3.3.1.1 Experiment 1

In the first experiment, participants were cultivated via an online advertisement challenging people to take a test of their susceptibility to semantic social engineering attacks. This advertisement was posted in a number of popular online forums and social media communities, including Reddit, StumbleUpon, Facebook and Twitter. Additionally, undergraduate and research students were recruited via email. The recruitment methodology of presenting the questionnaire primarily as a challenge and secondarily as a research medium proved successful because participants were eager to test themselves on a real-world skill that is becoming increasingly important. As a result, our advertisement gained reputation quickly by being up-voted and shared within a variety of social media platforms, resulting in a substantial sample size that allows meaningful statistical analysis (4,333 responses). Our sample included participants across a broad range of online platforms, as well as technical and non-technical environments from within our university's undergraduate population. Also, in many studies in this area, the real nature of the study is initially hidden from the participants, so that the strength of a deception attempt is not weakened by suspicion. Here, instead we use the participants as human binary classifiers of exhibits into attack versus non-attack. In this manner, we can reveal the nature of the study from the beginning, which addresses key technical and ethical challenges associated with temporarily deceiving the participants.

3.3.1.2 Experiment 2

In the second experiment, a controlled recruitment policy was employed in order to achieve a balanced sample of participants who had received some security training and were technology savvy and generic online users with little or no training. New undergraduate and research students were invited to participate in the experiment if they were studying a computer security program and the professional service *Qualtrics Panels* was used to recruit participants from a wider, more generic population demographic. Specifically, participants from the US ranging between the ages ranging from 18-65, both female and male, were defined as the participant selection criteria. No specific technology or security training attributes were defined in the *Qualtrics Panel* recruitment. Figures 3.2 and 3.3 shows the geographical distribution of the participants for both experiments.

3.3.2 Experiment Design

The survey portion of the experiment required participants to answer a series of questions related to age, gender, general education, security training (S), platform familiarity (FA), frequency (FR) and duration of access (DR), computer literacy (CL) and security awareness (SA):

Security Training (S)

Formal computer security education (S1), work-based computer security training (S2) and self-study computer security training (S3), each coded as a binary response: Yes (1), No (0). In relation to the terminology used in [267], we directly map formal education as “Formal Learning”, work-based training as “Non-formal learning”, and self-study as “Informal learning”.

Familiarity (FA)

We use FA for familiarity with a particular provider’s platform (e.g., GMAIL), coded as: Not very familiar (1), Somewhat familiar (2), Very familiar (3).

Frequency (FR)

For each platform category presented in the susceptibility test, coded as: Never (1), less than once a month (2), atleast once a month (3), weekly (4), daily (5).

Duration (DR)

For each platform category presented in the susceptibility test, coded: None (1), less than 30 mins (2), 30 mins to 1 hour (3), 1 to 2 hours (4), 2-4 hours (5), 4 hours+ (6).

Computer Literacy (CL)

Self-reported level of computer literacy using a 0 (novice) - 100 (expert) scale.

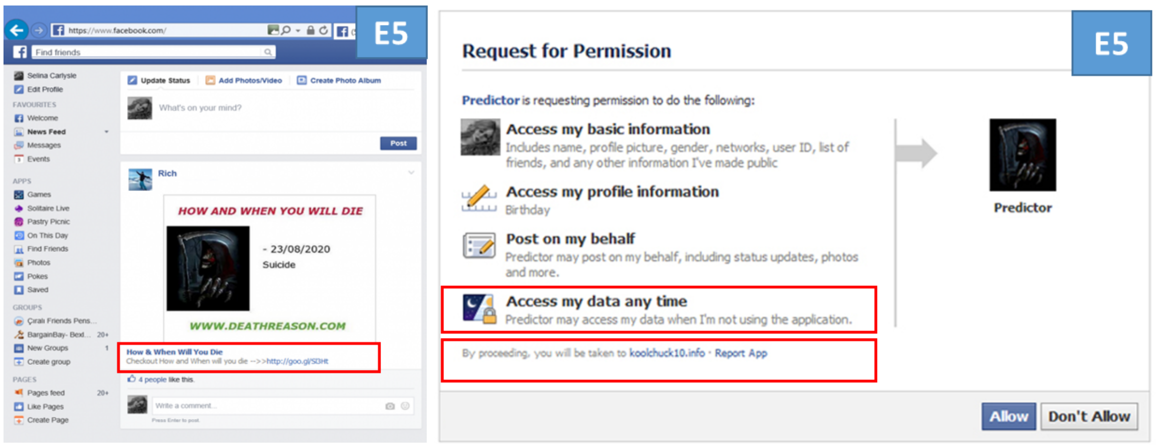
Security Awareness (SA)

Self-reported level of security awareness using a 0 (novice) - 100 (expert) scale.

Each experiment included a series of 12 exhibits (6 attacks and 6 non-attacks), each containing a concise scenario followed by an exhibit, consisting of one or more screenshots, GIF animations or videos. For each, participants were asked to examine the exhibit and provide a binary response to categorise each one as: “Most likely an attack” or “Most likely not an attack”. In our analysis, correct responses were coded as 1 and incorrect ones as 0.

To determine general indicators of susceptibility, the attack exhibits chosen spanned a range of semantic social engineering attacks across different platforms. We have developed each semantic attack according to the three different types of deception vectors of the semantic social engineering taxonomy in chapter 2. In accordance with this, and as described in chapter 2, deception vector refers to the mechanism

Figure 3.4: Experiment 1: Exhibit 5 (screenshot) - Fake "Clickbait" app on Facebook



How & When Will You Die
Checkout How and When will you die -->>http://goo.gl/SI3Ht

Request for Permission

Predictor is requesting permission to do the following:

Access my basic information
Includes name, profile picture, gender, networks, user ID, list of friends, and any other information I've made public

Access my profile information
Birthday

Post on my behalf
Predictor may post on my behalf, including status updates, photos and more.

Access my data any time
Predictor may access my data when I'm not using the application.

By proceeding, you will be taken to koolchuck10.info · Report App

by which the participant is deceived into facilitating a security breach. It can be cosmetic (DV1), where the semantic attack is visually convincing, but does not necessarily conform to expected platform behaviour; behaviour-based (DV2), where the attack behaves in a manner that is expected or accepted within platform convention, but is not visually convincing; and both cosmetic and behaviour-based (DV3), where the attack needs to be both visually and behaviourally convincing to deceive the user.

A breakdown of the 24 exhibits developed for the two experiments is presented in Table 3.2, along with the participants' average score in each exhibit. The average score can serve as an indication of the difficulty of each exhibit. To illustrate the style of the presentation of the exhibits to the participants, we have also included three indicative examples of attack exhibits (Figures 3.5, 3.4 and 3.6, which correspond to exhibits Exp1.11, Exp1.5 and Exp2.11 respectively). For presentation purposes here, we have added red outlines to represent visual attack indicators in the exhibit. These outlines were obviously not visible to the participants.

3.3.3 Overall participant performance results

To determine overall accuracy and precision, we follow the approach defined in signal detection theory [268, 269], which is geared towards analysing data generated from human experiments, where the task is to categorise participants' responses generated by a known process or by chance. This approach is common in analysing experiments

Figure 3.5: Experiment 1: Exhibit 11 (screenshot) - “Qrishing” attack leading to Steam phishing site

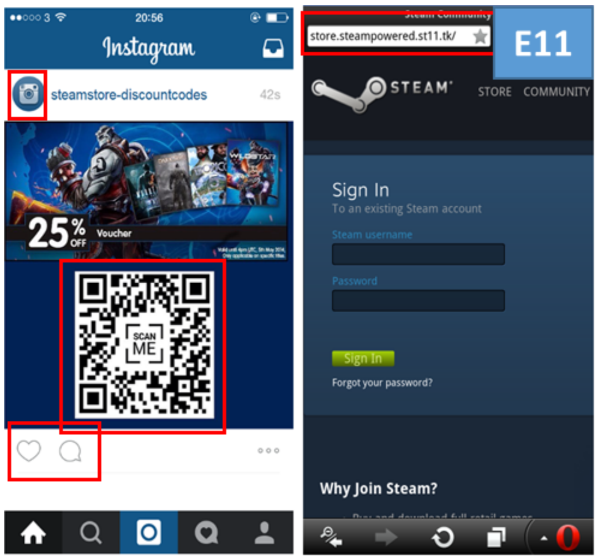
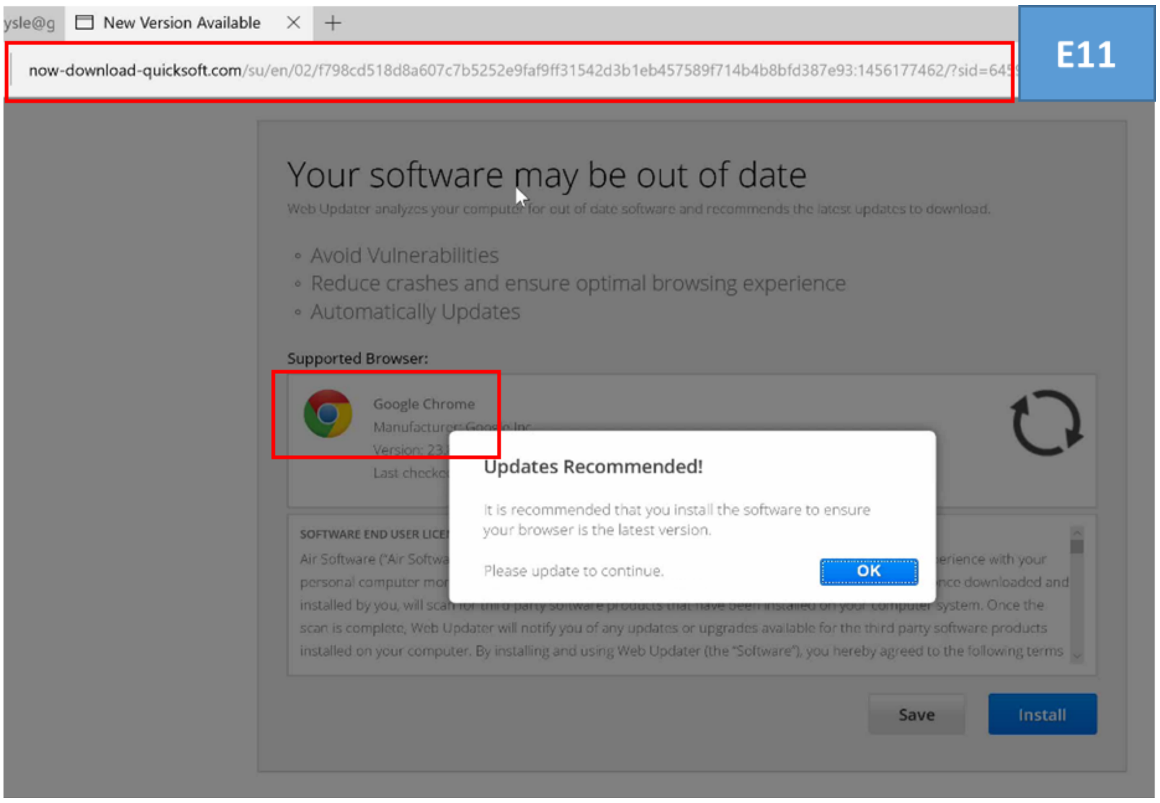


Figure 3.6: Experiment 2: Exhibit 11 (Video) - “Typosquatting” attack on Microsoft Edge browser leading to an attack website with a malicious update prompt for Google Chrome browser


New Version Available
now-download-quicksoft.com/su/en/02/f798cd518d8a607c7b5252e9faf9ff31542d3b1eb457589f714b4b8bfd387e93:1456177462/?sid=64
E11

Your software may be out of date

Web Updater analyzes your computer for out of date software and recommends the latest updates to download.

- Avoid Vulnerablities
- Reduce crashes and ensure optimal browsing experience
- Automatically Updates

Supported Browser:

Google Chrome
Manufacturer Google Inc
Version: 23
Last checked

Updates Recommended!

It is recommended that you install the software to ensure your browser is the latest version.

Please update to continue.

OK

SOFTWARE END USER LICE

Air Software ("Air Softwa ... erience with your personal computer mor ... nce downloaded and installed by you, will scan for third party software products that have been installed on your computer system. Once the scan is complete, Web Updater will notify you of any updates or upgrades available for the third party software products installed on your computer. By installing and using Web Updater (the "Software"), you hereby agreed to the following terms

Save Install

that involve semantic attacks, such as phishing [243]. In the standard formulas used below, for exhibit $k \in [1, K]$, $T_{p,k}$ is the number of true positives (i.e., correctly identified as attack), $T_{n,k}$ is the number of true negatives (i.e., correctly identified as non-attack), $F_{p,k}$ is the number of false positives (i.e., incorrectly identified as attack), and $F_{n,k}$ is number of false negatives (i.e., incorrectly identified as non-attack). Note that in this case, $K = 12$, and by accuracy and precision, we are referring to the average accuracy and average precision across all 12 exhibits.

$$Accuracy = \frac{1}{K}\sum_{k=1}^{K}\frac{T_{p,k} + T_{n,k}}{T_{p,k} + T_{n,k} + F_{p,k} + F_{n,k}} \tag{3.1}$$

$$Precision = \frac{1}{K}\sum_{k=1}^{K}\frac{T_{p,k}}{T_{p,k} + F_{p,k}} \tag{3.2}$$

To facilitate the analysis of the participants' responses, we developed an equal number of attack and non-attack exhibits for each platform category used in the experiment. In this section, our aim is to simply evaluate the overall performance of the users in our samples as human classifiers of the given exhibits. We also note the performance of individual groups that are commonly studied in this space, including groups by country, age and gender. We derive the country based on the IP of the participant, assuming that there is no strong reason to believe that several participants would have spoofed their IP while taking part in this experiment. Also, age and gender are self-reported. Again, we do not have any strong reason to believe that several participants would provide false details in this case.

Table 3.3 summarises the performance of users from different countries, which we observe to be almost identical across the world, with mean accuracy of 0.74 (with variance of 0.0002) and mean precision of 0.77 (with variance of 0.0002). For this reason, we did not consider the geographical factor in the second experiment. Also, this is advantageous when developing a prediction model to be applied across all populations. Table 3.4 summarises the performance of participants of experiments 1 and 2 based on age and gender. Here, we observe slightly more pronounced performance

Table 3.2: Brief description of the 24 exhibits developed for the two experiments. The overall score (percentage of correct answers) of the participants for each exhibit is an indication of its difficulty. SA refers to the existence or not of a semantic attack in each exhibit (Y/N). For the exhibits where there was an attack, DV is the deception vector (DV1: cosmetic; DV2: behaviour-based; DV3: both cosmetic and behaviour-based).

Experiment 1

ID	Perf.	SA	Platform Type	Provider Platform	DV	Exhibit	Description
E1	0.72	N	Social Media	GoogleApps	-	Screenshot	Facebook app down from GoogleApps with app permissions
E2	0.63	N	Email	Gmail	-	Screenshot	Failed login report from Google for this email account based on location
E3	0.67	Y	Public WiFi	Starbucks WiFi	DV3	Screenshot	Evil Twin WiFi attack, fake SSID and web portal with private IP, login request email credentials
E4	0.76	N	Social Media	Facebook	-	Screenshot	Mistyped Facebook URL “Facebok” redirects to real Facebook website
E5	0.88	Y	Social Media	Facebook	DV2	Screenshot	“Clickbait” app shared by friend on user timeline; requests permissions and redirects
E6	0.67	Y	Email	Gmail	DV1	Screenshot	Spearphishing Twitter email responding to bogus functionality, with button to login
E7	0.88	Y	Social Media	Twitter	DV3	Screenshot	Twitter phishing website spoofing Twitter website real homepage, URL “http://twitteri.com”
E8	0.78	N	Public WiFi	BT WiFi Hotspot	-	Screenshot	BT FON public WiFi hotspot SSID (multiple found in area), login portal, secured with HTTPS
E9	0.67	Y	Email	Gmail	DV1	Screenshot	Email confirming Paypal purchase with link to “dispute transcation”
E10	0.72	N	Email	Outlook	-	Screenshot	Paypal email requesting activation via URL link. Approved sender address icon supplied
E11	0.86	Y	Social Media	Steam	DV3	Screenshot	QRishing attack on fake Steam Instagram account redirecting to fake website login
E12	0.64	N	Social Media	Twitter	-	Screenshot	Embedded game advert in Twitter app on mobile; shows app rating and downloads

Experiment 2

ID	Perf.	SA	Platform Type	Provider Platform	DV	Exhibit	Description
E1	0.78	Y	Social Media	Facebook	DV1	GIF	Video masquerading acting a “clickbait” link to bogus survey designed to steal user credentials
E2	0.55	Y	IM	Steam	DV2	Screenshot	Phishing instant message containing URL link
E3	0.83	N	Social Media	Facebook	-	Screenshot	Facebook app update on mobile device with permissions request
E4	0.75	N	Public WiFi	Starbucks	-	Screenshot	Starbucks / AT&T login capture portal, URL generated by AT &T web proxy
E5	0.80	N	Web browser	Android	DV3	Screenshot	Fake virus scan pop-up downloading Android scareware requiring purchase to remove infections
E6	0.91	Y	Social Media	Youtube	-	Screenshot	Youtube blocked video due to location settings and copyright
E7	0.55	N	Email	Gmail	DV1	Video	Phishing email from Natwest Bank containing a URL link
E8	0.79	Y	Social Media	Facebook	-	Video	Application permissions request to add game to Facebook profile (Desktop version)
E9	0.86	N	Social Media	Twitter	-	Video	Twitter account confirmation on website; email from Twitter requesting confirmation via link
E10	0.46	N	Social Media	Twitter	-	Video	Comodo email with link to encryption certificate; link broken due to timeout of download
E11	0.61	Y	Web Browser	MS Edge	DV2	Video	”Faceb00k.com” typosquatting redirection to fake Chromeupdate website
E12	0.36	Y	E-commerce	Gumtree	DV3	Video	Phishing email redirecting to phishing Gumtree login; once entered redirects to real website

differences between the different groups. For example, we can see that female participants were less accurate and less precise (68%, 67%) than male participants (74%, 77%), which is in accordance with most of the related literature ([154], [243] and [124]). We also observe that accuracy and precision are fairly consistent between the ages of 18 and 44, but drop in the 45+ age groups. Overall, the performance of the samples of participants in both experiments is largely coherent and consistent. The sample sizes of the groups that performed slightly worse in both experiments were relatively low. Moreover, they represent protected personal characteristics, and are thus impractical for our purposes. As we aim to develop prediction models suitable for use in a technical system, age and gender need to be omitted as candidate predictors because they do not satisfy the ethical criterion that we have set. For example, an organisation implementing security controls that are stricter or less strict based on age or gender would be seen as discriminatory. Overall, the reported performance of the participants provides no strong indication that omitting these demographic variables (geography, gender, age) would have a major impact on the chosen predictor features' accuracy and precision.

Perf.	US	UK	Canada	Germany	Australia	Netherlands	Brazil	Other
Acc.	0.74	0.74	0.74	0.74	0.74	0.77	0.72	0.76
Prec.	0.77	0.77	0.77	0.77	0.76	0.80	0.76	0.80
Sample	1863	454	293	207	161	107	138	1234

Table 3.3: Experiment 1 participant performance by country

Experiment	Perf.	Overall	18-24	25-34	35-44	45-54	55+	Male	Female
	Acc.	0.74	0.74	0.76	0.75	0.71	0.67	0.74	0.68
1	Prec.	0.75	0.75	0.81	0.82	0.81	0.77	0.77	0.67
	Sample	4333	2936	1074	190	68	65	3879	456
	Acc.	0.65	0.72	0.69	0.66	0.57	0.59	0.71	0.59
2	Prec.	0.57	0.67	0.58	0.53	0.33	0.55	0.64	0.39
	Sample	315	95	104	65	27	29	232	85

Table 3.4: Experiment 1 and 2 participant performance by age and gender

3.4 Stage 1: Association Rule Mining on the results of experiment 1

While performing prediction based on the large dataset collected in experiment 1 would be an attractive prospect, in practice, the high-level features used (described in Section 3.2) would not be granular enough. Our attempts to produce prediction models solely based on them produced relatively low accuracy rates, just above the null rate for each exhibit. Instead, the primary objective of experiment 1 was to use it as a mechanism to determine which features should be explored further. For this purpose, we have performed association rule mining (ARM). ARM is a standard data mining methodology successfully employed in network intrusion detection [270], bioinformatics [271], recommender systems [272], social network advertising [273] and several other applications. It can help identify frequent itemsets (collections of attributes that frequently occur together) and association rules to determine whether strong relationships exist between two or more items.

As a brief introduction to ARM, an association rule is composed of an itemset, which comprises an antecedent, consisting of one or more attributes and forming the "IF" of a rule, and a consequent, which forms the "THEN". The percentage of cases of an item's existence amongst frequent itemsets is referred to as *support*, while the conditional probability of observing a particular exhibit response under the condition that the participant attributes contain a particular set of participant attributes is referred to as *confidence*. Here, we employ the apriori algorithm [274] to create association rules by comparing frequent itemsets to a specified support/confidence threshold that determines the strength of the rule.

Using the *Arules* package in *R* [275], we have conducted frequent itemset discovery and association rule generation configuring a threshold for support larger than the system default and the default threshold for confidence, which are 0.15 and 0.8 respectively. For each association rule, we evaluate its importance using five commonly used metrics: support/confidence as the primary interest measure for each rule, as well as lift, coverage and odds ratio of each rule as individual measures of independence. For each metric's formula below, X refers to the frequent itemset attribute(s) that consist of the participant indicators defined in section 3.3.2, forming the rule antecedent(s). The rules consequent Y defines a correct response to an attack exhibit, coded as RESPONSE=1 (i.e., for participants who classified particular exhibit correctly). In summary:

$$\text{Support: } supp(X \Rightarrow Y) = P(X \cup Y) \tag{3.3}$$

$$\text{Confidence: } conf(X \Rightarrow Y) = \frac{P(X \cup Y)}{P(X)} \tag{3.4}$$

$$\text{Lift: } lift(X \Rightarrow Y) = \frac{P(X \cup Y)}{P(X)P(Y)} \tag{3.5}$$

$$\text{Coverage: } cover(X \Rightarrow Y) = P(X) \tag{3.6}$$

$$\text{Odds Ratio: } \alpha(X \Rightarrow Y) = \frac{P(X)/1 - P(X)}{P(Y)/1 - P(Y)} \tag{3.7}$$

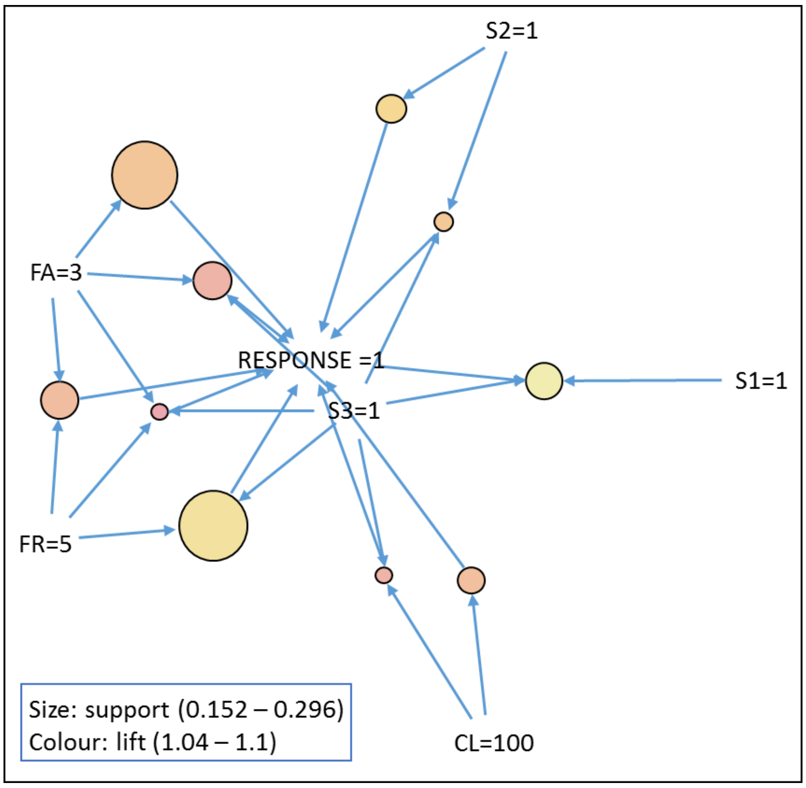
Note that on investigation, no rules were reported for RESPONSE=0 (i.e., for participants who classified particular exhibit incorrectly) to satisfy the support/confidence threshold 0.15/0.8. This indicates that within the data there exists a high degree of variability between participants who were susceptible and no distinguishable pattern between their attributes could be determined.

For lift, a value of 1.0 indicates independence of X and Y, while values greater than 1.0 indicate that participants with attributes X contain more correct attack exhibit responses Y (i.e., RESPONSE=1), than those without these attributes. An Odds Ratio of 1 indicates that Y is not associated to X, which is to say that an exhibit response is not related to the participant attributes.

Using the apriori algorithm, a total of 24 association rules were initially identified. These were then pruned by removing super rules of any other rule that has the same or higher lift. Pruning resulted in reduction from 24 to 10 association rules.

In Figure 3.7, the 10 association rules are shown where each item and vertex indicates the formation of a rule, where vertices leading to "RESPONSE=1" show the consequent of the rule. The size of each circle linking vertices is related to the

Figure 3.7: Association Rules graph with items and rules as vertices. The size of each circle linking vertices relates to the support of a rule, while the colour indicates the lift.



support of a rule, while the colour indicates the lift. In Table 3.5, the importance measure of each rule is summarised, in order of confidence, lift and odds ratio (OR).

The association rule with the highest lift indicates that participants who had had security training through self study and also used the type of platform shown in the exhibit daily and were very familiar with the exhibit platform itself were highly likely to correctly identify a semantic attack on this platform. In other words, the rule antecedent "S3=1, FR=5, FA=3" was reported by 18% of the total participants in the survey, where 15% of the total participants who also reported these attributes correctly identified a semantic attack "RESPONSE=1"; resulting in a 85% confidence that these participants were not susceptible. Of course, this was expected. In respect to odds ratio, participants with these attributes were 82% less likely to be susceptible to a semantic attack. Here, a lift value of 1.1 means that the participants who were not susceptible (RESPONSE=1), who have security training through self-study, use

Rules	Support	Confidence	Lift	Coverage	OR
S3=1,FR=5,FA=3	0.15	0.85	1.1	0.181	1.82
CL=100,S3=1	0.15	0.84	1.091	0.181	1.71
S3=1,FA=3	0.21	0.84	1.091	0.248	1.78
FR=5,FA=3	0.21	0.83	1.083	0.247	1.69
CL=100	0.18	0.83	1.08	0.212	1.62
FA=3	0.28	0.83	1.075	0.34	1.69
S2=1,S3=1	0.16	0.83	1.073	0.193	1.53
S2=1	0.18	0.82	1.059	0.224	1.42
S3=1,FR=5	0.3	0.81	1.049	0.366	1.42
S1=1,S3=1	0.2	0.8	1.04	0.252	1.27

Table 3.5: Pruned Association Rules reported for participants with correct exhibit response

the target platform type daily and are very familiar with the specific platform, are observed 10% more than the percentage of the participants that were not susceptible in the total participant dataset. Within the 10 pruned rules, the appearance of frequently occurring items provides insight into association between specific attributes and reduced susceptibility to attacks. For example, familiarity with the specific platform provider (FA), frequency of access with a particular type of platform (FR), self-study (S3) and computer literacy (CL) are consistently reported attributes. On the contrary, duration of access (DR) and security awareness (SA) do not appear in the rules at the support/confidence threshold. Overall, the association rules indicate that security training through self-study, daily access to a type of platform and familiarity with a specific platform in this type category, as well as high confidence with computer literacy are associated to reduced susceptibility to semantic attacks. However, given the minor variations in lift between these rules, the lack of 100% confidence in any rule and relatively low support, a large proportion of non-susceptible users without these attributes may not be represented. Therefore, employing these rules as classification criteria would likely result in susceptibility prediction that produces many false negatives. Instead, we utilise these findings to identify the attributes that we should study in greater detail in stage 2 of our analysis, as presented in Section 3.5.

3.5 Stage 2: Experiment 2 Model and Feature Selection

Following on from the initial investigation of each high-level feature studied in experiment 1, experiment 2 was conducted on a smaller participant base (315 respondents), where participants were tested on a new set of semantic attacks, consisting of screenshots, animations and videos, and were asked to provide considerably more detail on their profile. In experiment 2, we add further granularity to the high-level features identified in the ten ARM rules of Section 3.4. In detail, we extend FR and FA to include both specific provider platforms (FR1, FA1) in combination with types of platform (FR2, FA2). Also, we adapt and extend security training as follows: S1, S2 and S3 are converted from a binary answer to a length of time since last training for S1, S2 and S3, with a scale of: Never, over 1 year, up to 1 year, up to 6 months, up to 3 months, up to 1 month, up to 2 weeks. The second measures security training by platform types and specific provider platforms, including length of time since last training. Each high-level security category is also extended to include the training methods commonly used for each respective security category, such as: self study (S3) through online videos, formal education (S1) through coursework, etc. Features SA and CL are not altered. In order to identify whether features DR and SA are truly non-informative, redundant features, we include them in the model feature selection process alongside the newly expanded, granular feature-set; extending DR to specific provider platforms (DR1 - platform type, DR2 - platform provider). As a result of expanding and adapting the feature-set, we increase from 8 candidate predictors in experiment 1 to 22 in experiment 2, as summarised in Table 3.6.

With the adapted feature-set from experiment 2, using *R* [275] and the *Caret* package [276], we identify machine learning models that can predict a user's ability

Table 3.6: Predictor variables utilised in Experiment 2

Feature	Description	Scale
FA1	Familiarity with provider platform	Not very familiar (1), Somewhat familiar (2), Very familiar (3)
FR1	Frequency of use for provider platform	Never (1), less than once a month (2), once a month (3), weekly (4), daily (5)
DR1	Duration of use for provider platform	None (1), < 30 min (2), 30 min to 1 h (3), 1-2 h (4), 2-4 h (5), >4 h (6)
ST1	Time since training for provider platform	Never (0), > 1 year (1), 6 months to 1 year (2), 3 to 6 months (3), 1 to 3 months (4), 2 weeks to 1 month (5), < 2 weeks (6)
FA2	Familiarity with platform type	Not very familiar (1), Somewhat familiar (2), Very familiar (3)
FR2	Frequency of use for platform type	Never (1), less than once a month (2), once a month (3), weekly (4), daily (5)
DR2	Duration of use for platform type	None (1), < 30 min (2), 30 min to 1 h (3), 1-2 h (4), 2-4 h (5), >4 h (6)
ST2	Time since training for platform type	Never (0), > 1 year (1), 6 months to 1 year (2), 3 to 6 months (3), 1 to 3 months (4), 2 weeks to 1 month (5), < 2 weeks (6)
S1_1	Formal Education through lectures	No (0), Yes (1)
S1_2	Formal Education through tests	No (0), Yes (1)
S1_3	Formal Education through coursework	No (0), Yes (1)
S1T	Time since formal education	Never (0), > 1 year (1), 6 months to 1 year (2), 3 to 6 months (3), 1 to 3 months (4), 2 weeks to 1 month (5), < 2 weeks (6)
S2_1	Work-based through tests	No (0), Yes (1)
S2_2	Work-based through videos	No (0), Yes (1)
S2_3	Work-based through games	No (0), Yes (1)
S2T	Time since work-based training	Never (0), > 1 year (1), 6 months to 1 year (2), 3 to 6 months (3), 1 to 3 months (4), 2 weeks to 1 month (5), < 2 weeks (6)
S3_1	Self-study through websites	No (0), Yes (1)
S3_2	Self-study through videos	No (0), Yes (1)
S3_3	Self-study through games	No (0), Yes (1)
S3T	Time since self-study training	Never (0), > 1 year (1), 6 months to 1 year (2), 3 to 6 months (3), 1 to 3 months (4), 2 weeks to 1 month (5), < 2 weeks (6)

Table 3.7: Total number of times each predictor feature is selected for an attack exhibit's best performing model

Exhibits	FR1	FR2	FA1	FA2	DR1	DR2	ST1	ST2	SA	CL	S3T	S2T	S1T	S3_1	S3_2	S3_3	S2_1	S2_2	S2_3	S1_1	S1_2	S1_3
E1	✓	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
E2	✓	-	✓	-	✓	-	-	-	✓	✓	-	✓	-	-	✓	-	✓	-	✓	-	-	-
E5	-	✓	-	-	-	-	✓	-	✓	✓	✓	-	✓	-	✓	-	-	-	-	-	✓	-
E7	✓	-	-	-	-	-	-	-	-	-	✓	-	✓	-	-	-	-	-	-	-	-	-
E11	✓	-	-	-	✓	-	-	-	✓	-	✓	-	✓	✓	✓	-	-	-	-	-	-	✓
E12	✓	-	✓	-	-	✓	-	-	-	-	✓	-	✓	✓	-	-	-	-	-	-	-	-
Total	5	2	2	0	2	1	1	0	3	2	4	1	4	2	3	0	1	0	1	0	1	1

to detect attacks. Firstly, we select and compare two distinct machine learning algorithms; modelling both a linear and non-linear approach to prediction. Secondly, for each model we have applied automatic feature selection with sequential backward selection in Recursive Feature Elimination (RFE); obtaining an optimal model for each machine learning algorithm.

3.5.0.1 Susceptibility Prediction: Logistic Regression Vs. Random Forest

For a user susceptibility model to be practically usable by a technical security system, it must employ predictor features that can be practically measured in real-time, automatically and ethically. As our experiment sample includes each participants exhibit classifications, these responses are used to label each participants predictor features in the sample; which is suitable for supervised learning models. To evaluate whether a linear model can be sufficient in predicting susceptibility, we first employ logistic regression (LR), which performs well in linear spaces, functioning by definition as a special case generalised linear model using a Bernoulli distribution for a binary response [277]. LR is relatively robust to noisy data and over-fitted models, where the data contains high variance. By computing a LR model we receive an estimation of the probability of a binary response Y (e.g., dependent variable) as $P(Y)$, in this case where $Y = 0$ (i.e., susceptible) or $Y = 1$ (i.e., not susceptible), based on one or more predictor (or independent) features X. The result is a log odds (logit) coefficient which is either positive or negative and represents the effect that the predictor(s) has on the outcome, respectively. The LR of binary response Y on features X_1, ..., X_k estimates the parameter values for the coefficients of each feature $\beta_0, \beta_1, ..., \beta_k$ using a maximum likelihood estimation to establish a multiple linear regression function as the following formula:

$$logit(p) = log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + ... + \beta_k X_k \tag{3.8}$$

To find the probability response for the logistic regression (which is the probability of the binary response outcome, given a one unit increase in the primary independent feature variable with all other features held constant), which is the solution for P, the formula is:

$$\hat{P} = \frac{e^{\beta_0 \sum \beta_k X_k}}{1 + e^{\beta_0 \sum \beta_k X_k}} \tag{3.9}$$

In comparison to LR, another method that is resilient to variance in model predictions is an algorithm known as bagging (also known as Bootstrap Aggregating [278]), where the algorithm produces replicates of an original data sample by creating new datasets by random selection with replacement. With each dataset, multiple new models are constructed and gathered to form an ensemble of models. Within the prediction process, all of the models in the ensemble are polled and the results are averaged to produce a result that is low in variance. Random forest (RF) is a popular bagging algorithm that can also be described as an ensemble decision tree classifier. In RF, a number of decision trees are trained with different re-sampled versions of an original dataset and then used to predict data that was omitted from each sample as an embedded measure of training accuracy; this is called the out-of-bag error. To avoid biases through collinearity in the feature space, Random Forest implements a facility for randomly selecting a range of feature variables for splitting at each decision node in a decision tree. The following equation provides a high level overview of the bagging and Random Forest training and prediction process:

For trees $t = 1, ..., T$: sample with replacement n training sets from X, Y, then train a decision tree P_t on X_t, Y_t. Where dependent variable $Y = c$, which is the binary response class or probability (if defined), and f is the set of features to be used in each decision tree, the final trained RF model is summarised as:

$$P(c|f) = \sum_{1}^{n} P_n(c|f) \tag{3.10}$$

After training of all decision trees, predictions on unseen samples (e.g., test data) x' is made by taking the majority vote (e.g., average class distribution) amongst all decision trees on x':

$$\hat{P} = \frac{1}{T}\sum_{t=1}^{T} P_t(x') \qquad (3.11)$$

Here, RF reduces the high variance inherent in a decision single tree by creating n trees that are averaged to reduce the variance of the final model [279]. Unlike LR, RF handles non-linearity naturally. Predictor variables are randomly chosen at each decision split in the decision tree which results in a randomised, non-linear approach. As compensation for loss of model interpretability that is available in simpler, linear statistical learning algorithms such as Logistic Regression, Random Forest is relatively simple to optimise and tune to most types of data and has been shown to be an excellent general purpose machine learning algorithm, overall outperforming 179 different machine learning classifiers across 121 different datasets in a comprehensive study carried out by Delgado et al [280].

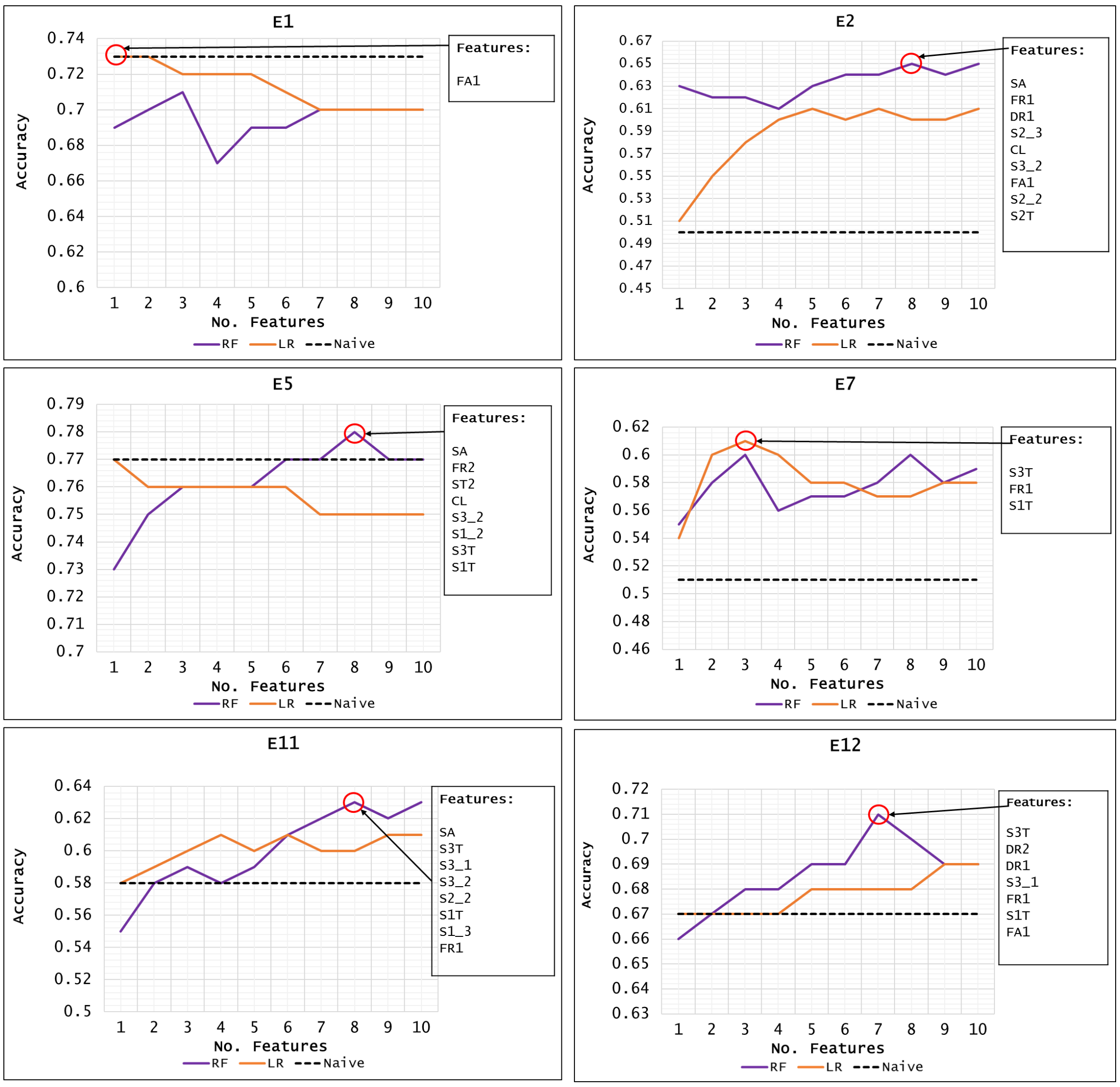
3.5.0.2 Recursive Feature Elimination

Employing the predictor features summarised in Table 3.6, for both LR and RF models, we have used an automatic feature selection method to identify the most informative predictor features and build a single prediction model for each individual attack exhibit. Recursive Feature Elimination (RFE) is an automatic backwards feature selection algorithm. It starts by fitting a model to all 22 features, ranking the latter based on their variable importance to the model, and gradually excluding the features with the lowest importance in each iteration, recursively considering smaller and smaller feature sets. In RF, variable importance is calculated within the model by recording the out-of-bag prediction accuracy for every predictor variable permutation in each decision tree. At each feature iteration, model accuracy is compared between the prior and permuted model, averaged over all trees and then normalised by the standard error. Since LR has no model-specific method to estimate importance, the *Caret* package conducts receiver operating characteristic (ROC) curve analysis on each feature iteration by evaluating the area under the ROC curve (AUROC), which is used as the variable importance for LR [281]. A ROC curve illustrates the performance of a binary classifier at different prediction probabilities by plotting the true positive rate (TPR) against the false positive rate (FPR) at various thresholds. AUROC represents the area under the ROC curve, where a random guess area of 0.5 (0,0 to 1,1) is typically used as the reference area from which to evaluate model performance. The result of LR is the selection of those features that have a statistically significant impact on the probability of a user's correct prediction.

One possible drawback of RFE is the potential for over-fitting to predictor variables, as the procedure can focus on nuances in the sample data that may be anomalous and therefore not present in future data. For example, where predictors randomly correlate with the dependent variable being predicted, RFE may assign a good importance ranking to these variables, even if they were to make no practical sense. During training, this would indeed lower prediction error, but when validating the model on new data it might reveal that the predictors are actually non-informative, in a case referred to as "selection bias" [282]. To avoid this problem, and as is standard practice in supervised machine learning experiments and models, we have employed an outer layer of resampling through a repeated 10-fold cross-validation to provide a robust estimate of model feature-selection and test error as evaluated by RFE. Cross-validation (CV) is a model validation technique for assessing model performance on unseen, independent data sets and is an important tool for avoiding exaggerated model accuracy results (e.g. over-fitting a model by testing it on the same data the model has been trained on). In the 10-fold CV process, the data sample is partitioned into 10 equal folds, where nine folds are used to train the model and the remaining one fold is used to test it. This process is repeated 10 times so that the model is tested on each fold in order to produce an average model test error, which in our case reports model test error at each variable selection step in RFEs backwards selection process. With repeated 10-fold CV, for each 10-fold training process, the process is repeated another 10 times.

In Figure 3.8, we present the results of LR and RF CV test error for each attack and the optimal set of predictors selected by RFE. We compare LR and RF with each other, as well as with a naive classifier, which, for each exhibit would always select

Figure 3.8: Each graph presents the results of 10 times repeated 10-fold cross-validation for each exhibit, using recursive feature elimination with a LR and a RF model. Results are presented in the form of overall test accuracy



the answer (0 or 1) that is the most common in the sampled population (the sample response rate). This is the maximum accuracy of a model that uses no features for predicting the sample population outcome. For five out of six attack exhibits, both LR and RF models reported superior classification accuracy than the sample response rate. RF outperformed LR in four of the six attack exhibits.

Table 3.7 (see page 94) shows where each feature was selected for an exhibit's final prediction model (whether it was the LR or the RF model that was best performing). We observe that frequency of access to the specific provider platform in the exhibit (FR1) was included in the best performing prediction model for 5 out of 6 exhibits, followed by length of time since security training through self-study and formal education, which appeared in 4 out of 6 exhibits' final models.

On the other end, familiarity with the exhibit platform type, security training with a particular platform type, security self-study through games, work-based through tests and formal education through lectures were not selected for any exhibit's final prediction model. Removing these five features, we prune the candidate-feature set from 22 to 17 within a final RFE model selection process process with the aim to build a final model for susceptibility prediction. In order to build a prediction model that can potentially be employed across any platform and with any semantic attack, we combine each of the exhibits' sample responses into a stacked data sample, where all users' responses are included in a single dataset for all attacks. So, the values for each feature relate to the particular attack's settings in a particular entry in the dataset. For instance, the feature "familiarity with platform type" in an attack that utilises Facebook would refer to the familiarity with platform of type "social network". This approach enables the construction of a single model that contains a range of semantic attacks, platform types and specific provider platforms. Creating a single model for each attack would be impractical, as we would need one model for each platform/attack combination. Training a model based on a wide-range of disparate platforms and attacks, and using a combined dataset for a single prediction response, enables more widely applicable prediction of susceptibility that can be utilised in a technical security system.

3.5.1 Susceptibility Model: Results and Analysis

A reliable and widely applicable user susceptibility prediction model can have several applications as part of a defence mechanism against semantic attacks. It can help predict a specific user's a) degree of "susceptibility" to semantic attacks (likelihood of being deceived by one), or equivalently b) expected performance if they were to act as a human classifier (likelihood of spotting attacks). The former can help a security system identify whether a user is particularly susceptible to semantic attacks and consequently whether the system environment needs to adapt accordingly (e.g., by privilege adjustment, targeted warnings, security enforcing functionality, etc.). The latter can help evaluate to what extent a user can be relied upon as a "Human as a Security Sensor" (HaaSS) of semantic attacks, where user reports are taken into account so as to strengthen an organisation's cyber situational awareness.

In Tables 3.8 and 3.9 the tuning and configuration parameters for each of the final LR and RF models are provided. For both models, due to the relatively small data sample, we have opted to use 80% of the participant dataset for each models training sample; validation testing each model with a 10-fold cross-validation. Each model is then tested for actual accuracy on unseen data using the remaining 20% as the test sample. In RF, for the optimum number of random variables to try at each decision tree split (e.g., dependent variable 0,1 association) we use the recommended default which is square root of the total number of features in the model. The number of trees generated for RF was set to the default of 500; no improvement in model access was observed with an increased number of trees. By default, RF employs embedded re-sampling with replacement, as part of the bagging (bootstrap aggregation) process. The LR model uses a primary independent variable, with all other features set to static values in order to compute class probabilities; here we have employed FR1 which received the highest variable importance in the RFE feature selection process. Feature interaction was not used so as to preserve the interpretability of the LR model.

For both applications, it is important to measure the model's performance based on its general accuracy in predicting which participants will detect the attacks and which will not, and secondly its ability to reduce false positives or false negatives by using a probability cut-off threshold. Table 3.10 compares the LR and RF classifiers' overall performance against the naive classifier, which always selects the answer with the highest probability in the population sample (so, always 0 if population's success rate is below 50% and always 1 otherwise, for a given exhibit). The test split used for classification consisted of 215 correct (1) and 147 incorrect (0) responses. In

Parameter	**Value**
Model Type	Classification
Parallel Processing	True
Validation	10-Fold cross-validation
Train split	0.8
Test split	0.2
Number of feature sets ($= \sqrt{\text{no. features}}$)	4
Number of trees	500
Sample replacement	True
Class Probabilities	True
Metric	Accuracy (ACC)
Decision threshold (δ)	0.5

Table 3.8: Random Forest model configuration

Parameter	**Value**
Model Type	Binomial (General Linear Model)
Parallel Processing	True
Validation	10-Fold cross-validation
Train split	0.8
Test split	0.2
Primary individual variable	FR1 (Frequency with provider)
Feature Interaction	False
Class Probabilities	True (default)
Metric	Accuracy (ACC)
Decision threshold (δ)	0.5

Table 3.9: Logistic Regression model configuration

Table 3.12, the predictors selected by the RFE process for the LR and RF models are presented. For the RF model, to evaluate feature variable importance, as metric, we use the reduction in out-of-bag error during the model training process. For the LR model, we use the increase in AUROC.

3.5.2 Key observations

Both the LR and RF models satisfy the statistical significance threshold of 0.05 and both appear to outperform comfortably the naive classifier, which is a good sign in terms of their practical applicability. There is a slight advantage of RF over LR across all metrics (higher accuracy and precision, and lower false positives), but this comes

Classifier	**Accuracy**	**Precision**	**FN**	**FP**	**P value**	**Features**
RF	0.71	0.72	0.10	0.19	<0.001	16
LR	0.68	0.70	0.12	0.20	<0.001	7
Naive	0.59	0.59	0	0.41	0.5	0

Table 3.10: Prediction performance comparing the accuracy, precision, false positive and false negative detection of the final LR and RF models against the Naive (i.e., no information detection rate) classifier. See B.1 for R console output of prediction test accuracy

Features	Intercept	FR1	CL	S3T	S2_2	DR2	S1T	S3_1
OR	0.20	1.22	1.01	1.10	0.57	1.13	1.06	1.29

Table 3.11: Feature Odds Ratios for Logistic Regression Model

Model	Selected Features
Random Forest	CL (0.114), S3T (0.112), SA (0.107), FR1 (0.099), FA1 (0.097), S3_1 (0.087), S1T (0.073), S2T (0.073), S2_2 (0.068), FR2 (0.06), DR1 (0.05), S2_3 (0.048), S3_2 (0.048), DR2 (0.046), ST2 (0.039), S1_3 (0.033)
Logistic Regression	FR1 (0.035), CL (0.032), S3T (0.029), S2_2 (0.027), DR2 (0.023), S1T (0.021), S3_1 (0.018)

Table 3.12: RF and LR model predictor features selected through recursive feature selection (in order of variable importance measure: decrease in out of bag error rate for RF, and increase in AUROC for LR). The higher the variable importance (in brackets), the more important the feature is to its model.

at the expense of practicality, because it requires a large number of features to be monitored (16 against LR's only 7). Moreover, as RF employs a black box modelling approach, this makes it less interpretable than the LR model as to why each feature within the model informs prediction. In LR, interpretation is more straightforward, because it produces each feature's odds ratio (OR), which is the increase in the probability of a user correctly identifying an attack for every one unit increase in that feature's scale, when all other features remain fixed. For example, from table 3.11, we see that a unit increase in the scale of frequency of use (e.g., from once a

Figure 3.9: Random Forest model performance for false positive (not susceptible), false negative (susceptible) prediction and overall prediction accuracy at each probability cutoff

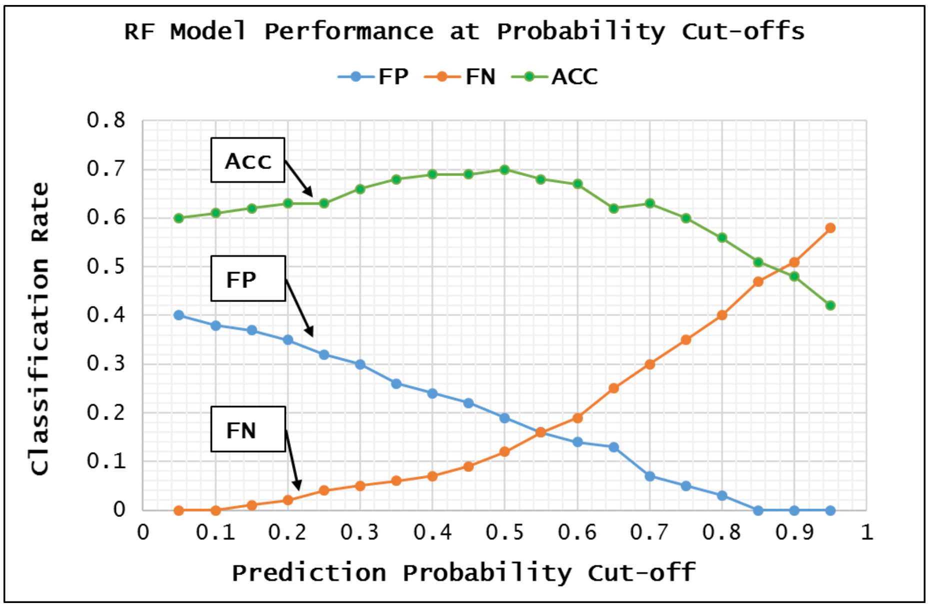
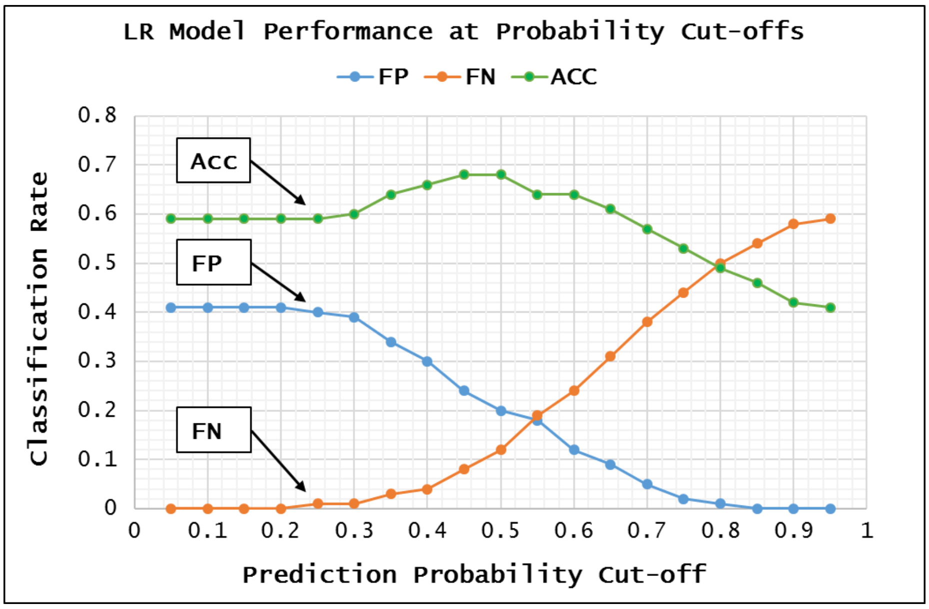


Figure 3.10: Logistic Regression model performance for false positive (not susceptible), false negative (susceptible) prediction and overall prediction accuracy at each probability cutoff



month to weekly), increases the probability of correct detection by 22%. So, ORs can also be used to cross-reference with variable importance in interpreting each feature's influence to the prediction outcome.

As one would have expected, computer security training does make a difference, with all three forms (formal education, work-based training and self-study) appearing in some form in both models. In general, we observe that the length of time since last training (whether self-study, formal education or work-based training) is particularly important, with time since last self-study (S3T) appearing to be overall the most important in the training category. This is reasonable, because semantic attacks evolve continuously and any guidelines or technical information learned in training needs to be updated often. Five years ago, semantic attacks were almost entirely based on generic phishing and ransomware. Today, watering holes, WiFi evil twins and social media friend injection attacks have become the norm, and phishing has expanded to all forms of user interaction, from Quick Response (QR) codes, to near-field communication (NFC) and Bluetooth [9]. Interestingly, formal security education through lectures was not chosen as a useful predictor of susceptibility to semantic attacks by any of the models and for any of the exhibits.

Frequency of access to the specific provider's platform (FR1) rather than generally to the type of platform (e.g., specifically Facebook rather than generally social networks) was shown to reduce susceptibility noticeably, being the fourth most important variable in RF and the first in LR. Frequency of access to the general type of platform (FR2) was utilised by RF as one of the features with the lowest variable importance (0.06), and was not utilised at all by LR.

Duration of access to the same platform type was important in both models, with 13% increase in the probability for each unit increase in the LR odds ratio. In RF, frequency and duration was also important for the platform type. Also, in the RF model familiarity with the platform provider was the fifth highest important variable.

Computer literacy (CL) was shown to be the most important feature for RF and the second most important for LR. This reinforces the need for a mechanism to monitor and record computer literacy as a gauge of an organisation's cyber risk.

Unlike the high level predictors Security Awareness (SA) and Duration of Access (DR), which were not included in the association rules in experiment 1, the RF model included both Security Awareness and the expanded DR features: duration of using a specific platform provider (DR1) and specific platform type (DR2), whereas the LR model included DR1 only. Surprisingly, in RF, SA was the third most important feature, whereas in LR it was not included at all; as per the association rules. On the other hand, more in line with the association rules omitting DR in experiment 1, both DR1 and DR2 were were given relatively low variable importance in RF, placing 11th and 15th out of the total 17 features, respectively, with DR2 placing 5th out of a total 7 features for LR and DR1 omitted from the models feature-set. For both models, SA, DR1 and DR2 were given a lower degree of variable importance than all other features that were expanded from their higher level counterparts (FR, FA, CL, S3, S2, S1) reported in experiment 1's association rules frequent item sets; with the exception of time since last security training through formal education (S1T), which slightly less important than DR2 (0.21 compared to 0.23) in the LR model. The indication is that the original high level predictors show a consistent association with reduced susceptibility across both experiments, even after adding further granularity to their measurement scale and context, and as a result also gained sufficient predictive power for determining the probability of a participants susceptibility to semantic attacks with a reasonable degree of accuracy.

There is no doubt that a user's susceptibility to semantic social engineering attacks depends also on personality traits, social context, psychological state and other human and contextual factors, which are, however, impractical, as they cannot be measured in real-time, automatically or ethically. Without knowledge regarding these factors, one cannot expect a highly accurate prediction of susceptibility. So, the accuracy improvement of around 10% against the naive classifier achieved here is significant. In practice, we have developed this method to act as a baseline for an organisation's technical security system, which can then adapt over time, as it learns the characteristics of the organisation's own users.

Equally significant is that one can utilise these models to identify an appropriate probability threshold depending on preference in minimising false positives, minimising false negatives or maximising accuracy (Figure 3.9 and 3.10). By probability threshold, we refer to the value over which a technical security system should consider a user to be susceptible to a semantic attack. For instance, if the aim were to maximise accuracy, the probability threshold for determining whether a user is susceptible or not, should be 0.5 for both models. However, it would be 0.55 if the aim were to keep both false positives and false negatives below 0.2. Overall, RF appears to perform slightly better than LR in terms of false positives at low probability thresholds, but is slightly worse at higher probability thresholds. For false negatives, the reverse is observed. In an organisation that is tolerant of false positives, but not tolerant of false negatives, to keep the false negatives below 0.02, both LR and RF models would yield a false positive rate just under 0.4. For RF, this would correspond to a probability cut-off of 0.15, and for LR to a cut-off of 0.3. For an organisation that is tolerant of false negatives, false positives can be effectively avoided using the RF model at a 0.85 cutoff, but this results in an approximate 20% decrease in overall classification accuracy, with the number of false negatives increasing to 0.48.

3.6 Discussion

3.6.1 Limitations

In our exhibit-based experiment there are a few limitations that must be considered. Participants were primed to the purpose of the survey and subsequent test and thus may have been more vigilant and sensitive to a semantic attack's deception (therefore weakening its effect) than they would have normally been.

For the first experiment, the simple approach of using screenshots to represent the exhibits was very useful in conducting a large-scale study online and on any computer platform. However, the use of screenshots is more appropriate for DV1 attacks that rely on cosmetic deception than for DV2 (and partially DV3) that rely on behavioural deception, which is less straightforward to convey via screenshots. To address this, in the second experiment we included video exhibits where behavioural deception can be more accurately emulated (in terms of context and system behaviour), rather than depicted visually.

Potential limitations may also exist in the selection of features for our susceptibility model. We have focused on a number of high-level concepts with the aim to create a model for predicting susceptibility that is applicable across a wide range of semantic attacks. One example is computer security training where we focus on the type and mode of delivery of security training rather than its content. Prediction taking into account the content too would have probably been more accurate, but would presume that an organisation can collect such detailed information for its users, which may be impractical. However, as training content is often attack specific [147, 25, 261, 256] (e.g., phishing emails/websites), which limits users' exposure to other threats, alternatively measuring whether security training has been received training for different types platform (i.e. ST2 - e.g., Social Media) can help address threats observed on similar types of platform that share characteristics (e.g., taxonomy attack vectors in chapter 2); thus covering a wider range of attacks. The range and type of susceptibility predictor features we have identified is by no means exhaustive, but by employing a series of high level user susceptibility indicators - identified as practically measurable in a technical system, we have demonstrated the feasibility for accurately modelling user susceptibility against semantic attacks. It is also possible that in our experiment some features were selected and others omitted due to the limited profiles of the experiment participants, however in general we expect this less of a problem as the best performing classifier selected 17 out of a total 22 of features.

3.6.2 The viability of self-efficacy features in our model

It is important to evaluate the general reliability of self-efficacy features included in our user susceptibility model in order to determine whether they can be accurately and practically validated. In Table 3.13 we have conducted a number of correlation-based analyses to identify relationships between user activity (e.g., auditable features) and contextually related self-efficacy rating. We also test the ability to predict a user's self efficacy by developing a number of Random Forest models. For the correlation analysis, as per best practice, between ordinal categorical variables we employ spearman correlation [283] and between continuous and ordinal categorical variables we employ polyserial correlation [284].

Self-efficacy	**Auditable**	**Corr**	**Method**
FA1	FR1	.69	Spearman
FA1	FR2	.31	Spearman
FA1	DR1	.65	Spearman
FA1	DR2	.22	Spearman
SA	S3T	.33	Polyserial
SA	S2T	.36	Polyserial
SA	S1T	.39	Polyserial
SA	ST1	.36	Polyserial
CL	FR1	.11	Polyserial
CL	DR1	.11	Polyserial
CL	FR2	.20	Polyserial
CL	DR2	.21	Polyserial
CL	FA1	.23	Polyserial

Table 3.13: Correlation analysis of association between self-efficacy and auditable user susceptibility features (all results were statistically significant at the 0.001 level - see B.2 for R studio output)

Unsurprisingly the results show that the level of familiarity a user associates with a specific platform (FA1) is highly positively correlated with the frequency and duration with which the user accesses it. On the other hand, a low to medium positive correlation was found to exist between specific platform efficacy and the frequency and duration that a users accesses that general type of platform. These results demonstrate that a user is very likely to associate a level of familiarity with a platform the more often and longer that they use it, and this is also influenced to some degree by using platforms of a similar type. For computer security awareness (SA), all security training types reported a medium positive correlation, whereas for computer literacy (CL) frequency and duration features reported low positive correlations. The lower correlation results for security awareness and computer literacy indicate that individual security training and platform usage features alone are not sufficient to fully explain a user's selection of these self-efficacy criteria.

In Table 3.14, using a Random Forest classifier we review the prediction accuracy for each self-efficacy feature based on each of the auditable features tested in the correlation analysis.

Self-efficacy	**Model**	**Test Error**	**Metric**
SA	SA$\sim$S3T+S2T+S1T+ST2	.19	RMSE
CL	CL$\sim$FR1+FR2+DR1+DR2	.25	RMSE
FA1	FA$\sim$FR1+FR2+DR1+DR2	.30	Accuracy

Table 3.14: Prediction test error of self-efficacy features using auditable features - see B.3 for R studio output

The relatively low model test error for predicting platform familiarity, computer security awareness and computer literacy user self-efficacy demonstrates a strong association between auditable features and their direct influence on users' self-assessed skill. Notably, computer security awareness achieved the lowest test error, which means that even though individual security training is not highly correlated with users' self-efficacy, overall aggregate security training appears to have a strong affect on users' perceived ability. In general, the model test results suggest that the self-efficacy features included in the user susceptibility model are strongly associated and influenced by user activity attributes that can be measured accurately on a technical system. As a result, employing contextually related activity features as validation criteria for self-efficacy can potential help to reduce the effect of bias and also help to highlight abuse of self-assessment were users claiming a level of efficacy that is not evident in their auditable usage pattern.

3.6.3 Challenges in producing datasets for semantic social engineering susceptibility prediction

Real-world, authoritative datasets for user susceptibility to semantic attacks are not available. An organisation may not publicly reveal that their business has been exploited because of the perceived reputational damage it could cause or simply because employees fail to report breaches for fear of disciplinary actions. Security authorities and organisations, such as Symantec [285], who actively publish data from those businesses, and users who do report attacks tend to anonymise and censor the data to a point that profiling information that could show context leading to an attack is removed before being made publicly accessible. Therefore, development of user datasets through research experiments is necessary to understand which behaviours and identifying factors help determine susceptibility and thus inform the design and development of new security mechanisms against semantic attacks. In this section, we identify a number of persisting problems for the development of robust semantic attacks datasets:

- **Ethics**. A prevalent limitation for access to user susceptibility data is ethics. Ethical consideration and approval can be a barrier to the collection of rich user data for aiding researchers and developers in the development of user-centric defences against semantic attacks. Experiments with human participant require ethics approval from an institutional of governmental review board, and therefore there are often a number of requisite requirements which limit researchers ability to produce truly representative results. For example, in [286], participant deception and debriefing, privacy and institute review board approval were determined to be the main challenges that affect the design and execution of phishing experiments. Mouton et al [287] proposes a normative perspective for ethics in social engineering which can help ethics committees in the process of experiment approval. Here, reporting susceptibility would be considered from a utilitarian and deontogical standpoint; that is, whether or not the collected and reported data would be ethical given the consequences of the specified action (utilitarianism) or the duty and obligations related to that action (deontology). In [288], researchers developed what has become widely accepted approach for designing ethical social engineering experiments, but the method proposed focuses solely on phishing emails and it is unclear how it can be extended to a wider range of semantic attacks and platforms other than email.

As well as ethics approval, semantic attack research poses legal implications [289], where researchers are increasingly conducting phishing experiments without the knowing consent of participants. In this case, the data collected may prove more representative of natural user behaviour, but cannot be validated as legitimate research without formal approval.

One approach towards tackling this fundamental problem in the research of semantic social engineering attacks is to provide a platform that enables users and organisations to anonymously report semantic attacks, without omitting crucial contextual information such as whether the attack was successful or not, the scenario in which the attack occurred, whether or not the target had been trained, etc. This database of user susceptibility information would provide an invaluable resource for researchers seeking to analyse trends or predict behaviour to semantic attacks. Most importantly, collection of data in this format removes the complexity and damaging effect on user experiment data that ethics approval may require.

- **The experiment population against data collection detail trade-off**. Participants in semantic attack research tend to be recruited from the institution in which the study is conducted (e.g., university students, organisation's own staff, etc.) and often this is noted as a limitation of the research as the results may not be representative the wider target population. This poses a major problem for empirically proving the validity of research outcomes. In the first study, we recruited a large number of participants from multiple different geographical locations on the Internet, but this approach limits the ability to collect more detailed data from the participants. There is a trade-off to be considered when recruiting participants that are more representative of the user base against the qualitative data that can be extracted from a user population that is easily accessible. In the case of the former, collecting user responses from a large number of disparate demographic backgrounds is fairly simple when the Internet is the recruitment platform, but these participants cannot be easily observed or interviewed at any stage of the research. For the latter, researchers have localised access to participants and therefore a higher degree of detail regarding user behaviour can be recorded and analysed.

Ultimately, semantic attack research is affected by both circumstances and as such context should dictate the most suitable approach. In our study, it was more important to recruit sufficient numbers of participants to allow the evaluation of statistical machine learning models. For research focusing on psychological impact of exploitation from semantic attacks, detailed qualitative data may be a more relevant goal, in which case research would most likely benefit from a smaller population.

- **Attack coverage**. In table 3.1, the majority of research related to predicting susceptibility to social engineering attacks has focused on phishing, which is only one type of semantic attack. Conclusions made from research solely reliant phishing experiments may not be applicable to the wider semantic attack problem space. As with network and operating system attacks, there are many types of semantic attack, crossing multiple platforms, and therefore like an anti-virus for an operating system or firewall for the network, it is crucial that experiments consider and evaluate a wide range of semantic attacks in order to build defence systems that can mitigate multiple threats. Furthermore, specific attacks may become less popular over time as new platforms emerge or more successful techniques are developed, and therefore it is also important that an experiments results remain relevant for addressing future attacks.

  In chapter 2, the taxonomy for semantic attacks can provide a useful baseline to build such experiments that measure user susceptibility across a series of generic attack attributes. In this study, we have included *Deception Vector* only for clustering attacks on the same and different platforms, simplifying the modelling process and ability to classify susceptibility with a single, general model. For research aiming to understand user vulnerabilities to removable media or targeted cyber-supply chains, other items of the taxonomy such as

Method of Distribution and *Target Description* may provide useful categories for clustering a wide range of attacks in a single experiment.

- **Lack of an authoritative archive**. Repositories of historic and current phishing emails and websites do exist [290, 291, 292], but do not cover the wider range of semantic attacks and do not include data on the profiles of the users who have or have not been deceived by them. An open archive of semantic attacks and corresponding user profile data would be immensely helpful to researchers in this field.

3.6.4 Human as a Security Sensor (HaaSS)

The concept of the *human as a sensor* has been used extensively and successfully for the detection of threats and adverse conditions in physical space, for instance to detect noise pollution [293], monitor water availability [294], detect unfolding emergencies [295] etc. In relation to semantic attack threats, the concept is very new. There is one example specifically for phishing attacks [245]. We argue that the concept can be explored much further and for most semantic attacks, where the human user's situational knowledge can help detect attacks that are otherwise largely undetectable by technical security systems. For example, there are no known technical countermeasures to attack E11 in experiment 1 ("Qrishing") and attack E1 in experiment 2 (Video masquerading "clickbait"), but in our experiments, users were able to detect them with a probability of 86% and 78% respectively (see Table 3.2). This is certainly not a rigorous way for evaluating HaaSS, but we feel is an indication of its potential. Introducing a HaaSS element in an organisation's security can empower users to become its strongest link. In this context, predicting the performance of an individual user as human sensor of semantic social engineering attacks is the equivalent of measuring the reliability of a physical sensor. For example, within a HaaSS reporting platform, a prediction model that measures the probability of a user's report being correct can provide security engineers with the ability to triage the review of reports; prioritising the ones from users that are more accurate human sensors.

3.7 Conclusion

We have conducted two experiments, each consisting of a survey and an exhibit-based test, asking participants to identify whether specific exhibits were likely to show attacks or not. Based on the data collected, we identified a set of features from which we produced logistic regression and random forest models for predicting susceptibility to semantic attacks, with accuracy rates of .68 and .71 respectively. The slight performance advantage of RF over LR is countered by the larger number of features that it requires to be monitored (16 against LR's 7), but also explains that the relationship between features and a user's susceptibility is not strictly linear. In terms of the features themselves, we observe that security training makes a noticeable difference in a user's ability to detect deception attempts, with frequent self-study appearing to be a key differentiator. Yet, formal security education through lectures was not chosen as a useful predictor by any of the models and for any of the exhibits. More important features were computer literacy, familiarity and frequency of access to a specific platform. The models developed can be configured in terms of preference in minimising false positives, minimising false negatives or maximising accuracy, based on the probability threshold over which a user would be deemed to be susceptible to an attack. For both models, a threshold choice of 0.55 would keep both false positives and false negatives below 0.2; minimising equally the number of incorrect predictions of non-susceptibility and susceptibility, respectively.

We have also identified a number of challenges associated with developing datasets for predicting susceptibility to semantic attacks, where addressing these challenges can help produce rich and representative user susceptibility data that can aid developers and researchers of user security defence systems. In future work, our model can be experimentally validated with a technical implementation and using a wider range of semantic attacks for each deception vector in order to provide empirical results for the model's performance in practice. As deception-based attacks utilised in the wild evolve continuously, the baseline model and classification rules can be continuously improved with new training data from different user populations and attack types.

The advent of the Internet of Things [13] promises to compound the problem of semantic attacks and extend to physical impact, exposing user interfaces of systems previously inaccessible to the standard user, let alone via a distributed application in the Internet [14]. The more effective such cyber-physical attacks prove, the more the deception attack surface will grow. Semantic social engineering threats in the Internet of Everything are likely to expand attack surfaces via ubiquitous connectivity which practically facilitate new and convincing semantic attacks; the impact of a phishing email may no longer be limited to stolen user credentials or malware infection, but can also bring down a national power-grid [296]. Providing users with the ability to report suspected semantic attacks can help provide system developers and security practitioners with key insights in how to design or update systems to mitigate such threats, while at the same time instilling users with a sense of empowerment in protecting their technological environment. To this end, report credibility provides a crucial role in identifying the likelihood that an attack has indeed occurred, so as to prioritise reports and utilise their information to augment defence mechanisms. Predicting user susceptibility as a performance measure of semantic social engineering attack reporting provides a first step towards this vision.

Chapter 4

A Human-as-a-Security-Sensor framework and a prototype implementation for detecting semantic social engineering attacks

In chapter 3, we combined two of the three key defence techniques highlighted in chapter 2, user awareness and machine learning, to develop a predictive model of user susceptibility to semantic attacks. By identifying and measuring features associated to users' semantic attack detection efficacy, we demonstrated that it is feasible to predict user performance in detecting semantic attacks across a range of different attack vectors and disparate computer platforms. We then proposed the concept of Human-as-as-Security-Sensor (HaaSS), where we argue that a user's situational knowledge can help in the detection of threats alongside existing technical security systems (e.g., phishing), as well as the detection of attacks that are otherwise largely undetectable by technical security systems (e.g., multimedia masquerading).

In this chapter, we expand on the concept of Human-as-a-Security-Sensor (HaaSS) by unifying the three key defence techniques (User Awareness, Machine learning and Sandboxing) identified in chapter 2 into a single defence framework for detecting semantic attacks. The proposed framework forms the architectural blueprint of a technical platform which utilises users as human sensors for the detection of semantic attacks. Using the proposed framework, we take the first steps towards exploring the applicability of the HaaSS concept for semantic attack detection in an empirical context, by testing the reliability of human users as security sensors in an experiment, where we introduce *Cogni-Sense*, a proof of concept HaaSS platform implementation derived from our prototype framework.

4.1 Human-as-a-Security-Sensor: a technical framework

There is a growing realisation in the security industry that the users need to be at the core of any system's security design [297, 298, 299, 300, 301, 302]. Our aim is to progress a step further and empower users to directly contribute to the security of themselves, their organisation or the wider community actively. However, to capture the detection of semantic social engineering attacks, it is users which require an interface which provides functionality to report suspicious or anomalous activity that uses deceptive attack vectors, rather than relying on technical exploitations; for which the human user often is a more accurate sensor than an organisation's technical security systems.

From a system standpoint, humans are autonomous, multi-sensory systems with the ability to produce inferential output data based on multiple experiential and environmental input data. Within human society, therefore, this ability means that humans (as physical sensors) are often best placed to provide information in various contexts where technical systems alone are not adequate. The aim of HaaSS (and indeed most user-driven defences) is not to replace technical security systems, especially those that have been shown to work well in detecting and mitigating certain semantic attacks (e.g., phishing websites [28]), but to enhance or complement them by leveraging human sensing capacity and experience. More specifically, HaaSS can be used to actively augment existing technical defence mechanisms by combining telemetry generated by user threat detection with threats flagged by technical defence platforms; helping confirm the existence and highlight the extent of the threat, or crucially, for detecting semantic attacks that have been largely undetectable by technical systems.

In this capacity, HaaSS allows for proactive and preemptive detection of semantic attacks by positioning (and empowering) the user as a platform security sensor in order to identify and report suspected attacks in real-time.

To add clarity to the function of HaaSS in the context of semantic attacks and the wider computer security threat space, we propose the following definition:

Human-as-a-Security-Sensor. The paradigm of leveraging the ability of human users to act as sensors that can detect and report information security threats.

Stembert et al. [245] have recently proposed combining a reporting function with blocking and warning of suspicious emails and the provision of educative tips (such as which technical factors indicate malicious intent e.g., URL structure), so as to harness the intelligence of expert and novice users in detecting email phishing attacks in a corporate environment. Initial experimental results of their mock-up have been encouraging for the applicability of the human as a security sensor concept in this context. Another recent example of utilising the human as a sensor security concept was demonstrated by Malisa et al. [303] where the researchers developed an accurate and automated mobile application spoofing detection system by leveraging user visual similarity perception; integrating the human sensing data collected as an integral component of the technical systems detection decision making. To detect semantic attacks with some accuracy, both systems utilise explicitly user expertise and knowledge, but there is no exploration or measurement of what determines the users' performance as security sensors. By establishing such insight, a technical system could highlight the key attributes associated to user threat detection and as a result improve system performance by recognising which threat reports are more reliable.

As we have experimentally demonstrated in chapter 3, the reliability of a users attack reporting (and therefore attack detection efficacy) depends on their activity profile, as defined by characteristics including the amount and type of security training, familiarity with each system, frequency and duration of system access etc. [304, 305]. The profile also serves to define one's predicted susceptibility to semantic attacks. However, before building a system that depends extensively on a particular type of sensor (and the human sensor is no exception), one needs to be able to measure or estimate its overall reliability. In the case of HaaSS, this means expanding upon theoretical observations made under survey or questionnaire conditions (which often limit the ability to produce lasting empirical conclusions about a security system's practical usefulness [306]), by testing the concept in an representative environment.

Here, we take the first steps in empirically evaluating the HaaSS concept for detecting semantic attacks by designing a technical HaaSS framework (subsection 4.1.1), which we then use to develop an applicable prototype system (section 4.2), tested under both laboratory (subsection 4.3.1) and real-world (subsection 4.3.2) conditions.

4.1.1 Framework architecture

To build upon chapter 3 and systematise HaaSS sensors within a practical defence system, we propose a set of three processes which organise users as physical sensors (i.e., the HaaSS Sensor) within a typical cyber security defence architecture. Each discrete process (attack detection, classification and response) represents root system functions as technical components, which consist of a series of modular sub-system components.

Below, each of the processes are described according to domain specific functions in the HaaSS framework. However, they can also be traced to more generic applications of threat detection and resolution observed in technical computer security systems that do not employ HaaSS functions.

- **Process 1 - Detection**. Threat detection is the primary function of the HaaSS sensor and concerns the reporting of semantic attacks that are observed by HaaSS sensors (e.g., human users) on the user-computer interface. Here, threat detection can be an ongoing active or passive process, depending on the context of the HaaSS sensors (e.g., actively searching for threats, or symptomatic exposure based on activity profile). Within the detection process, HaaSS sensor activity is continuously monitored to establish, dynamically, their detection efficacy for semantic attacks.

- **Process 2 - Classification**. In the HaaSS framework, the classification process is where HaaSS threat reporting is actively translated into decision actions for informing and coordinating the implementation of technical security defence rules for mitigating semantic attacks. Therefore, HaaSS sensor report classification forms the focal function of HaaSS defence, as it is the framework component where security response enforcement decision making is made for HaaSS reports. Specifically, threat report classification process decides whether semantic attack reports received by HaaSS sensors are credible attacks or not. However, unlike conventional defence approaches, in HaaSS the primary and initial decision process relies on the scoring of the HaaSS sensor's detection efficacy, rather than the attack data, to determine the reliability of the report being an actual threat. Detection efficacy is used as a filter to automatically respond to HaaSS threat reports, or to forward and prioritise reports for attack data review and classification.

- **Process 3 - Response**. Like any security system, the result of an attack report classification outcome (e.g., attack decision: true/false) is either the deployment of threat mitigation functions or not. In HaaSS, the response is the execution of security enforcing functions and rules (e.g., blocking a website URL, adding an email domain to spam lists, creating a malicious file signature, or sending out user threat notifications) designed to mitigate positively classified HaaSS threat reports. The response process is also responsible for feeding back classification output to enforce continuous learning of correct and incorrect HaaSS detections, with the aim to continuously improve classification accuracy whilst adapting the system to its specific HaaSS sensor-base.

Shown in Figure 4.1 is the detailed HaaSS framework with the system criteria and underpinning functional components. In the next subsections, each of the frameworks system and sub-system components are described as to their role and function. The experimental system settings are included as tested in the prototype technical implementation called *Cogni-Sense*. *Cogni-Sense* is discussed in detail in section 4.2.

Detection system process: functional components

- **User Interface (semantic attack exposure)**. The user interface is an implicit component in the HaaSS framework and the source of HaaSS sensors exposure to semantic attacks. The user interface applies to any computer system, whether served locally, remotely, by cyber or physical means. The taxonomy in chapter 3 describes the various methods of attack distribution on the user interface; which can be used to identify the optimum positioning of the HaaSS sensor to implement reporting mechanisms for the widest coverage.

- **Feature collection**. HaaSS sensor activity on the user-computer interface generates real-time profile data which formulate the on-line, auditable features for computing the HaaSS sensor detection efficacy metric. Here, we refer to on-line, auditable features as features generated as a result of real-time HaaSS activity (e.g., duration of platform access). By comparison, self-efficacy features are collected in real-time, but are not necessarily continuous by nature (e.g., HaaSS sensor self-efficacy). Auditbale feature data is generated through HaaSS sensor interaction with the user interface and is collected by the underpinning HaaSS reporting platform integrated with the user interface. If feasible, feature measurement can be automatic through the platform collecting the feature. However, for specific features such as self-efficacy based data, manual HaaSS sensor input is required. The specific process and algorithms for auditable and self-efficacy feature collection in the framework are described in detail in section 4.1.2.1.

- **Sensor Report (Feature interpolation and attack data)** An attack report is initiated by a HaaSS sensor using a HaaSS report interface when a suspected semantic attack is detected. The HaaSS report interface provides a mechanism for generating an attack report using the existing user interface available to the HaaSS sensor. Once initiated, a sensor report captures, in real-time, the HaaSS sensor feature-set for the specific attack report context (e.g., platform).

The sensor report uses report context to extract the HaaSS sensor feature-set, where raw feature data is interpolated and discretized into a format readable by the HaaSS susceptibility model; this process is described in more detail in section 4.1.2.1. The formatted feature-set is unique to each report, formulated dynamically based on the attack report context and time. The sensor report attaches attack data alongside HaaSS features to be sent for classification and response (e.g., defence). In a HaaSS report, the attack data is built by extracting key threat information automatically from the user interface, as well as optionally allowing for manual data input supplied by the HaaSS sensor. For example, attack data may consist of (but is not limited to) video, images, files, links, interface meta-data, text description (supplied by HaaSS sensor), diagnostic data on the platform serving the user interface. Once sent, sensor reports are delivered to a remote platform for classification in the HaaSS attack detection validation component, where attack data is forwarded to a semantic attack sandbox and HaaSS features are used to compute the sensors detection efficacy for the report.

Classification system process: functional components

- **Compute HaaSS Score *H* (HaaSS features)**. A focal point of the HaaSS framework is the measurement of HaaSS sensor detection efficacy and reliability for semantic attacks reports. The H score forms the primary decision making process within the HaaSS framework and provides a mechanism to distinguish between credible threats for triggering appropriate defence mechanisms. HaaSS report classification is initiated on-demand when a attack report is received by a HaaSS sensor and is computed as a validation measurement of the HaaSS sensor's attack detection efficacy. The validation measure utilises a susceptibility model (using the approach developed in chapter 3) to generate a detection probability metric which we coin as the HaaSS score (H). The H score is primarily used to determine the likelihood of whether a HaaSS report is a credible semantic attack or not, which, depending on the score and classification threshold defined, would result in automatic classification and immediate execution of security enforcing functions for classified attacks. Alternatively, it is used to inform manual report classification of the likeliness of the report being a semantic attack. By default, once computed, each attack report received by a HaaSS sensor is assigned a H score generated based on the reports accompanying HaaSS features. We expand considerably upon the concept of the H Score (H) in section 4.1.2.

- **Report repository (Attack data, HaaSS features and the Semantic attack sandbox)**. Within the HaaSS framework, the report repository stores all received HaaSS reports received by HaaSS sensors and serves as the storage source which supplies attack data and report information for review and classification in a semantic attack sandbox. Conventional computer security sandboxes are designed as safe containers which evaluate heuristically the semantics of untrusted code execution, or the meta-data of computer system files in a secure environment away from the host computer platform to detect attack patterns or anomalous activity. In HaaSS, the semantic attack sandbox is designed to expose behavioural and cosmetic user-interface attributes for analysis, to distinguish between legitimate or malicious intent (as suspected by the HaaSS sensor). As deception vectors (see taxonomy chapter 2) primarily target the user-computer-interface, instead of technical software analysing system behaviour, it is human users who form part of the sandbox architecture by analysing the system behaviour through human sensory (e.g. visual interpretation); such as human users as security operators of the HaaSS system or peer HaaSS sensors. Currently there exist a number of online platforms providing sandbox functionality for email and website phishing, but to a limited extent. Online phishing repository *PhishTank* provides a web interface for reporting and reviewing phishing website and email attacks, supplying a screenshot of the report and for phishing websites the ability to interact with the attack itself via a HTML iframe (providing the phishing website has not been removed from circulation). By comparison, Millersmiles, provides only a simple text scrape of phishing emails for review (see section B.6). Unlike the HaaSS framework, these environments are limited to phishing attacks only, do not integrate with any security systems (where classified attacks result in direct defence measures) and provide no mechanism for report prioritisation based on the reporters detection efficacy profile. In the framework, all HaaSS sensor reports are forwarded immediately to the semantic attack sandbox, where report attack data is presented in a secure container for analysis. Sandbox attack data can consist of a number of formats such as images and video, however to guarantee security and persistence in the sandbox, only static data is supplied instead of linkage to the actual attack source. This ensures the sandbox provides the necessary visual interaction with deception vectors for effective threat classification, without the risk of exploitation. The semantic attack sandbox also provides a mechanism for pro-active defence, where report classifications can be reviewed again if historic automatic or manual classifications turned out to be incorrect.

- **Susceptibility threshold (δ).** The HaaSS score susceptibility threshold controls when the system automatically classifies a HaaSS report as an attack (1), non-attack (0) or whether it defaults to an unclassified state (e.g., NULL). The threshold allows for adjustable control of false positive and false negative report classification, which vary depending on the HaaSS score model accuracy. For a configured threshold, if the upper or lower threshold is met, the HaaSS report is automatically classified (which also serves to automatically set the response label when adding the report to the HaaSS model training and validation dataset). In the case of an unclassified report state (i.e., HaaSS score <upper threshold and >lower threshold), the HaaSS report is marked as unclassified in the semantic attack sandbox for manual classification. Here, the HaaSS score is then utilised as a prioritisation metric for manual classification by a HaaSS report reviewer (e.g., security operations/platform personnel).

Response system process: functional components

- **Rule enforcement (System and user defence).** HaaSS report classification results in an attack decision which is true or false. Attack classification results directly in a response that issues a security enforcement function on a set of pre-configured rules based on the context of the report. This involves implementing a function that would protect against the semantic attack on the system or user and requires the HaaSS system to have a security enforcement module (SEM) that is integrated locally or remotely to external technical security platforms. The rule enforcement process provides traceability of HaaSS sensing and detection through to autonomous and pre-emptive defence measures against semantic attacks, by utilising a report classified as an attack to invoke rule enforcement that results in technical security configuration and execution. For example, in the case of an organisational HaaSS system, for a HaaSS report of a phishing website classified as credible (automatically or manually), the HaaSS system can enforce a rule that sends a configuration setting to the organisations web proxy (using the website URL) to block the phishing website. Undoubtedly, this would require API connectivity or middle-ware to translate the rule to configuration input. Another example, in a wider use case setting might be email distribution containing the details of the phishing website which is sent to all HaaSS subscribers (e.g., home internet users and other HaaSS sensors using the system). For each attack decision response, the HaaSS features and attack decision (true or false) are fed back into the H score training sample for the H score adaptive remodelling function.

- **HaaSS Score adaptive remodelling.** Adaptive remodelling is a continuous feedback mechanism designed to learn the behaviours and profiles of a HaaSS sensor-base by periodically re-training the HaaSS score prediction model and improving its accuracy. The remodelling procedure activates when the overall HaaSS detection distribution within the training data has deviated significantly from the HaaSS score models original training data distribution; distribution refers to the user detection rate which is based on the number of HaaSS report samples with correct or incorrect classification in the most recent baseline training and validation dataset. The task can be periodic (e.g., running once a day, week or month), employing mean deviation as the trigger to invoke remodelling; the calculation takes x as the values of the most recent base-lined training data distribution and the current (at that point in time) distribution where $mean\ deviation = \frac{\sum|x-\mu|}{2}$. If distribution exceeds a specific deviation threshold (with experiment configuration set as 0.5), remodelling is invoked. When remodelling is triggered the HaaSS score model is retrained on the most current HaaSS data sample with any tuning parameters defined i.e., train and test sample split, machine learning algorithm, cross-validation (CV), automatic feature selection. After retraining, the HaaSS score model's predictive performance is reviewed at different class probability thresholds to define the optimum threshold (δ) for minimising false positives or false negatives when computing the HaaSS score and setting the threshold for automatic or manual report classification as defined in the classification process. Adaptive remodelling can be configured to run as an autonomous process, or manually by a HaaSS system administrator.

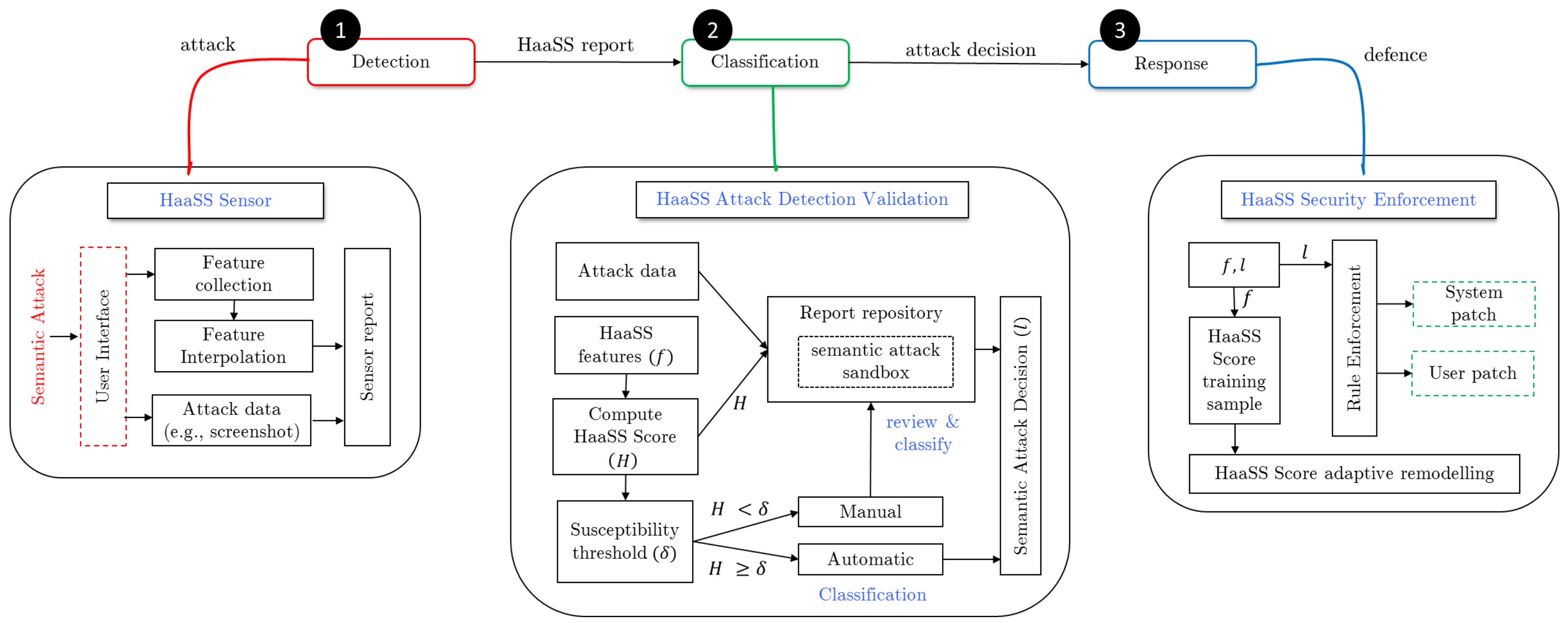


Figure 4.1: Human-as-a-Security-Sensor framework

4.1.2 Predicting Sensor Reliability: The HaaSS Score

The concept of Human-as-a-Security-Sensor places users at the heart of a security system designed to detect deception-based attacks on the user-computer interface, where instead of computer-based security sensors, it is human sensing capacity and experience that is utilised for threat detection.

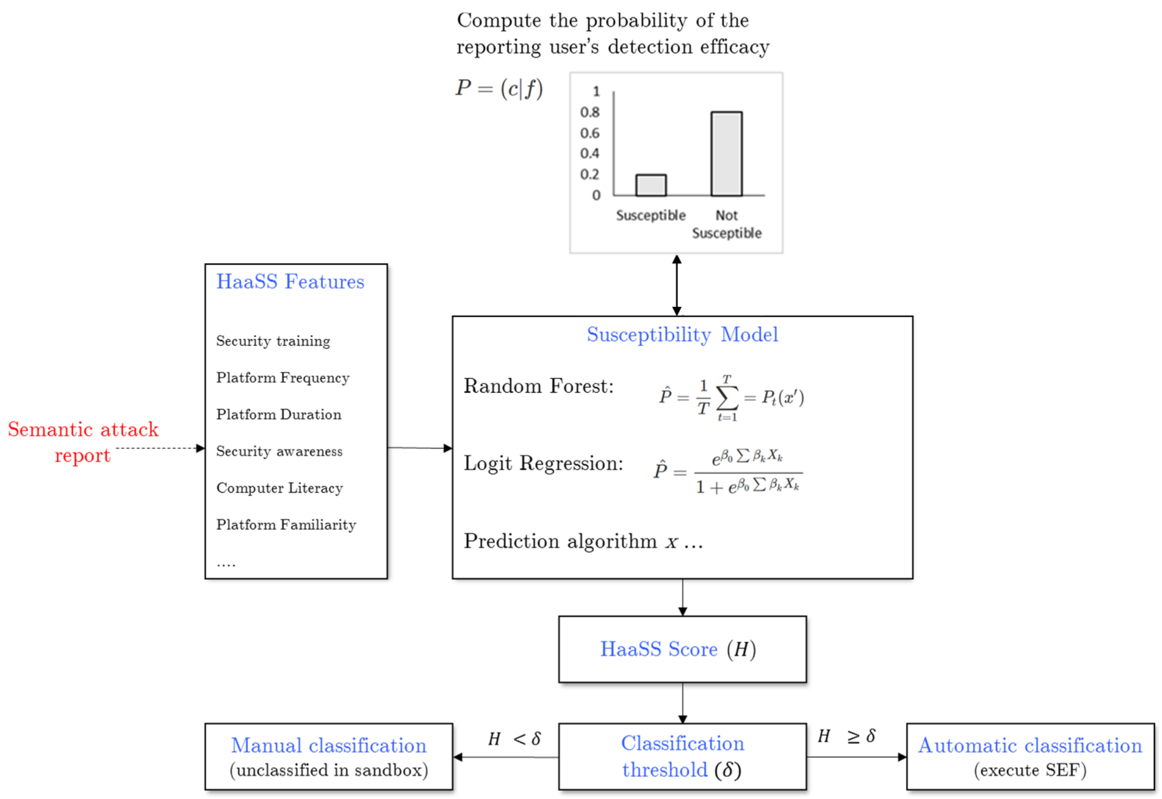
Whilst similarities between computer-based and human-based security sensors can be drawn by a mutual requirement to measure their attack detection accuracy, a major distinction between computer and human sensing (in this case) is that human sensor detection is much more unpredictable by nature and therefore less consistent from one sensor to another. This is because in a technical system, for a specific computer-based security sensor (such as an anti-virus scanner), the same attack detection reliability is shared amongst all sensors (as each sensor is effectively a clone of the other - assuming they are fully updated), whereas in HaaSS, detection reliability is uniquely applicable to each individual HaaSS sensor based on their specific detection efficacy profile. To harness and channel the unpredictability of human sensing and to model humans as physical sensors that form an effective and systematic medium for semantic attack detection, we employ a metric, which we call the H score. The H score represents the trustworthiness of the report as a result of computing a HaaSS sensor's predicted detection efficacy for a specific report context (e.g., attack platform type and sensor's current security training). This is achieved by utilising the susceptibility model developed in chapter 3, to profile HaaSS sensors through a series of key user susceptibility indicators, which are then used to form a dynamic prediction of their expected detection efficacy. The H score is computed by extracting a HaaSS sensor's most recent detection efficacy profile attributes (i.e., their susceptibility indicator predictors) at the current point in time when a semantic attack report is initiated; using these attributes as a series of features to determine the likelihood of whether the attack report is or is not a correctly identified semantic attack.

In the HaaSS framework the H score facilitates three key objectives of HaaSS sensor reporting, (1) immediate classification of HaaSS reports (i.e. attack / not an attack) for executing automatic security enforcing functions, (2) prioritisation of the most trustworthy HaaSS reports (where the H score is too low or high for automatic classification) for manual classification with the aim to minimise the exposure time of vulnerable users by review precedence, and (3) the ability for a HaaSS system to learn and adapt to the changing detection efficacy profiles of its incumbent HaaSS sensor-base through H score accuracy remodelling. In these applications the H score can function in two modes independently or simultaneously. These are classification mode for autonomous, system-initiated security enforcement and prioritisation mode for manual, human-initiated security enforcement. In classification mode, the probability value is evaluated against a defined classification threshold which determines whether automatic report classification will occur and is defined by the sensitivity and specificity requirements of the environment. Depending on the general HaaSS score accuracy, different classification thresholds will result in either higher false positives or false negatives or an optimum minimum for both. For prioritisation mode, the higher the probability, the more trustworthy the report and therefore the higher precedence it is afforded by manual human review and classification.

Whilst the H score is primarily designed as a mechanism for accurate treatment of semantic attack reports that are received from HaaSS sensors, it can also be utilised as monitoring tool for continuous analysis of HaaSS sensor detection "health". For example, in the case where a HaaSS sensor lacks the detection efficacy to identify a specific semantic attack on a specific user-interface (and platform), it is reasonable to assume that a report of this particular threat would never be received from/by this sensor. Conversely, for HaaSS sensors that are over-suspicious and therefore highly sensitive to suspected deception on the user-computer interface, the HaaSS sensor is likely to produce many false positive reports. To identify these weaknesses and perhaps deliver targeted training where needed, periodic review of the passive H scores for HaaSS sensor detection efficacy would be beneficial.

Figure 4.2 provides an overview of the H score function, expanded from the "HaaSS attack detection validation" component which forms the classification process of the HaaSS framework. Within the HaaSS attack detection validation, the function

Figure 4.2: H score prediction and report classification process



Compute the probability of the reporting user's detection efficacy
$P = (c|f)$
1
0.8
0.6
0.4
0.2
0
Susceptible
Not Susceptible
HaaSS Features
Security training
Platform Frequency
Platform Duration
Security awareness
Computer Literacy
Platform Familiarity
....
Semantic attack report
Susceptibility Model
Random Forest: $\hat{P} = \frac{1}{T}\sum_{t=1}^{T} = P_t(x')$
Logit Regression: $\hat{P} = \frac{e^{\beta_0}\sum \beta_k X_k}{1 + e^{\beta_0}\sum \beta_k X_k}$
Prediction algorithm x ...
HaaSS Score (H)
Manual classification (unclassified in sandbox)
$H < \delta$
Classification threshold (δ)
$H \geq \delta$
Automatic classification (execute SEF)

extracts HaaSS sensor features for incoming semantic attack reports (shown is an extract the features selected as part of the statistical modelling in chapter 3), which are fed to the susceptibility prediction model for H score predication. The susceptibility model prediction method is defined by the machine learning algorithm used to train model, for the models trained in this work the prediction formula (which forms the H score probability value) for Logistic Regression and Random Forest are used as per Chapter 3. For the prototype system developed in section 4.2, the Random Forest model was implemented. The H score output is the the probability P of the output class c (susceptible/not susceptible or equally not detected/attack detected) for input HaaSS features f in the report, being $P = (c|f)$, as computed by the susceptibility model algorithm. The H score generated is evaluated in classification mode against a user defined threshold δ to decide whether automatic or manual classification should be performed on the attack report.

Next we describe the framework's procedure for collection and measurement of the HaaSS features which are used to predict the H score.

4.1.2.1 Feature Generation and Measurement

The following section describes how the auditable and self-efficacy predictor features defined in the susceptibility model chapter 3 are generated from user attribute data and measured in order to supply detection efficacy features for the H score computation.

auditable features can be collected in real-time and automatically either on the user's client-side system or remotely through API's that are integrated with the HaaSS platform; this function operates whenever there is user activity that can be monitored. Feature collection is a continuous task that initiates at user log-in and operates passively until the user logs off or shuts down their system. For remote feature collection, computation of auditable efficacy features can be conducted in batch mode - e.g., on demand or in real-time depending on whether all auditable features are sent to the server-side HaaSS database. Real-time computation can be less efficient as platforms forwarding features to the HaaSS database whether on the users machine or in the Internet must continuously monitor user activity from different locations and sending multiple data streams over the network and therefore introduce scalability problems. Similarly, local feature collection on the users systems can be computed in real-time or batch mode (e.g., on demand - which is the point at which a user makes a report) this is achieved by either continuously checking the activity history to generate features values or periodically extracting the recorded history of user activity on the platform associated to the suspected attack. In the case of the former the processing requirements of continuous feature generation are less efficient, whereas batch feature generation still allows for real-time feature selection when a user sends an attack report. If a user were to be exploited both approaches equally facilitate the ability to forensically review user activity to generate a "snapshot" in time feature-set that would identify their susceptibility profile at the time of exploitation. In the case of the former, feature values would already have been generated whereas for batch mode, the features would require generation from the existing activity history that has been recorded. The process of feature generation, which applies to both local or remote collection, is presented visually in Figure 4.3.

4.1.2.2 Auditable Features

In the following section we describe the technical process of auditable feature generation using local batch-mode collection only. In the HaaSS framework, it is assumed that a continuous monitoring mechanism collects raw data from each measurable user-interface that the HaaSS sensor accesses and interacts with (e.g., web platform, application, file, device etc.). On different systems and for a range of disparate user-interfaces, the method for collecting raw access data from user-interaction may vary and therefore the exact technique for raw access collection is not prescribed within the framework. We do however develop a specific collection technique within the prototype HaaSS platform *Cogni-Sense* in section 4.2, to demonstrate the viability of automatic access activity collection and to facilitate feature generation and interpolation using the following algorithms. Here, the algorithms for feature generation and interpolation are designed to function on any source of raw access data, assuming that the data has an an accurate time-stamp.

- **Frequency of access (FR)**. The feature measurement algorithm for Frequency of Access (FR) is designed as a generic algorithm which applies to a monitored user interface (which refers to a measurable computer platform or application interface). By default, frequency of access is measured a using a 31 day period to represent a month of activity (rounding upwards based on a 30.5 day leap year month), which defines the threshold for moving between frequencies. For example, for user platform activity to be classified with a frequency of "daily" access (i.e., there is a maximum of one day elapsed between platform usage), this daily activity must be continuous for a period of 31 days, without more than one day difference between accesses. The same rules apply for each frequency category, for example "weekly" access would require that there is no more than 7 days between each indexed access date; the same rule is applied for the remainder of the frequencies categories. However, different frequency transition thresholds can be applied when increasing or decreasing the frequency granularity scale. When a HaaSS sensor attack report is initiated the related user-interface access activity specifically related to the report (e.g., platform, application, file ... etc.), are indexed in an array of dates $\cdot v$ from the current recorded access up to 31 days from the latest measurement taken at the time of a report. Here, it is assumed that the collection mechanism records each individual access of a platform user-interface continuously and therefore may consist of multiple accesses on the same day; for the frequency algorithm only the first recorded platform access is indexed for each specific day. For the length of the array, beginning with the first array index i=0, the elapsed period E_i is measured in absolute days between the reference date time v_i (which for the first index in the array is the date the report is made) and the next index v_j, where $j = i + 1$ (which is the recorded access time-stamp for when the platform interface was last accessed since the report was made) is computed. v_i and v_j are then both iterated to the next index in the array, where the absolute value in days between each index value is computed again; this process is repeated for each index for the length of array.

Once each of the absolute elapsed days between each platform interface access is completed, we are left with a frequency list $(E_1, E_2, ..., E_n)$ which is then iterated through with a case function (e.g., IF, ELSE) to calculate the frequency of access (e.g., daily, weekly etc.). In the case where there is no recorded access activity prior to the HaaSS sensor attack report, then the frequency of access is “never” (which is to say no recorded access before this point in time). Algorithm 1 in Appendix 2, provides a step-by-step description of the feature generation and interpolation, using the default 31 day frequency threshold shift.

- **Duration of access (DR)**. The feature measurement equation for Duration of access (DR) is designed as a generic algorithm which applies to a monitored user interface (which refers to a measurable computer platform or application interface). Here the same assumption for the access activity collection mechanism is employed, where it is continuous for each individual access on a specific day and therefore may consist of multiple accesses on the same day. With this in mind, for each access in a 31 day period, each chronological time-stamp is ordered to represent a start and stop elapsed time in seconds for each day. The elapsed time in seconds for each individual access per day, at the time of a HaaSS attack report, is then indexed in an array $\cdot v = (v_{1_1}, v_{1_2}, ..., v_{n_j})$. By iterating through all the access instances recorded for a specific day i_j up to the last measurement taken before the HaaSS sensor attack report, beginning at the first observation i, the duration of access is summed in seconds and is computed as the total duration of platform access in seconds for each recorded day sec_i. The result is a list of duration in seconds for each day which is then divided by the the total number of days $D = len(sec_i)$ measured, to compute an overall average duration of access T_{sec} for the reports platform interface.

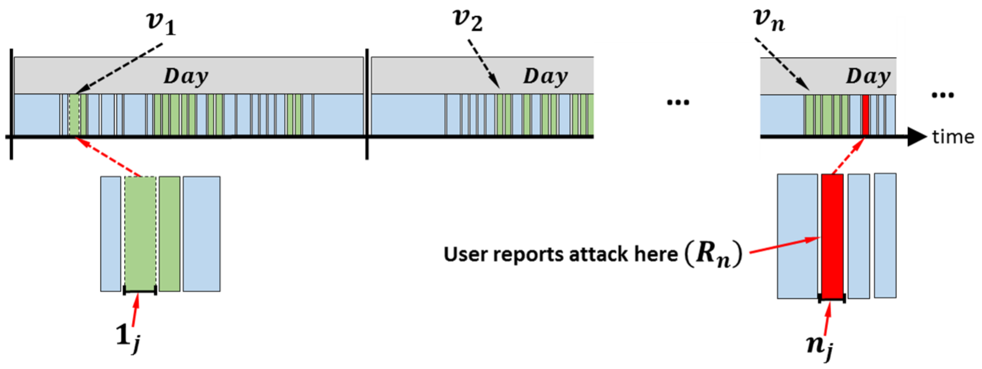
  Finally, the average total duration in seconds T_{sec} is converted to an ordinal integer scale using a case function to form the input value for the prediction models duration feature (DR). As per frequency of access, in the case function different duration thresholds can be applied when increasing or decreasing the duration granularity scale as required. Algorithm 2 in Appendix 2 provides a step-by-step description of the feature measurement algorithm.

- **Computer Security Training (S1T, S2T, S3T, S1_3, S2_2, S2_3, S3_1, S3_2, ST2)**. Computer security training is measured based on its recency from the date it was last received, based on the training type (S1 - formal, S2 - work-based, S3 - self-study), delivery (S1 - coursework, S2 - video and games, S3 - video and websites) and platform type (e.g., email, social media, e-commerce etc.). Therefore, the elapsed time of computer security training increases symptomatically until it is reset/reduced when training is next received. However, like frequency and duration its recency is in relation to the date and time that a HaaSS sensor report is made. Whilst monitoring activity of security training could utilise the frequency and duration algorithms for similar measurement criteria, here we assume that security training is recorded and validated prior to determining the time since it has been received, as well as the format, delivery method and platform. For example, on taking security training the system delivering the training will provide a record of the training for the HaaSS sensor(e.g., on work, formal education, or self-study subscribed system). This record of training will be supplied to the HaaSS system to update their HaaSS sensor security training features. Therefore, security training is not locally measured and triggered at the point of attack report, rather on receiving a report to the HaaSS system, before H score prediction, the HaaSS sensor's detection efficacy features are queried on the HaaSS system for their security training record. In this respect, security training is updated by external systems or users themselves manually after completing training, and symptomatically becomes less recent as time elapses since the record was updated. For manual updates of security training, validation mechanisms would be required such as entering completion codes (issued by the delivery platform) which is a common form of validating the attendance of training and assessment.

In chapter 3, each type of security training and the elapsed time since it was received (i.e., S1:3T) was measured separately from the delivery method related to that type (e.g., formal - coursework, work - games, self-study - websites and so on), as was elasped time since security training for particular platform types (e.g., email, social media etc.). However, when measuring security training periodically it assumed all three will be entered as a single record as representative of a training received. So, whilst they are implemented and computed as separate features in the susceptibility model for generating the H score, it is assumed that there will be direct, time-based correlations between them that were not possible to accurately measure in the experiments conducted in chapter 3.

In Figure 4.3, a visual representation of a continuous measurement time line is shown, where the frequency and duration feature algorithms applies interpolation and discretisation to raw recorded activity data from a HaaSS sensor.

Figure 4.3: Real-time collection of auditable HaaSS sensor features (FR and DR), variable green periods represent variable durations of access to the same platform and blue periods represent variable durations of access to other platforms over the period of one day. Access activity for frequency measurements is made by recording the first observed access instance for a day on a specific platform. Features are computed at the point in time when a user reports a suspected semantic attack for the specific platform in question, where R_n represents the nth report for jth platform interface accessed by the HaaSS sensor.



For each of the auditable feature collection algorithms, the discrete measurement scale configuration can be adjusted to increase or decrease the granularity of activity monitoring. For example, frequency measured on a five point ordinal scale (daily, weekly, monthly, less than monthly, never) can be extended to measure a wider access frequency range (daily, every two days, weekly, every two weeks, every month, every two months etc.), or reduced to {daily, weekly, less than weekly}. The same applies to duration of access. Depending on the range and granularity of the discrete scale, this would either increase the required learning time to move between frequency threshold e.g., 31 days for a five point scale, or decrease e.g., seven days for a three point scale such as daily, weekly, less than weekly. The greater the range of measurement, the more accurate the picture of HaaSS sensor activity. However, one constraint for increasing beyond a default scale that has been used to training the susceptibility model is the need for retraining the model to utilise the new feature scale effectively. This would requite a prolonged period of activity and associated feature collection from the HaaSS sensor-base, over an amount of time sufficient to produce suitable sample data for training and testing of a new susceptibility model based on a new feature scale.

4.1.2.3 Self-Efficacy Features

For self-efficacy we refer to the generation and measurement of features that are supplied as part of self-assessment by HaaSS features in real-time (e.g., at the time of a HaaSS report), or over elapsed time as part of their HaaSS sensor profile. Whilst we include features that were selected during the modelling process in chapter 3, the process for measuring self-efficacy applies generically for future features within the HaaSS framework.

- **Platform Provider Familiarity (FA1)**. Each time a user initiates an attack report, the user is required to provide their self-assessed familiarity for the platform interface they are reporting an attack on (as previously mentioned here we refer to the original scale defined in the susceptibility model developed in chapter 3). If practically available (e.g., in a fully integrated production HaaSS system), FR and DR features can be used to provide estimated validation of the familiarity reported by the user to define accepted correlation threshold's between the self-assessed familiarity and the actual frequency and duration in which the user accesses the platform; downgrading the familiarity value if this threshold is not satisfied. This mechanism can prevent users claiming familiarity with a system they do not use often and as result helps to preserve the integrity of the familiarity feature. The familiarity feature is supplied as a report meta-data input within the HaaSS sensor attack report interface.

- **Computer Literacy (CL)**. When a user updates their security training automatically through a compatible platform, or manually through an online form, they are requested to enter their self-assessed general computer literacy; using the original scale defined in the susceptibility model developed in chapter 3. If practically available (e.g., in a fully integrated production HaaSS system), the computer literacy metric can be compared against a user's recorded and validated computer usage, platform training and qualifications to develop an acceptable correlation threshold between between these attributes and the users self-assessed computer literacy. As demonstrated in experiment 2 in chapter 3, platform-oriented features for frequency and duration of access can be used to accurately predict a user's security awareness and computer literacy self-efficacy score. This mechanism can prevent users claiming a level of computer literacy that they are unlikely to have attained and as result helps to preserve the integrity of the computer literacy feature.
- **Security Awareness (SA)**. When a user updates their security training automatically through a compatible platform, or manually through an online form, they are requested to enter their self-assessed general computer security awareness; using the original scale defined in the susceptibility model developed in chapter 3. If practically available (e.g., in a fully integrated production HaaSS system), the security awareness metric can be compared against a user's recorded and validated computer security training, qualifications, correct HaaSS reports and detection of emulated attacks (e.g., in an embedded security training tool) to develop an acceptable correlation threshold between these attributes and the users self-assessed security awareness. As experiment 2 in chapter 3 demonstrated, features of different types of security training can be used to accurately predict a user's security awareness self-efficacy score. As with familiarity and computer literacy features, this mechanism can prevent users claiming a level of computer security awareness which they are unlikely

to have attained and as a result helps to preserve the integrity of the security awareness feature.

4.2 Cogni-Sense: a prototype HaaSS platform

Employing the HaaSS framework as a technical design blueprint, we have developed a prototype HaaSS platform called *Cogni-Sense*. The prototype's technical architecture is shown in Figure 4.4, where each of the coloured boxes within the architecture refer to a component's functional role within the HaaSS framework's defence system processes in Figure 4.1. By combining each of the system processes functional components into a real technical system, the development of *Cogni-Sense* provides a HaaSS sensor with a practical facility to report suspected semantic attacks and a platform in which to evaluate the concept of HaaSS for semantic attack detection. In Table 4.1 the technical components within *Cogni-Sense* are summarised, with direct traceability to the HaaSS framework functinal components, by technical integration, platform configuration and a description of their functionality. The development of *Cogni-Sense* in this work demonstrates the technical feasibility of the HaaSS framework for building a real-world system around computer users, utilising the medium of human detection as a physical threat sensor for semantic attacks.

The *Cogni-Sense* architecture consists of four key high-level components, (1) the HaaSS sensor detection platform (i.e., *Cogni-Sense* app), (2) a centralised cloud-based platform for classification and security response, (3) a security operations centre platform (e.g., web browser) that is used to access the cloud platform and (4) a security enforcement module (SEM) for the rule enforcement response process which provides integration between the cloud platform and external security platforms. Each of the components are described below:

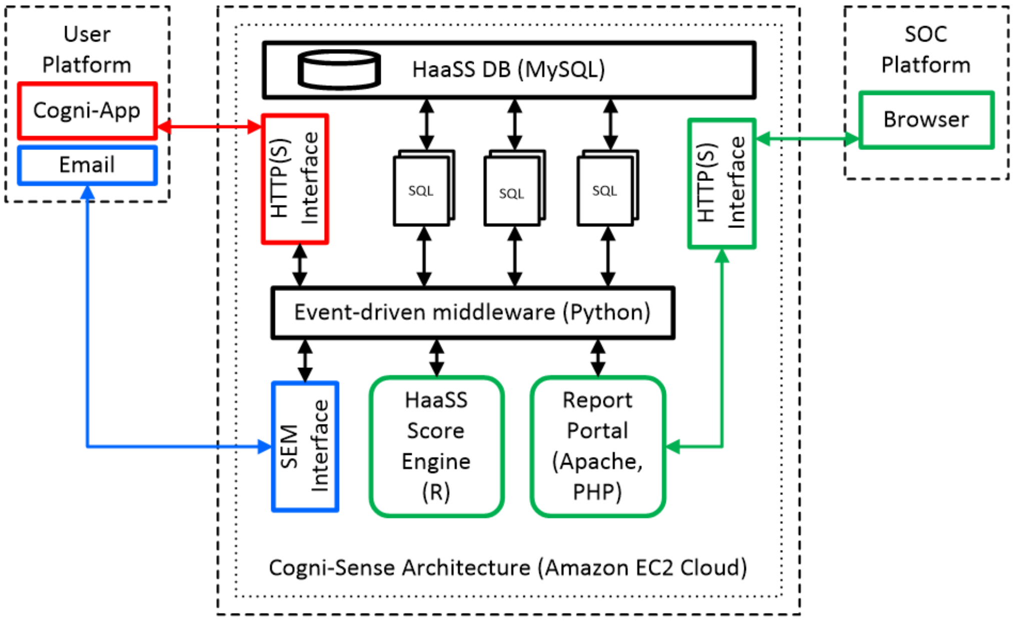
- ***Cogni-Sense* app**. The HaaSS sensor application is a multi-process python application that runs locally on the HaaSS sensor host device OS (in the case it was programmed for Windows OS). The local host OS was selected as the platform in which to develop the feature data collection and attack reporting

Table 4.1: *Cogni-Sense* HaaSS platform configuration and alignment to the HaaSS framework architecture

Framework		*Cogni-Sense*			
Process	**Component**	**Integration**	**Configuration**	**Component**	**Description**
Detection(1)	User interface	Local host OS	Windows	HaaSS sensor device	User-computer interfaces accessible by device andsystem platform hosting the HaaSS Sensor reporting interface for attack detection (e.g., *Cogni-Sense* app).
	Feature collection	Self-efficacy auditable	Manual Automatic*	*Cogni-Sense* and web form *Cogni-Sense* app and web form	Thread on python app continuously monitoring FR and DR of platform interface activity and web form hosted on Cogn-sense web platform for computer security and self-efficacy updates.
	Sensor Report	Host OS Interface Data	Windows Python app Screenshot Text description Deception Vector	*Cogni-Sense* app	Thread on python app creating system tray icon which launches sensor report window when clicked. Captures screenshot of user-interface and manual input from HaaSS sensor.
Classification (2)	Compute H score	H score Metric Mode	Random Forest Accuracy Class probability	H score engine (R)	R machine learning engine with Random Forest H Score susceptibility model configured to output class probability predictions for HaaSS attack report classification and prioritisation
	Report repository	Repository Sandbox	MySQL PHP, Javascript (Apache)	Report Portal	Repository consists of a MySQL database storing HaaSS profiles and HaaSS attack report data. The sandbox on the report portal consists of a PHP and Javascript middleware which aggregates report meta-data alongside an expandable image-frame of the user-interface for attack analysis
	Susceptibility threshold	Upper Threshold Lower threshold	0.85* 0.1*	Python middleware	Event-driven python middleware scripting triggered by automatic or manual classification events exected on HaaSS attack reports
Response (3)	Rule enforcement	Trigger Rule	Attack classification Attack email alert	Python middleware SEM interface (SMTP)	Security enforcement rule set to respond across python interface to SMTP executing an semantic attack alert and awareness e-mail HaaSS sensors (and subscribers).
	HaaSS Sccore adaptive remodelling	Distribution Dev. trigger Train split Test split CV Feature selection ML algorithm ML Tuning	0.59* 0.05 0.8 0.2 10-Fold RFE RF 500 trees Mtry=4	H score engine (R) and python middleware	Event-driven python middleware scripting triggered by periodic deviation calculations on the response distribution of HaaSS reporting against ground truth labels (i.e. classification). Executes a re-modelling procedure within the H engine on the HaaSS training dataset and replaces the existing model if the validation accuracy increases.

*Configuration based on experiment results in chapter 3

Figure 4.4: High-level overview of *Cogni-Sense* technical architecture



interface as it provides the widest coverage of user-computer interfaces. For example, were the *Cogni-Sense* app developed as browser extension it would have only been able to report attacks and collect access activity from the browser interface, instead of wide range or other interfaces such as local applications, removeable media, cyber-physical interfaces (e.g., NFC) etc. Therefore, the host OS provided the largest user-computer interface coverage available for the HaaSS sensor interface application.

The first process is the HaaSS platform interface access activity monitoring for raw feature data collection. Whilst a number of mechanisms could have been used to extract platform interface identity, through application programmable interfaces (APIs) or graphical user interface libraries provided by the host OS or third-party applications, for simplicity we employ the PYWIN32 library to hook into the Windows WIN32 system for reading the text of applications windows in foreground (e.g., focused) of the user-computer interface which provides platform identity data that represents the identity of the user interface. This approach is less accurate and robust than validating directly the platform interface through an API, but is much less resource-intensive and proved suitable for the prototype implementation to record raw data used to create the H score auditable feature-set. In a production system, other programming platforms such as C++ (instead of Python) may prove more suitable for raw feature data collection due to their accessibility to lower-level interface functions in the host platform which benefit granular and accurate activity monitoring. For example, the excellent employee monitoring software *ActivTrak* [307] provides platform-specific implementations for Windows and MAC OSX, using C++, to measure accurately user activity. However, in the case of this thesis such development would take extensive development time and would be expected to form part of a production monitoring system, which is outside the scope of this project. The raw data collection in the *Cogni-Sense* app for the activity monitoring process is shown visually in Figure 4.5. Platform interface recognition in the activity recording is performed by matching the window text in the foreground, using regular expressions against a known list of platforms in a local SQLITE database, where platforms were not in this list they were recorded as unknown with a text-based watermark which allows for aggregated access and measurement of this specific unknown platform. Using this approach we only record specific platform access as required for the experiments in this project, preserving the privacy of participants. However, in a live system the SQLITE database can be updated with new platforms and interfaces in a modular fashion in the same approach used to download a white-list to a spam filter, web proxy or website category classifier.

The second process is the HaaSS sensor semantic attack reporting interface, which runs as a process in the system tray process as an eye icon, which when clicked, spawns a reporting window identifying the platform interface (as well as generating the local audi-table HaaSS features based on the platform context). Using the reporting window, the HaaSS sensor enters their platform interface familiarity self-efficacy feature, as well as attack meta-data such as suspected attack vector and general report-related information. A button is provided to send

Figure 4.5: Platform usage meta-data collected by user activity monitor agent

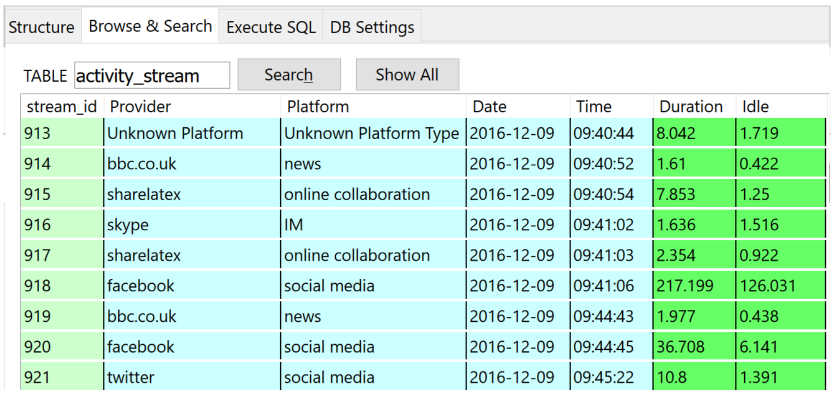
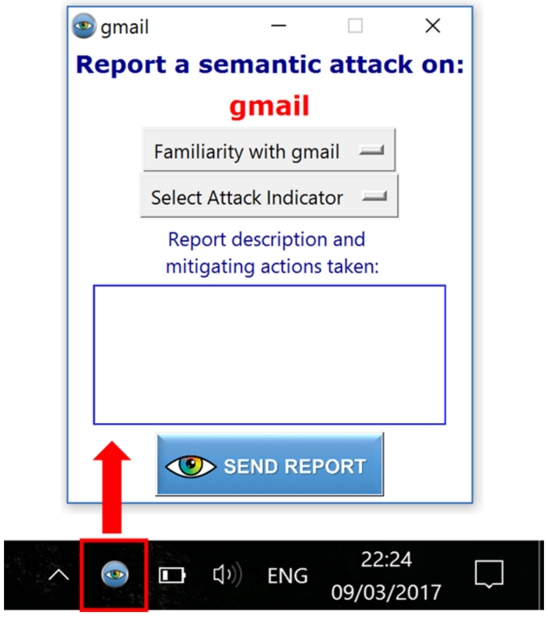


TABLE activity_stream

stream_id	Provider	Platform	Date	Time	Duration	Idle
913	Unknown Platform	Unknown Platform Type	2016-12-09	09:40:44	8.042	1.719
914	bbc.co.uk	news	2016-12-09	09:40:52	1.61	0.422
915	sharelatex	online collaboration	2016-12-09	09:40:54	7.853	1.25
916	skype	IM	2016-12-09	09:41:02	1.636	1.516
917	sharelatex	online collaboration	2016-12-09	09:41:03	2.354	0.922
918	facebook	social media	2016-12-09	09:41:06	217.199	126.031
919	bbc.co.uk	news	2016-12-09	09:44:43	1.977	0.438
920	facebook	social media	2016-12-09	09:44:45	36.708	6.141
921	twitter	social media	2016-12-09	09:45:22	10.8	1.391

the report, which when clicked takes a screenshot of the user-computer interface, gathers the HaaSS features and meta-data and sends the report via HTTP to the cloud-platform, where classification and security response is applied to the HaaSS report. In the case of the HaaSS features, the feature generation algorithms described in the HaaSS framework in section 4.1.2.2 are programmed directly into *Cogni-Sense* app (see Appendix 2 1 and 2 for pseudocode). The HaaSS attack reporting interface is shown in Figure 4.6.

Figure 4.6: The *Cogni-Sense* HaaSS reporting app icon running in system tray. When clicked, a reporting window is opened with the detected platform and report information



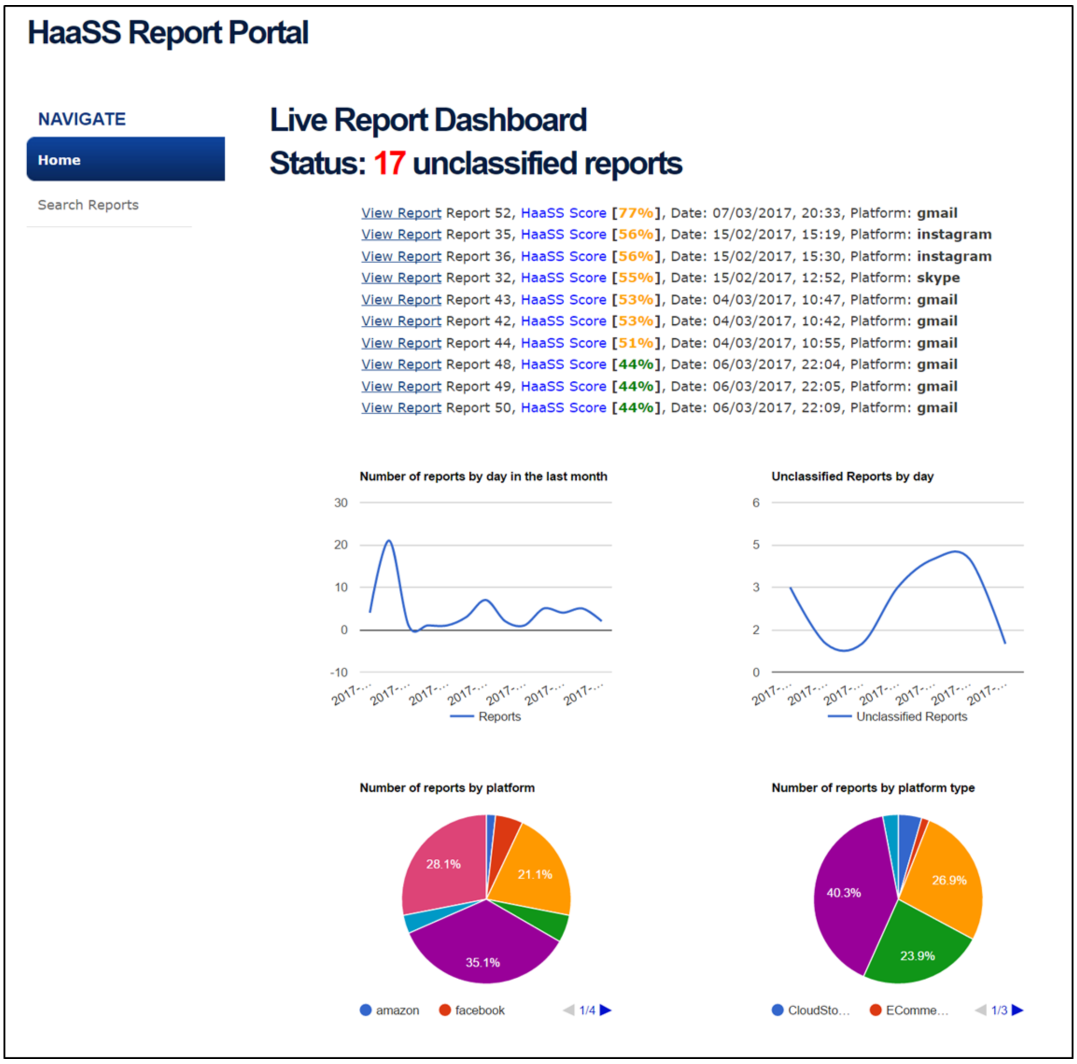
- **Cloud-plaform**.

  The cloud component of *Cogni-Sense* was implemented within Amazon Web services, on a Ubuntu Linux 14.04.3 virtual machine, with 1 GB RAM, a single-core 2.4 GHz Intel Xeon process with 30 GB of storage. Within the *Cogni-Sense* cloud-platform, the HaaSS processes of detection, classification and security response are coordinated between Python middle-ware which provides integration and communication between different components of the system, such as a MySQL database which stores all HaaSS profile and report data, an R-based Random Forest H score engine (i.e., the susceptibility model), an Apache, PHP and JavaScript web-server which hosts the HaaSS report portal and sandbox, as well as interfaces to external security platform connectivity such as SMTP e-mail integration for confirmed semantic attack alerting and awareness training.

  When a HaaSS attack report is received via the *Cogni-Sense* app, the report is stored immediately in the report repository (MySQL database), where the H score is then computed in the H engine and depending on the score and configured susceptibility threshold, is automatically classified or listed by H score priority on the report portal live feed dashboard. The report portal dashboard also includes analytics such as number of outstanding unclassified reports, the frequency of reports received, as well as the different platforms and platform types reported in attacks; which provides indications of where attacks are most concentrated or being targeted. The *Cogni-Sense* report portal dashboard is shown in Figure 4.7.

  For generating report H scores, *Cogni-Sense* utilises the Random Forest susceptibility model which has been developed and trained from experiments in [304]; integrating this model directly into the technical system for HaaSS attack report classification and prioritisation. The model can operate in two distinct prediction value modes, strict classification (0,1) or class probability (0:1.0) as an output criteria. However, strict classification is sensitive to false positive and

Figure 4.7: Example of the *Cogni-Sense* portal live report feed with predicted H score for HaaSS reports



HaaSS Report Portal

NAVIGATE

Home

Search Reports

Live Report Dashboard

Status: 17 unclassified reports

View Report Report 52, HaaSS Score [77%], Date: 07/03/2017, 20:33, Platform: **gmail**
View Report Report 35, HaaSS Score [56%], Date: 15/02/2017, 15:19, Platform: **instagram**
View Report Report 36, HaaSS Score [56%], Date: 15/02/2017, 15:30, Platform: **instagram**
View Report Report 32, HaaSS Score [55%], Date: 15/02/2017, 12:52, Platform: **skype**
View Report Report 43, HaaSS Score [53%], Date: 04/03/2017, 10:47, Platform: **gmail**
View Report Report 42, HaaSS Score [53%], Date: 04/03/2017, 10:42, Platform: **gmail**
View Report Report 44, HaaSS Score [51%], Date: 04/03/2017, 10:55, Platform: **gmail**
View Report Report 48, HaaSS Score [44%], Date: 06/03/2017, 22:04, Platform: **gmail**
View Report Report 49, HaaSS Score [44%], Date: 06/03/2017, 22:05, Platform: **gmail**
View Report Report 50, HaaSS Score [44%], Date: 06/03/2017, 22:09, Platform: **gmail**

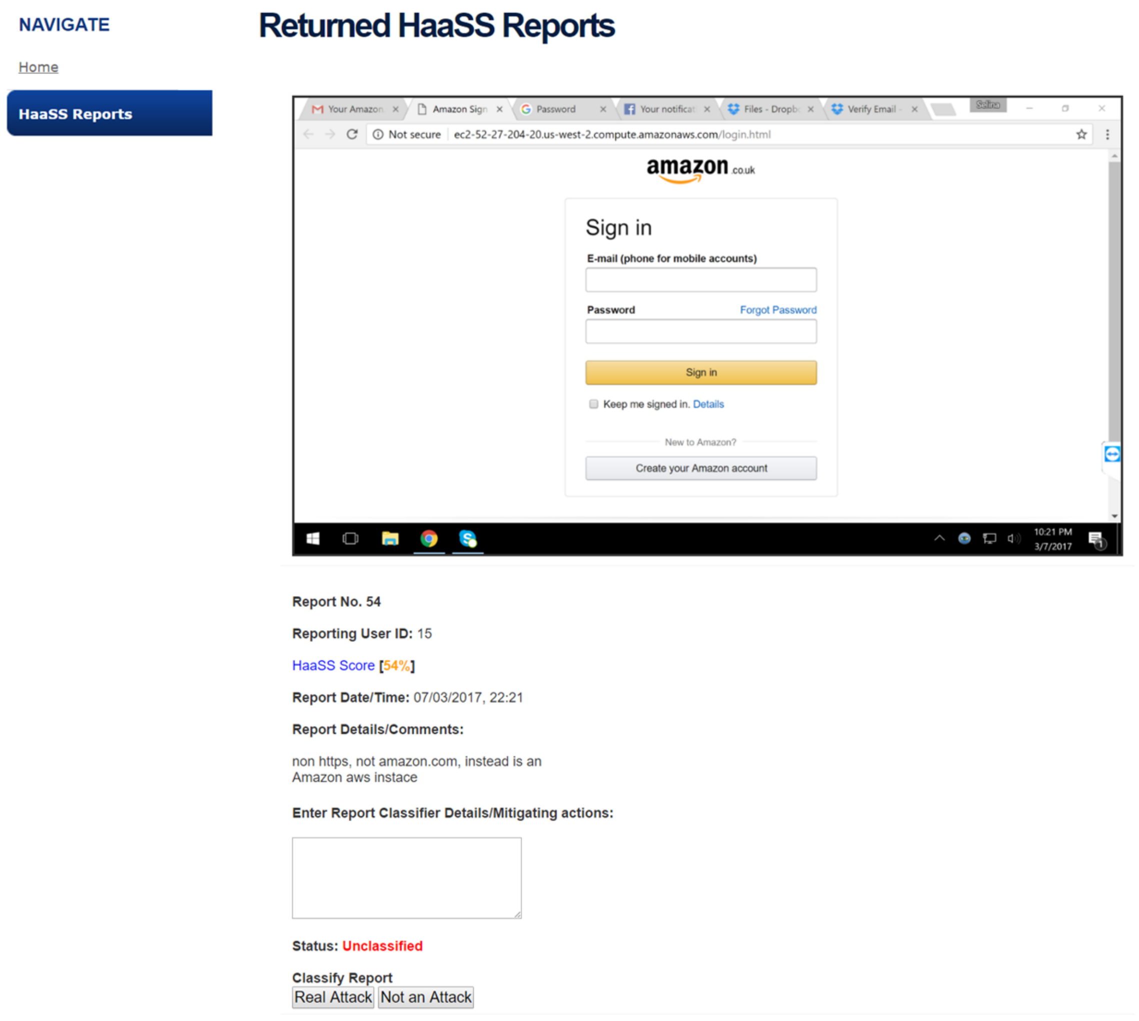
Number of reports by day in the last month
30
20
10
0
-10
2017-...
Reports
Unclassified Reports by day
6
5
3
2
0
2017-...
Unclassified Reports
Number of reports by platform
28.1%
21.1%
35.1%
amazon
facebook
1/4
Number of reports by platform type
40.3%
26.9%
23.9%
CloudSto...
EComme...
1/3

false negative output, which in practice would result in the discarding of accurate reports or nugatory time spent reviewing non-attack reports. Therefore, as per the HaaSS framework H score, *Cogni-Sense* utilises a class probability output mode for prediction of user detection efficacy, where both automatic classification may be used by defining pre-defined responses to different probability thresholds, and manual classification based on report trust through prioritisation of the probability output.

- **SOC platform**. The SOC platforms refers to the web browsing platform used to interface with the *Cogni-Sense* cloud-based platform, where the live feed dashboard (e.g., report portal) and sandbox are located. The SOC platform is designed to be a web-browser which is compatible with Javascript for interfacing with these features. The report portal live feed is the initial screen presented to SOC engineers (or peer HaaSS sensors) who manually review and classify semantic attack reports, where on selecting a report for manual review the image-container sandbox is opened for the reviewer. The sandbox provides an expandable image of the HaaSS attack report, the H score, report meta-data and a classification button (attack/not an attack) for report review. On selecting the report classification the rule enforcement (response process) is triggered. The sandbox is shown in Figure 4.8, containing an example HaaSS attack report received by a HaaSS sensor in the experiment conducted in case study one (section 4.3.1). An example of confirmed "semantic attack" classification for the HaaSS report in the sandbox is shown in Figure 4.8.

- **Security enforcement module (SEM)**. The SEM is configured as Python middleware which translates attack report classification into configuration parameters for security platforms that are integrated with the HaaSS system. For *Cogni-Sense*, integration with an SMTP server has been configured to automatically issue confirmed semantic attack alerts via e-mail to HaaSS-sensors. Given the modularity provided by the use of python middleware, in future iterations of *Cogni-Sense*, configuration rules issued via APIs to security products such as web proxies, anti-virus, firewalls etc., could easily be developed - however this is outside the scope of this project. For the confirmed attack report classification in Figure 4.8, SEM semantic attack e-mail alert response rule is shown in Figure 4.9.

Figure 4.8: Attack 3.2 (Amazon phishing website) *Cogni-Sense* portal report screenshot for HaaSS report by participant H5



Report No. 54

Reporting User ID: 15

HaaSS Score [54%]

Report Date/Time: 07/03/2017, 22:21

Report Details/Comments:

non https, not amazon.com, instead is an
Amazon aws instace

Enter Report Classifier Details/Mitigating actions:

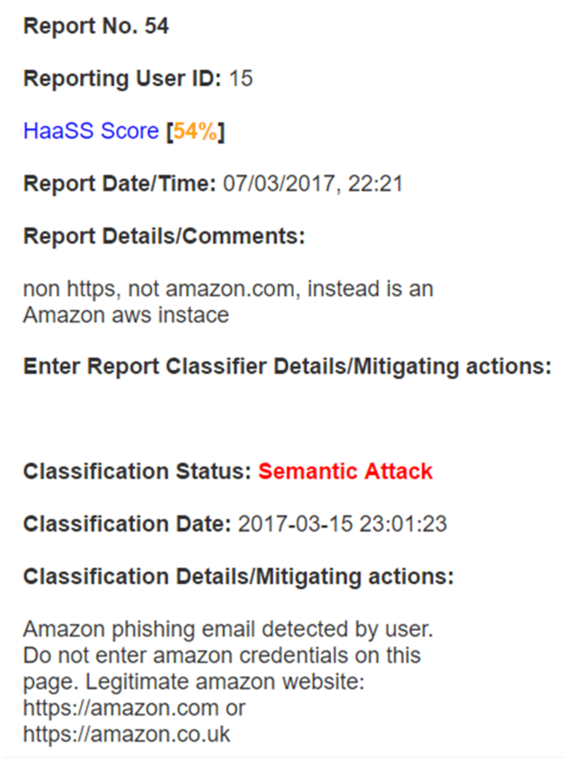
Status: **Unclassified**

Classify Report
Real Attack | Not an Attack

In Figure 4.11, we provide a flowchart for the *Cogni-Sense* semantic attack detection, classification and response process, which is described both systematically and visually to highlight the related platform interfaces made available to HaaSS sensors and semantic attack reviewers on the cloud-platform.

In the next section, using the prototype HaaSS platform *Cogni-Sense*, we conduct two case study experiments to evaluate the viability of the overall HaaSS concept for semantic attack detection and its advantages over current and existing technical defence methods.

Figure 4.9: Example of classified *Cogni-Sense* report, on selecting the option "Real Attack", the dataset for training the H score prediction is updated with the reporting users feature-set and classification decision as the training label. The Security Enforcement Module (SEM) then carries out any configured rules as part of the attack classification e.g., adding the report to user awareness training, sending out a threat alert email or adding the file name to a proxy gateway blacklist.



Report No. 54

Reporting User ID: 15

HaaSS Score **[**54%**]**

Report Date/Time: 07/03/2017, 22:21

Report Details/Comments:

non https, not amazon.com, instead is an
Amazon aws instace

Enter Report Classifier Details/Mitigating actions:

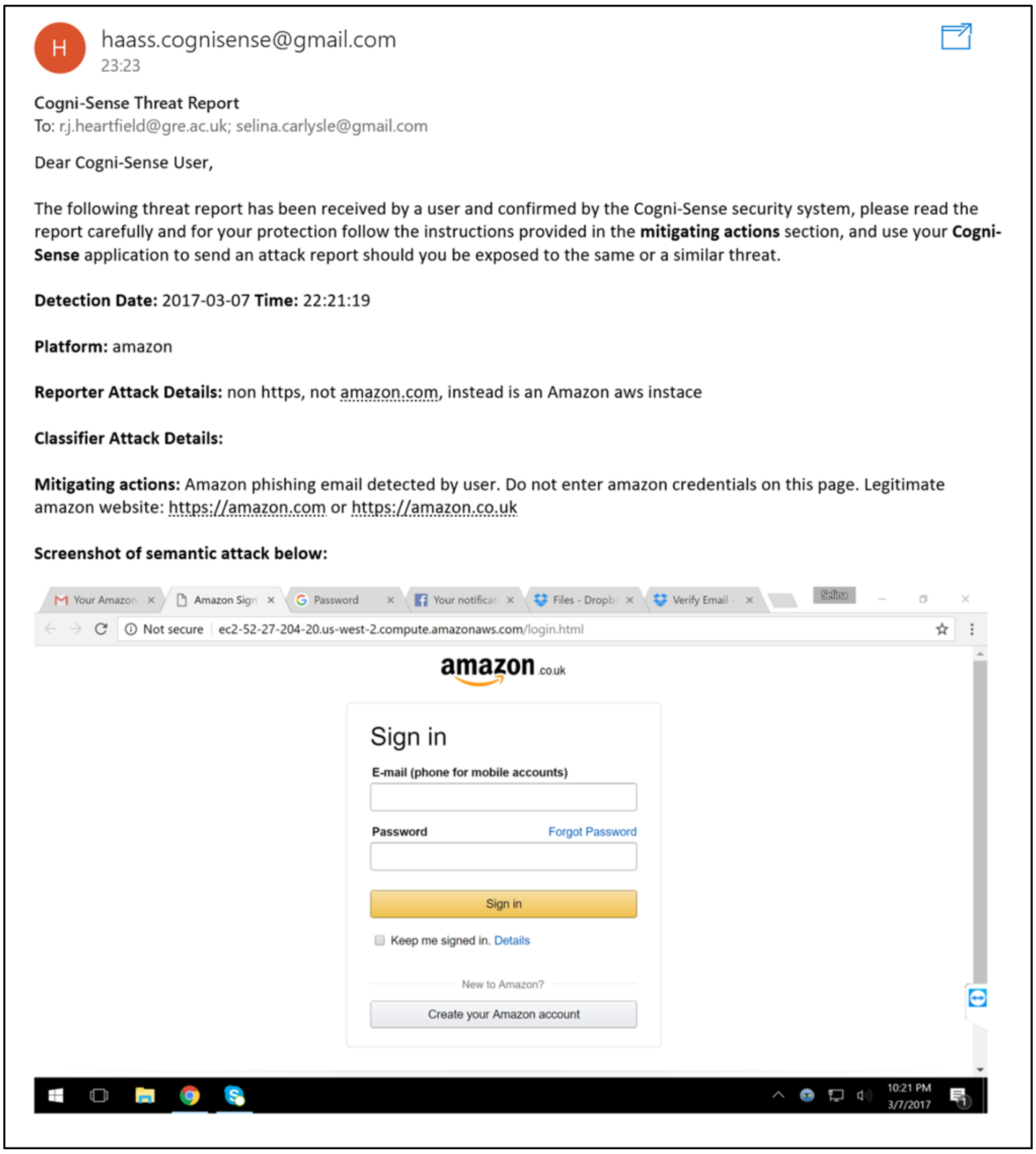
Classification Status: **Semantic Attack**

Classification Date: 2017-03-15 23:01:23

Classification Details/Mitigating actions:

Amazon phishing email detected by user.
Do not enter amazon credentials on this
page. Legitimate amazon website:
https://amazon.com or
https://amazon.co.uk

Figure 4.10: *Cogni-Sense* HaaSS report attack classification triggering SEM module rule: attack awareness email security enforcing function rule



haass.cognisense@gmail.com
23:23

Cogni-Sense Threat Report
To: r.j.heartfield@gre.ac.uk; selina.carlysle@gmail.com

Dear Cogni-Sense User,

The following threat report has been received by a user and confirmed by the Cogni-Sense security system, please read the report carefully and for your protection follow the instructions provided in the **mitigating actions** section, and use your **Cogni-Sense** application to send an attack report should you be exposed to the same or a similar threat.

Detection Date: 2017-03-07 **Time:** 22:21:19

Platform: amazon

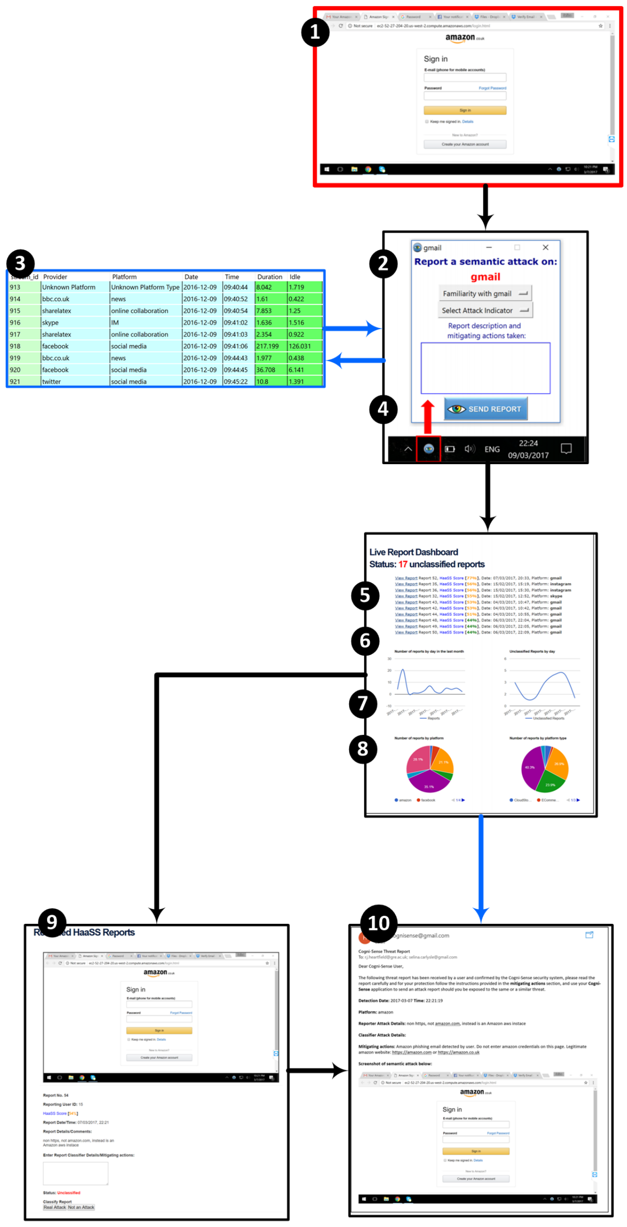
Reporter Attack Details: non https, not amazon.com, instead is an Amazon aws instace

Classifier Attack Details:

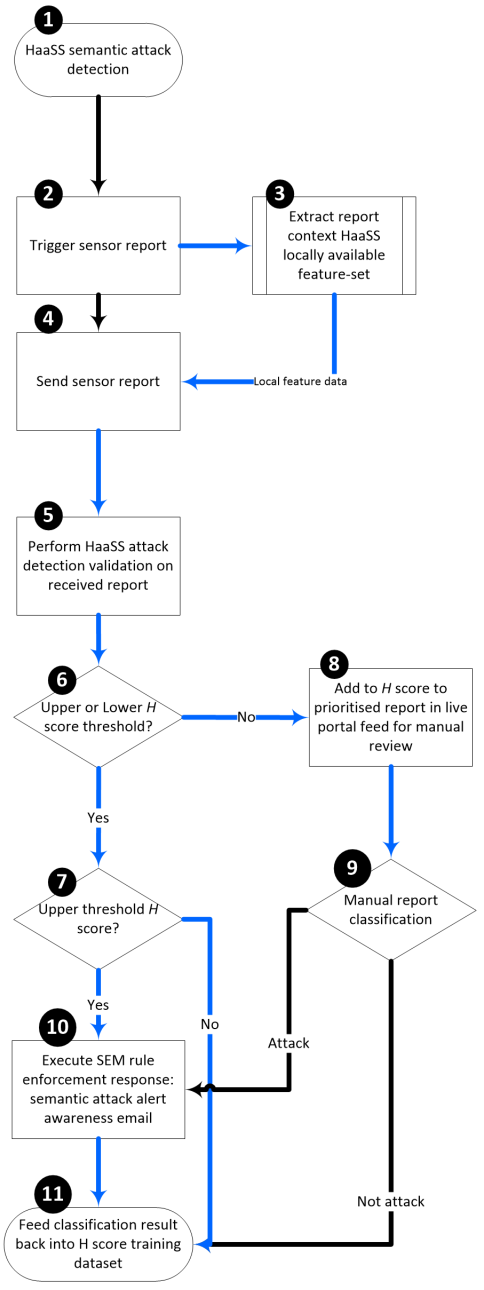
Mitigating actions: Amazon phishing email detected by user. Do not enter amazon credentials on this page. Legitimate amazon website: https://amazon.com or https://amazon.co.uk

Screenshot of semantic attack below:

Figure 4.11: Visual and system HaaSS semantic attack detection, classification and response flow diagrams. Numbers refer to related stages in the flow, blue arrows refer to automatic system activity and black arrows refer to manual human user activity



stream_id	Provider	Platform	Date	Time	Duration	Idle
913	Unknown Platform	Unknown Platform Type	2016-12-09	09:40:44	8.042	1.719
914	bbc.co.uk	news	2016-12-09	09:40:52	1.61	0.422
915	sharelatex	online collaboration	2016-12-09	09:40:54	7.853	1.25
916	skype	IM	2016-12-09	09:41:02	1.636	1.516
917	sharelatex	online collaboration	2016-12-09	09:41:03	2.354	0.922
918	facebook	social media	2016-12-09	09:41:06	217.199	126.031
919	bbc.co.uk	news	2016-12-09	09:44:43	1.977	0.438
920	facebook	social media	2016-12-09	09:44:45	36.708	6.141
921	twitter	social media	2016-12-09	09:45:22	10.8	1.391

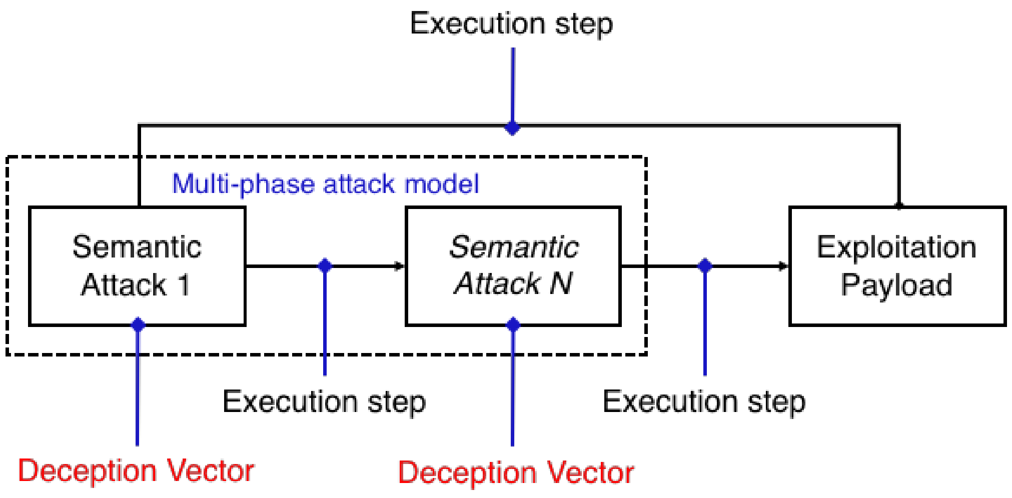

1
HaaSS semantic attack detection
2
Trigger sensor report
3
Extract report context HaaSS locally available feature-set
4
Send sensor report
Local feature data
5
Perform HaaSS attack detection validation on received report
6
Upper or Lower H score threshold?
No
8
Add to H score to prioritised report in live portal feed for manual review
Yes
7
Upper threshold H score?
9
Manual report classification
Yes
No
Attack
10
Execute SEM rule enforcement response: semantic attack alert awareness email
11
Feed classification result back into H score training dataset
Not attack

4.3 Evaluating the concept of HaaSS within a laboratory and real-world experiment

In this section we put the prototype HaaSS platform *Cogni-Sense* to the test, to evaluate the concept of HaaSS within two case studies, the first within a controlled laboratory environment and the second in an empirical real-world scenario. For both case studies we compare the ability for HaaSS sensors to detect each semantic attack's deception vector (see chapter 2) against a range of existing technical defences. Here, we define detection as the identification of the deception within a semantic attack and not the exploitation payload. That is, both HaaSS sensors and technical defences are evaluated by their ability to identify each semantic attack as a semantic attack, where a detection failure is recorded by either the HaaSS sesnsor or technical system allowing the deception to run up to the point of attack payload execution (e.g., clicking a link, entering credentials, or opening a file). Whilst some default behaviours may inherently block the execution of some of the experiment attack payloads (e.g., executable file opening attack landing page), all attack payloads are emulated and therefore a real malware may have different behaviour that allows it to successfully execute.

Therefore, for semantic attacks that logically consist of multiple phases (a semantic attack that directly lead to another separate semantic attack) we treat each phase as a separate attack in its own right, following the taxonomic attack classification methodology in chapter 2. This approach allows for measuring a HaaSS sensor's detection efficacy for each individual deception vector in a multi-phase semantic attack (e.g., (1) email URL $\rightarrow$(2) phishing website $\rightarrow$stolen credentials). In this case, a user exploitation to any single phase in a multi-phase attack could also be viewed as exploitation to an individual semantic attack which results in direct exploitation. For clarity, the attack model in Figure 4.12 highlights the points at which each semantic attack detection/exploitation is measured (deception vector text highlighted in red).

Figure 4.12: Experiment attack model for measuring H score and exploitation for individual semantic attack's deception vector in both singular and multi-phase semantic attacks



4.3.1 Case study 1: HaaSS Vs. Technical platform defences - the case of Selina Carlysle

In the following controlled laboratory-based experiment, we evaluate two major components of *Cogni-Sense*: (a) the HaaSS sensor semantic attack detection reporting mechanism *Cogni-Sense* app, installed on the participant experiment environment, as shown in Figure 4.6, and (b) the viability of the H score prediction as a utility to determine accurately HaaSS attack detection efficacy compared with a range of technical security platforms which claim to enforce anti-social engineering defences against different semantic attacks.

4.3.1.1 Zero-day semantic social engineering attacks

The vast majority of semantic social engineering attacks are largely undetectable by technical defence systems, because they primarily rely on cosmetic or behavioural deception vectors and as a result often leave very small technical footprint that can be analysed, especially if the deception has been designed to utilise intended user functionality [9]. Consequently, technical heuristic detection capabilities have a limited view of potential attack vectors through user actions, instead of system interfacing

Attack	Emulated Attack	Depend.	Description
1.1	Spear Phishing Email	-	Targeted participant email advertising job role specific to their profile from fake recruitment company with URL to purported job description PDF document on Google drive
1.2	Cloud Storage File Masquerading	1.1	Malware HTA file masquerading as PDF in online Google Drive folder
2.1	IM Phishing	-	Unsolicited Facebook message containing Facebook page link
2.2	Multimedia masquerading	2.1	Malicious image link masquerading as Facebook video post
3.1	Phishing Email	-	Order confirmation email from Amazon with order details and tracking URLs leading to phishing Amazon login web page
3.2	Phishing website	3.1	Amazon login phishing website which captures user login details

Table 4.2: Experiment emulated semantic attacks sent to participants with indicated date and time at which the attacks were launched for all participants (this does not guarantee that participants were exposed to the attacks at the time of launch)

malware. In most cases, technical defence systems rely on attack reports before they can develop signatures that can be matched against similar patterns when analysing potential threats, or attempting to pre-empt them.

For example, it is difficult to characterise a website as phishing if the URL is not registered with a spam database, and does not use obvious tricks such as similar domains names used as sub-domains, obfuscated by domain suffixes which are not related to the masqueraded website (e.g., amazon.net-shopping.tk). In cases where a phishing website name originates from a legitimate and credible service provider (and does not attempt to obfuscate its appearance), until the website has been reported as malicious (or contains easily identifiable malicious code or web re-directions in the web page), most technical defence platforms will not recognise the website as phishing. The same example can be seen in spam emails where spam protection mechanisms analyse components such as sender from and to address, subject title, domain, hyperlinks, attachments, salutation and common phrases (e.g., urgency) which match known common patterns in conventional phishing attacks. However, if the email body consisted purely of a deceptive image from a domain name not registered in a black list, then the classifiers effectiveness is significantly reduced, as a spam protection is unlikely to have the ability to interpret contextually the visual information in the image.

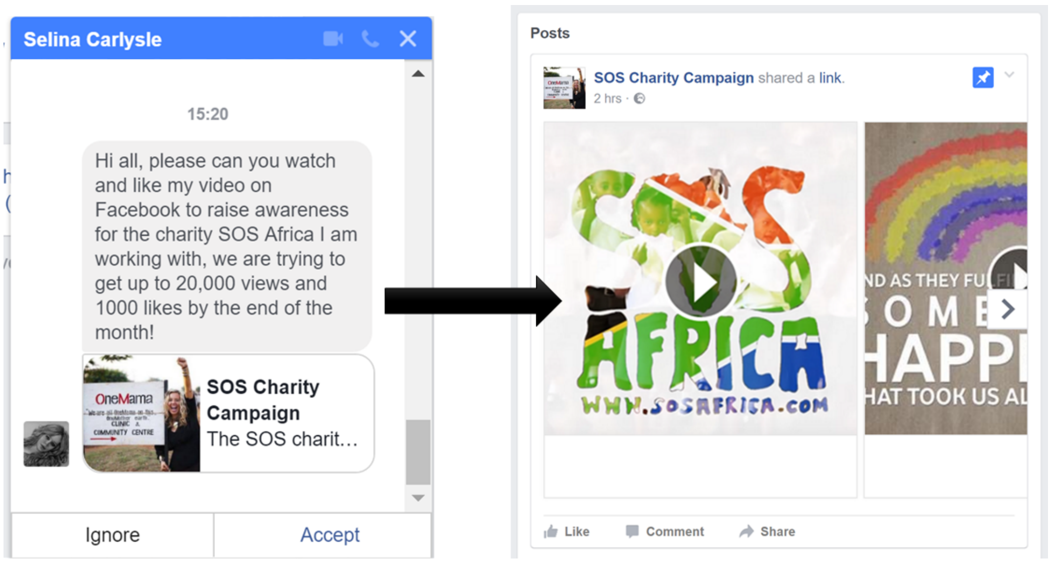
For technical defence systems to stand a chance in detecting unknown semantic attacks, defence mechanisms require the ability to interpret visual and behavioural attributes in real-time, simultaneously, to predict the likelihood that a deception attempt is occurring - which without knowledge of the targeted user or integration with the platform would likely result in many false positives or false negatives. On the other hand, human users, by definition, are implicitly interfaced with such attributes and are therefore best placed to decide whether system activity on the user interface is anomalous or not, based on their experience and knowledge.

As the attacks were developed specifically for this experiment, and therefore have not been seen by technical defence systems or users before, they are assumed to be zero-day semantic social engineering attacks at the time of the experiments. In Figures 4.13 (semantic attack 2.1 and 2.2) and 4.14 (semantic attack 3.2) we provide two examples of semantic attacks executed within case study 1. Table 4.2 provides an overview of the semantic attacks tested in case study one.

4.3.1.2 Laboratory environment

The experiment environment was presented in the form of a Windows 10 virtual machine which each participant could remotely access via *TeamViewer*. The task was a role-play exercise in the form of “a day in the life of Selina Carlyle” (an imaginary freelance artist), where all participants conducted a number of computer-based activities that Selina would typically carry out as part of her computer usage. This involved checking her e-mail on Gmail, accessing Facebook and reading messages, notifications, as well accessing other platforms such as Twitter, Pinterest and general web browsing for artwork. In the case of Twitter, Pinterest and web browsing, these were designed as noise activities to prevent participants from presuming that all attacks would reside within Gmail or Facebook.

Figure 4.13: Facebook automated phishing message and URL leading to a fake Facebook charity community page with malicious image link masquerading as a Facebook video post



Selina Carlysle

15:20

Hi all, please can you watch and like my video on Facebook to raise awareness for the charity SOS Africa I am working with, we are trying to get up to 20,000 views and 1000 likes by the end of the month!

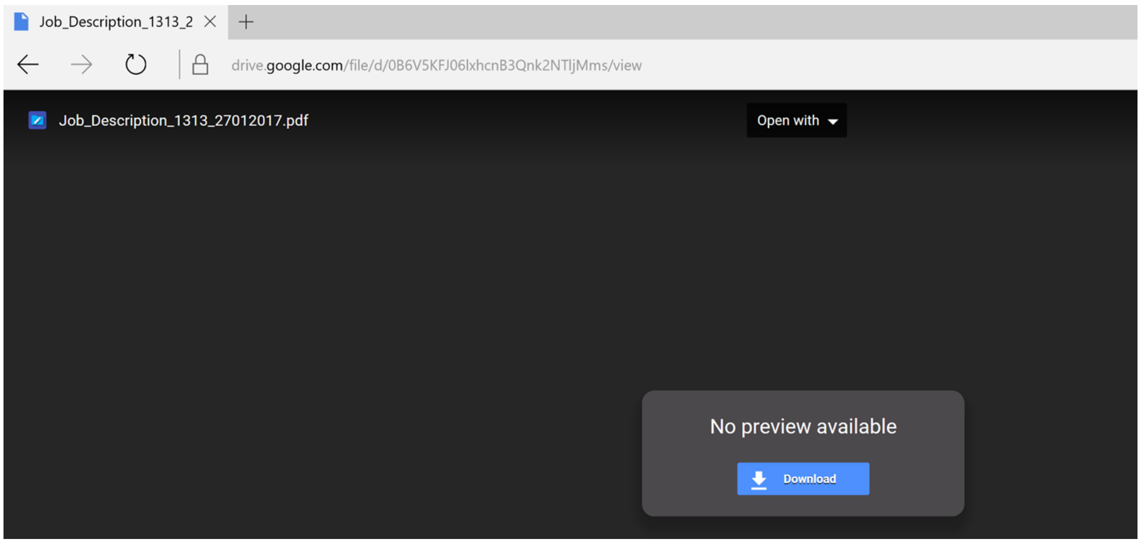
SOS Charity Campaign
The SOS charit...

Ignore Accept

Posts

SOS Charity Campaign shared a link.
2 hrs ·

Figure 4.14: File masquerading attack in Google drive cloud storage platform, filename appears to be a PDF but is in fact a HTML application (.HTA) file when downloaded.



Job_Description_1313_2

drive.google.com/file/d/0B6V5KFJ06lxhcnB3Qnk2NTljMms/view

Job_Description_1313_27012017.pdf

Open with

No preview available

Download

In total, seven participants were recruited for the experiment as HaaSS sensors by inviting a number of computer science students, lecturers and the general public to complete a questionnaire related to the experiment's purpose and their computer activity profile. The questionnaire described the role-play scenario, the goal of reporting detected attacks using the *Cogni-Sense* app, and collected "offline" the required HaaSS features for computing the H score of each participant with the Random Forest susceptibility model. Each of the participants were assigned a HaaSS number and reporting user ID to match reports to corresponding HaaSS sensors in the experiment. Participants were given also a user guide on how to use the *Cogni-Sense* reporting tool (Figure 4.6) when detecting a suspected semantic social engineering attack. The same Windows 10 virtual machine environment was also used to install and test individually each of the technical platform defences against each of the semantic attacks.

In table 4.4, each of the technical platforms and defence systems listed is evaluated according to the functional capabilities for detecting semantic attacks. Whilst email and browser platforms tend to offer anti-phishing, URL filtering and anti-malware defence, they do not directly employ heuristic scanning as part of this functionality, which as shown, is exclusively provided by the anti-virus software that we have evaluated. This means that in practice, most email providers rely on signature-based attack recognition for email by query through registered attack databases. It is important to note that for a number of the anti-virus products, installation of their full-product suite included browser security add-ons specifically designed for detection of website and email phishing threats and deception-based attacks. Amongst the anti-virus products, Norton, Sophos, Avast and Kaspesrky included and installed browser security add-on software as part of their security suite.

Due to the role-based constraints of the remote laboratory-based environment, participants' HaaSS feature-sets were implemented manually in the *Cogni-Sense* system under each HaaSS sensor (i.e., participant) profile. In this experiment, we did not use or evaluate the automatic HaaSS feature collection functionality developed in *Cogni-Sense* as the participants spent an average of thirty minutes attempting

ID	Ref	Security Platform	Platform Type	Phishing	Web Rating	URL blocking	Heuristics	On access malware
E1	[308]	Yahoo Mail	Email	✓	✗	✓	✗	✓
E2	[309]	Gmail	Email	✓	✗	✓	✗	✓
E3	[310]	Outlook	Email	✓	✗	✓	✗	✓
E4	[311]	ProtonMail	Email	✓	✗	✓	✗	✓
E5	[312]	Yandex	Email	✓	✗	✓	✗	✓
E6	[313]	GMX	Email	✓	✗	✓	✗	✓
E7	[314]	Mail.com	Email	✓	✗	✓	✗	✓
B1	[315]	Firefox	Browser	✓	✓	✓	✗	✓
B2	[316]	Chrome	Browser	✓	✓	✓	✗	✓
B3	[317]	Opera	Browser	✓	✓	✓	✗	✓
B4	[318]	Commodo Dragon	Browser	✓	✓	✓	✗	✓
B5	[319]	Avast Safezone	Browser	✓	✓	✓	✗	✓
B6	[320]	Microsoft Edge	Browser	✓	✓	✓	✗	✓
B7	[321]	Safari	Browser	✓	✓	✓	✗	✓
A1	[322]	Commodo Cloud	AV	✓	✗	✓	✓	✓
A2	[323]	AVG AntiVirus	AV	✓	✗	✓	✓	✓
A3	[324]	Avast AntiVirus	AV	✓	✗	✓	✗	✓
A4	[325]	Windows Defender	AV	✓	✗	✓	✓	✓
A5	[326]	Norton Security	AV	✓	✓	✓	✓	✓
A6	[327]	Kaspersky IS2017	AV	✓	✓	✓	✓	✓
A7	[328]	Sophos Intercept X	AV	✓	✓	✓	✓	✓
P1	[329]	Facebook	Platform	✓	✗	✓	✗	✗
P2	[330]	GoogleDrive	Platform	✓	✗	✗	✗	✓
P3	[331]	Windows10	OS	✗	✗	✗	✗	✗

Table 4.4: *Cogni-Sense* HaaSS platform configuration across the HaaSS framework architecture

to detect attacks, therefore sufficient learning time to collect activity data was unavailable. Instead, HaaSS features were extracted from input from the participant questionnaire as per the approach used in chapter 3. We do, however, evaluate the functionality of security enforcement module (SEM) integration attack classification e-mail alerting (as shown previously in Figure 4.9), which was triggered by conducting manual classification on a semantic attack report received by HaaSS sensor ID 15 during the experiment. Participants did not have access to the cloud-based *Cogni-Sense* portal during or after the experiment and were unaware of reports made by other participants during the experiment.

In this experimental environment, we aim to compare the detection efficacy of HaaSS sensors for identifying zero-day semantic social engineering attacks against the detection efficacy of technical defence systems, in order to show the potential usefulness of the HaaSS concept for detecting deception-based threats.

4.3.1.3 Case study 1 Results

In Tables 4.5, 4.6 and 4.7, we compare the experimental results for the HaaSS participants and each of the technical platforms for detecting the semantic attacks in Table 4.2. In each table the colour refers to detection result: red - not detected, green - detected, orange - precautionary measure taken, but no threat reported, grey - attack not seen (due to the nature of the attack for the technical defence or because the HaaSS sensor did not notice or was not exposed to the attack and therefore gave no response).

The spear-phishing email in attack 1.1 proved the most challenging one to detect for HaaSS participants, with only three out of seven participants correctly reporting the email as an attack. However, three out of four of the HaaSS sensors who were exploited by the email by clicking on the GoogleDrive link, then detected the malicious HTML application file in Google drive afterwards (attack 2.2). The H score prediction at a probability threshold of 50% threshold was only 43% accurate, but at a 65% threshold, it was 86% accurate, with a 100% true positive rate and false positive rate of 25%. By comparison only two out of the seven email providers, Yandex and Yahoo mail, sent the email to spam, with all others placing the email in the inbox folder with no warnings of a suspected attack. All browsers and AntiVirus failed to detect the email or subsequent file in GoogleDrive (in the online platform or when downloaded and run) as malicious; with no user warnings at all; with the exception of Firefox (B1) and Windows 10 (P3), which prompted the user that they were downloading/running an executable file. Moreover, Commodo Cloud AV (A1) also ran the downloaded file, but within in a sandbox as a default action of the software because it was an executable file of unknown provenance (i.e., missing a trusted digital signature).

For the IM phishing message (attack 2.1) on Facebook only three out of seven HaaSS sensors were actually exposed to the semantic attack, as the message originated from an account which was not a Facebook friend of the Selina persona. This meant that the message was placed under Facebook "message requests" which is more hidden than friend based messages or profile notifications. For the three HaaSS sensors that

	HaaSS						
Attack	**H1**	**H2**	**H3**	**H4**	**H5**	**H6**	**H7**
1.1	.68	**.92**	.61	.77	.64	.61	.78
1.2	.66	**.87**	.49	.19	**.77**	.52	.76
2.1	.78	**.85**	.64	.90	.74	.23	.84
2.2	.78	**.85**	.64	.90	.74	.23	.84
3.1	.68	**.92**	.61	**.77**	.64	.61	.78
3.2	.68	**.79**	.53	.46	.54	.57	.75

	Email Provider						
Attack	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**
1.1	✗	✗	✓	✗	✓	✗	✗
1.2	-	-	-	-	-	-	-
2.1	-	-	-	-	-	-	-
2.2	-	-	-	-	-	-	-
3.1	✗	✗	✓	✗	✓	✗	✗
3.2	-	-	-	-	-	-	-

Table 4.5: HaaSS sensor (left) and Email provider (right) detection results

	Browsers						
Attack	**B1**	**B2**	**B3**	**B4**	**B5**	**B6**	**B7**
1.1	✗	✗	✗	✗	✗	✗	✗
1.2	✓	✗	✗	✗	✗	✗	✗
2.1	✗	✗	✗	✗	✗	✗	✗
2.2	✗	✗	✗	✗	✗	✗	✗
3.1	✗	✗	✗	✗	✗	✗	✗
3.2	✗	✗	✗	✗	✗	✗	✗

	AntiVirus						
Attack	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
1.1	✗	✗	✗	✗	✗	✗	✗
1.2	✓	✗	✗	✗	✗	✗	✗
2.1	✗	✗	✗	✗	✗	✗	✗
2.2	✗	✗	✗	✗	✗	✗	✗
3.1	✗	✗	✗	✗	✗	✗	✗
3.2	✗	✗	✗	✗	✗	✗	✗

Table 4.6: Browser provider (left) and Anti-virus provider (right) detection results

were exposed, all three clicked on the link in the message, which was included as an image hyperlink leading to another Facebook page. At this point during the technical testing, no technical defences (including the Facebook platform itself) had flagged the message as malicious or anomalous. Even though all three HaaSS sensors were exposed to the video masquerading images on the fake Facebook charity page (attack 2.2), none of the users actually clicked on the image, but they also failed to report it as a semantic attack. Again, none of the technical defences and the Facebook platform itself detected the image as malicious (which was expected given the fact that the image was simply using the in-built Facebook image hyperlink functionality (posing as a video using cosmetic features in the image). Here, the H score predicted that the three users exposed to the threat would detect it, however given the lack of exposure to the threat in a realistic context (e.g., participants' own personal accounts), it is unclear whether the actual response is robust enough to discount the prediction of the H score.

For the amazon phishing email (attack 3.1), six out of seven HaaSS sensors detected the attack, whereas only the Yandex email provider sent the email to spam;

	Platform		
Attack	**P1**	**P2**	**P3**
1.1	-	-	✗
1.2	-	✗	✓
2.1	✗	-	✗
2.2	✗	-	✗
3.1	-	-	✗
3.2	-	-	✗

Table 4.7: Host platform detection results.

Yahoo email blocked the email's images but did not flag the email as malicious. All other technical defences failed to identify the email as a threat or provide any warnings. The *H* score correctly predicted six out of seven HaaSS detections at the 50% threshold, but would only have detected four out of seven correctly at the 65% threshold. The one HaaSS user that was exploited by clicking on the link in the Amazon phishing email, then correctly detected the following Amazon phishing login webpage - an action correctly predicted by the participant's *H* score. Again, all technical defences and platforms failed to detect the phishing website as malicious.

In Figure B.6 an example the amazon phishing email (3.1) attack is shown delivered into the inbox of Gmail, Outlook and Yahoo accounts. Figure B.7 shows an example of the Comodo Dragon web browser failing to detect both the amazon phishing email and website as malicious. Evaluating the phishing website with the browsers built-in *WebInpector* tool in Figure B.8 resulted in the detection of no malicious activity; failing to recognise the spoofing of the Amazon login page.

Overall, the HaaSS sensors were more efficient at detecting all threats than the technical defences exposed to the semantic attacks - without prior knowledge of the attacks themselves or any training provided prior to this experiment. By comparison, the technical defences in almost all cases failed to detect the existence of a threat. Surprisingly, Yahoo detected the spear phishing email as spam, but not the Amazon email which used as a simple header alias to look as if the email originated from Amazon. An example of a HaaSS detection result for attack 3.2 is shown in Figure 4.8, which is the screen capture of the users screen made by the *Cogni-Sense* reporting app (sent to the cloud portal) and the accompanying report details showing the computed

H score for the report, time and date when the report was made and the HaaSS sensor report observation details.

From a classification perspective, HaaSS was 68% accurate at a 50% probability threshold with a true positive rate of 93% and true negative rate of 43%, precision of 62%, false positive rate of 57% and false negative rate of 7%. However, at the 65% probability threshold, it achieved an accuracy of 64% with a true positive rate of 67%, true negative rate of 6%, precision of 67%, false positive rate of 4% and false negative rate of 33%. Where the *H* score was shown to be the same value for consecutive attacks on the same platform (e.g., email on Gmail, social media on Facebook), this is due to the report and classification processes occurring in essentially the same time period, with the same feature set.

In the case of organisational HaaSS defence, if these participants were HaaSS sensors in a security platform, prioritising these HaaSS reports based on the *H* score probability results in the experiment would ensure almost all of the attacks were identified before reviewing a non-attack report. Furthermore, by utilising such HaaSS sensors the organisation would indeed detect the semantic attacks in the first place, which would be unlikely if they were to rely purely on the technical defences evaluated in the experiment.

4.3.2 Case study 2: Crowd-sourced HaaSS in the cloud

In case study two we conduct a larger scale, real-world experiment for HaaSS semantic attack detection, integrating participants and their personal systems directly into the prototype HaaSS platform *Cogni-Sense*. Here, we use the concept of crowd-sourcing HaaSS sensor detection to utilise attack report telemetry for semantic attack detection. In this scenario crowd-sourcing HaaSS sensor capabilities can be representative of multiple HaaSS platform structures, such as general HaaSS sensors who are computer users subscribed to a public HaaSS system, within a business organisational setting (which would include general business users and security operations users), or even paid human intelligence task workers (HITs) [332, 333] that have been effectively integrated within a hybrid human/machine technical security system [334, 335], which draws from a pool of user profiles qualifying as HaaSS sensors. In the case of the latter, it is reasonable to assume that the highest performing HaaSS sensors, as proven by their track record and detection performance, would likely command a higher fee. In all cases, a key requirement is for HaaSS platform integration directly into different users computer devices, which in the era of Internet of Things and Bring-Your-Own-Device (BYOD) within working environments means that more often than not these computer devices are the users own.

4.3.2.1 Empirical experiment environment

We have developed case study two in adherence with the five design principles for user studies in security and privacy proposed in [336]. In experiment case study two, participants are assigned (i) a primary task of reporting suspected semantic attacks using the *Cogni-Sense* software; (ii) where the semantic attacks introduce a realistic risk by exposing users to real-world deception vectors on their own computer systems, where similar real attacks could be easily received by the participant before, during or after the experiment; (iii) the participants are not primed to the nature of the semantic attacks and how they will be received to prevent detection bias; (iv) adding element of double-blinding as the researchers do not know when participants will be exposed to the emulated experiment attacks or whether their systems security will block the attacks from reaching the participants and (v) ensuring the terminology and dissemination of the experiment, in relation to the delivery method of the semantic attack threat, security and privacy is consistent in order to prevent bias in the participants behaviour and overall reported results.

As the experiment HaaSS sensor report interface *Cogni-Sense* app is installed directly on personal computer devices, the experimental environment was primarily presented in the form of the participant's own system. However, as the HaaSS reporting interface in *Cogni-Sense* a HaaSS sensor app was developed for Windows operating systems, participant recruitment was constrained by the requirement for their primary computer devices to have a Windows operating system installed (Windows 7, 8 or 10). Consequently, this also required participant devices to be a desktop

PC, laptop or tablet; mobile devices were not used as part of this experiment. Due to this condition, the specific host platform configuration (aside from the *Cogni-Sense* app) was different from one participant to another. For example, different HaaSS sensors will have installed and use different software and computer security applications, therefore by employing a crowd-sourced approach to HaaSS sensor integration specific platform configurations associated to the HaaSS sensors device operating system and installed applications were unknown.

The goal for participants in the experiment was to detect and report any suspected attacks without falling victim to them by using the *Cogni-Sense* app HaaSS report interface within *Cogni-Sense*, which participants installed on their system. As part of the experiment, participants were advised to use the application to send a report whenever they detected a suspected attack. Participants were encouraged to use their computer devices as per their usual pattern of behaviour, under the condition that they may or may not be subjected to a number of emulated social engineering attacks, at random times, during the experiment time period; lasting for 6 weeks. As part of the recruitment process, participants took a questionnaire which described the overall experiments scenario, which was also used to collect the HaaSS computer security features. Unlike case study one, in case study two the experiment involves installation of a user app on the participants personal device, which enables *Cogni-Sense*'s automatic feature collection and generation functionality for frequency (FR1, FR2), duration (DR1, DR2) and familiarity (FA1) to be utilised. For computer security features, these were implemented manually through the recruitment questionnaire as a baseline set, and then updated during the experiment time period by sending weekly e-mail reminders to participants to update their security training profile (only if they had receive training), using online form, shown in Appendix 2, Figure B.4). As per case study one, each of the participants were assigned a HaaSS number and reporting user ID to match reports to corresponding HaaSS sensors in the experiment and each participant received a user guide on how to use the *Cogni-Sense* app reporting tool when reporting a detection of a suspected semantic attack. During the experiment, if any participant was exploited by one of the semantic attacks, they were either directed to a second attack, as shown in Table 4.2 and Table 4.8 or redirected to an attack landing page where the participant was required to enter their experiment ID and name. For both cases, should participants fail to report an attack or submit their details after exploitation, this information would still be available for offline analysis via the *Cogni-Sense* app activity collection process. The landing page is shown in Figure B.5

In total, 26 HaaSS sensors were recruited by inviting participants to partake in the experiment with the incentive of a 50 participation voucher given to each participant at the end of the experiment's six week period. The participants, which consisted of students, lecturers, working IT professionals and the wider general public, were recruited as an intentionally diverse demographic that vary in terms of their expert computer literacy and computer security awareness, without specifically organising participants based on their individual skills. In a HaaSS system, all employees in an organisation or the general public (e.g., from the Internet) are viable HaaSS sensors. However, it is the application of the H score that distinguishes between sensors by providing a metric of the expected reliability (and therefore detection efficacy) of different sensors semantic attack reports. With this in mind, it is preferable to be agnostic of the HaaSS sensor from a demographic perspective, instead relying on the H score to provide a unique probability of detection efficacy, which is dynamic based on time and training, rather than profiling a sensor individually (e.g., age, gender, personality); which we have shown to be of limited value and impractical in a technical system [304].

4.3.2.2 Case study 2 semantic attacks

In case study two, we extend beyond the semantic attacks listed in Table 4.2 by extending existing and adding new attacks. The new attacks are aimed to test HaaSS sensors against a wider range of deception techniques and platforms to evaluate detection efficacy rigorously against across a comprehensive suite of threats to confidently generalise detection performance. In Table 4.8, the new attacks are described. We

Attack	Emulated Attack	Depend.	Description
2.1a	Fake Facebook account	-	Friend request from fake Facebook account
4.1	Automated IM phishing	-	Unsolicited Facebook message from non-friend account with shortened URL
4.2	Phishing website	4.1	Facebook login phishing page
5.1	Spear USB	-	Packaged and branded USB designed specifically for target, delivered in the post
5.2	File masquerading	5.1	Executable masquerading as PDF file

Table 4.8: Experiment emulated semantic attacks sent to participants with indicated date and time at which the attacks were launched for all participants (this does not guarantee that participants were exposed to the attacks at the time of launch)

introduce a Facebook account friend-request from a fake profile, combined with an assocaited Facebook IM phishing message in attack 2.1, another Facebook instant messaging phish (with shortened malicious URL) and a spear USB attack, which contains a malicious executable file masquerading as a PDF file with an adobe acrobat PDF icon.

The spear USB attack branches the semantic attacks deployed in the experiment into physical space and tests whether HaaSS sensor detection can be equally useful in detecting threats that cross a cyber-physical domain, and to evaluate the detection of deception-based attacks where computer devices require physical interaction with hardware interfaces as an entry vector to the system. An example of the spear USB attack is shown in Figure 4.15 and PDF file masquerading attack in Figure 4.16. Each of the spear USBs were designed to be targeted by printing on them official logos associated specifically to platforms the participants reported to use and have a specific affinity for in terms of their Internet profiles.

Figure 4.15: Spear USB attacks (Facebook, Instagram and Blackhat participant profiles)

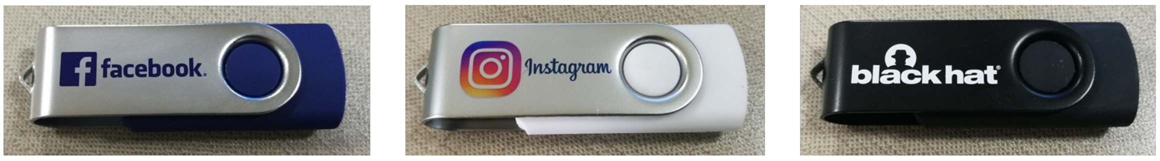
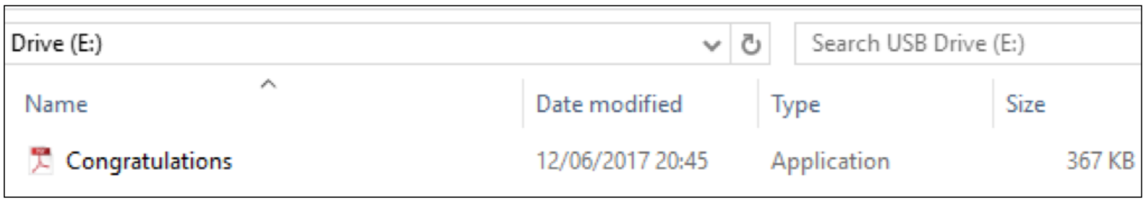


Figure 4.16: PDF File masquerading


Drive (E:)

Search USB Drive (E:)

Name	Date modified	Type	Size
Congratulations	12/06/2017 20:45	Application	367 KB

4.3.2.3 Case study 2 results

In Tables 4.9, 4.10 and 4.11, the HaaSS sensor participants results for case study two are shown, as well as the performance of the technical defence platforms for detection of the new attacks introduced.

General observations: In case study two, exposure to the experiment attacks was quite varied across HaaSS sensors. Most HaaSS sensors were not exposed to a majority of the attacks, with at least three HaaSS sensors only being exposed to one attack set. However, exposure was also dependent on attack detection efficacy. For example, a HaaSS sensor with perfect detection performance would only have ever been exposed to five of the eleven attacks in total. For Facebook attacks (2.1, 2.2, 4.1 and 4.2), at least 8 out of 26 HaaSS sensors did not have Facebook accounts or had prevented their profiles from being searchable, meaning these eight HaaSS sensors would definitely not have been exposed to these semantic attacks.

In total, 17 out of 26 (65%) HaaSS sensors detected at least one semantic attack, with the HaaSS sensor base in the experiment detecting all semantic attacks across all platforms; 2.2 (Facebook video media masquerading) was the only semantic attack not reported by a HaaSS sensor. By comparison, only 3 out of 24 (13 %) technical defences detected a semantic attack in the experiment and these detections were specifically limited to phishing emails only; all of which only identified one of the phishing emails each. Therefore, all other semantic attacks on different platforms went undetected by the technical defences.

In terms of individual attack detection, the HaaSS sensors did not find any specific attack particularly simple to detect. For attacks 3.1 (Amazon phishing email) and 4.1 (Facebook phishing message), there was an equal number of HaaSS reports to

	HaaSS sensors																									
A	**H1**	**H2**	**H3**	**H4**	**H5**	**H6**	**H7**	**H8**	**H9**	**H10**	**H11**	**H12**	**H13**	**H14**	**H15**	**H16**	**H17**	**H18**	**H19**	**H20**	**H21**	**H22**	**H23**	**H24**	**H25**	**H26**
1.1	.49	.72	.54	.55	-	.59	✗	.50	-	-	.70	.59	.48	-	.53	-	-	.33	.46	-	-	.46	.54	.70	.79	.66
1.2	-	.66	.53	-	-	-	✓	.26	-	-	-	-	.34	-	.18	-	-	.44	-	-	-	-	.17	-	.71	-
2.1A	-	-	-	-	-	.60	✓	-	-	-	.58	-	-	-	.48	-	-	-	-	.49	.30	-	-	-	-	-
2.1B	.51	-	-	-	-	-	✓	.51	-	-	.58	.43	-	-	.48	-	-	.32	-	.49	-	-	-	-	-	-
2.2	.51	-	-	-	-	-	-	-	-	-	-	-	-	-	.48	-	-	.32	-	-	-	-	-	-	-	-
3.1	-	-	.54	.55	-	.60	✗	.63	-	.51	.70	.59	-	.56	.56	.46	.49	.33	.46	-	-	-	.54	.70	-	.66
3.2	-	-	-	-	-	-	.62	-	-	.49	-	-	-	.48	-	-	.26	.44	.46	-	-	-	-	.49	-	.67
4.1	-	-	-	.63	-	.56	-	-	-	.53	.72	.43	-	-	-	-	-	-	-	.49	-	-	-	-	-	-
4.2	-	-	-	-	-	-	-	-	-	.56	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5.1	-	.67	-	.61	✗	.63	✗	.39	.35	-	.65	.52	-	-	.49	-	-	.45	.47	-	.30	.37	.66	.44	-	.68
5.2	-	-	-	-	✗	-	✓	-	.35	-	-	-	-	-	-	-	-	-	.47	-	.57	-	.68	.72	-	.64

Table 4.9: HaaSS sensor attack detection results for case study 2. The value in each cell refers to the *H* score. Note that the HaaSS sensor number shown here is different to the HaaSS sensor ID assigned to participants during the experiment, as the IDs in *Cogni-Sense* were automatically generated by a database and not contiguous.

	Browsers						
Attack	**B1**	**B2**	**B3**	**B4**	**B5**	**B6**	**B7**
2.1A	✗	✗	✗	✗	✗	✗	✗
4.1	✓	✗	✗	✗	✗	✗	✗
4.2	✗	✗	✗	✗	✗	✗	✗
5.1	-	-	-	-	-	-	-
5.2	-	-	-	-	-	-	-

	AntiVirus						
Attack	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
2.1A	✗	✗	✗	✗	✗	✗	✗
4.1	✗	✗	✗	✗	✗	✗	✗
4.2	✗	✗	✗	✗	✗	✗	✗
5.1	✗	✗	✗	✗	✗	✗	✗
5.2	✓	✗	✗	✗	✗	✗	✗

Table 4.10: Browser provider (left) and anti-virus provider (right) detection results for new attacks in case study two

	Platform	
Attack	**P1**	**P3**
2.1A	✗	-
4.1	✗	-
4.2	✗	-
5.1	-	✗
5.2	-	✗

Table 4.11: Host platform detection results for new attacks in case study two

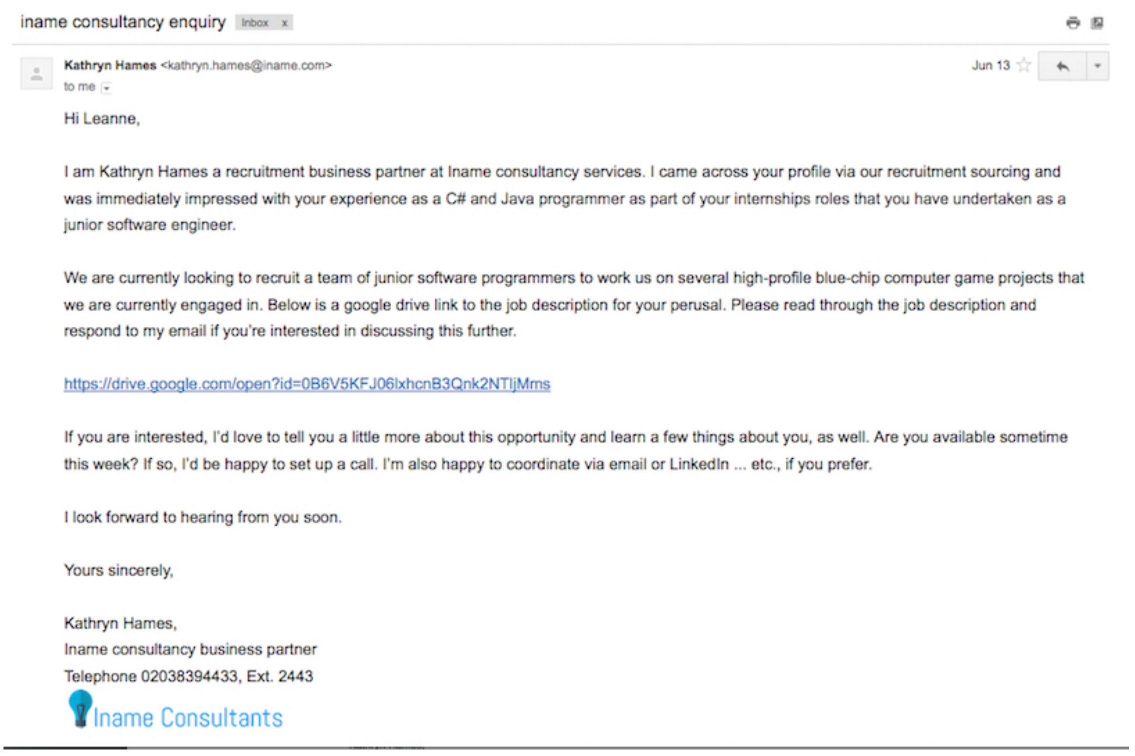
HaaSS attack exploitation, which was shown to be the highest performance for the HaaSS sensors in the experiment. Nonetheless, attack 1.1 (spear phishing email), similar to case study 1, continued to be a challenging attack to detect for the HaaSS sensors and in this case was specifically tailored to the HaaSS sensor (and therefore contextually relevant to each sensor). The new USB spear phishing attack 5.1 also proved equally challenging, with both attacks 1.1 and 5.1 exploiting at least 35% and 38% of HaaSS sensors, respectively. For HaaSS sensors exploited by attack 1.1, 67%

were also exploited by attack 1.2 (Google Drive file masquerading), and for attack 5.1, 60% were also exploited by 5.2 (PDF file masquerading). The most difficult attack in the experiment for HaaSS sensors appeared to be the Google Drive PDF file masquerading, whereby 88% of HaaSS sensors exposed to the attack were deceived into downloading the fake file.

For case study 2, the H score was generally less accurate for the spear phishing e-mail and spear USB attacks (as was the case with case study one), predicting high probabilities of detection efficacy for specific HaaSS sensors. However, it is worth highlighting the complexity of these two semantic attacks in particular, compared to the rest of the attacks deployed in case study two. The spear phishing attack was particularly well-crafted, with very little indication of the signs the the e-mail was illegitimate. For example, by analysing Figure 4.17 the Google drive URL is in fact legitimate, and the e-mail communication appears credible with the exception of the domain name that even though it matches the email sender appears to originate from a company called "Iname", which is in fact fictitious. In this case, a quick online investigation identifies "@iname.com" as a free email domain provided by Mail.com email provider and the company has no public website listed or details available by search engine.

For the spear USB attack, the device was posted directly to HaaSS sensors' home addresses, labelled professionally with logos from platforms and companies based on online profiles they associated most affinity to during the participant survey recruitment. The HaaSS sensors needed only to insert the USB into any of their systems for the deception to have been successful, with no technical defence being capable of preventing this physical action from being executed; other than that of the HaaSS sensors themselves (although as a pre-emptive technical defence a USB write-blocker could potentially provide some defence here). In a real world scenario it is quite possible that the USB device would contain zero-day malware, which may have compromised a users system immediately, irrespective of whether they used a Windows, Mac or Linux operating system; a malware designed for these platforms could easily have been planted on the device. On the other hand, for this attack in particular, it

Figure 4.17: Spear phishing e-mail (attack 1.1) from Iname Consultants tailored to each specific HaaSS sensor participant


iname consultancy enquiry Inbox x

Kathryn Hames <kathryn.hames@iname.com>
to me

Jun 13

Hi Leanne,

I am Kathryn Hames a recruitment business partner at Iname consultancy services. I came across your profile via our recruitment sourcing and was immediately impressed with your experience as a C# and Java programmer as part of your internships roles that you have undertaken as a junior software engineer.

We are currently looking to recruit a team of junior software programmers to work us on several high-profile blue-chip computer game projects that we are currently engaged in. Below is a google drive link to the job description for your perusal. Please read through the job description and respond to my email if you're interested in discussing this further.

https://drive.google.com/open?id=0B6V5KFJ06lxhcnB3Qnk2NTljMms

If you are interested, I'd love to tell you a little more about this opportunity and learn a few things about you, as well. Are you available sometime this week? If so, I'd be happy to set up a call. I'm also happy to coordinate via email or LinkedIn ... etc., if you prefer.

I look forward to hearing from you soon.

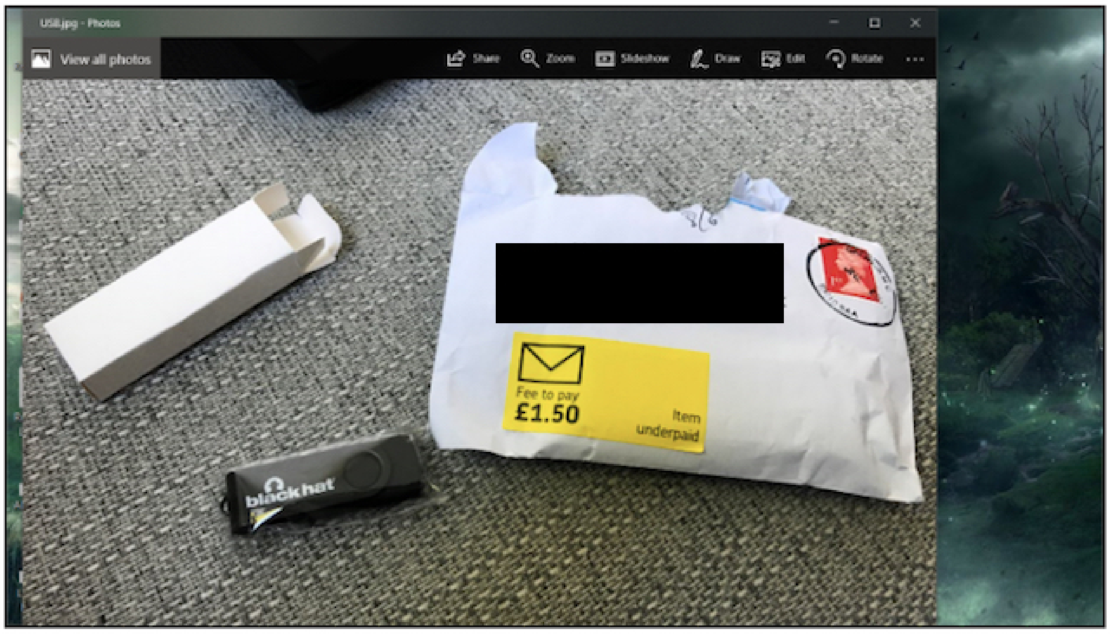
Yours sincerely,

Kathryn Hames,
Iname consultancy business partner
Telephone 02038394433, Ext. 2443
Iname Consultants

is also possible that participant HaaSS sensors did not expect to be exposed to, or even expected to report semantic attacks that manifest in cyber-physical form.

Nevertheless, 23% of HaaSS sensors detected attack 5.1 and generated semantic attack reports for the spear USB attack in physical space, which demonstrates deception-based attack detection that simply would not have been possible for a technical defence system. One such report is shown in Figure 4.18, where a picture was taken of the USB and reported via the *Cogni-Sense* app.

Figure 4.18: HaaSS sensor report for spear USB through *Cogni-Sense* app



Report No. 107

Reporting User ID: 29

HaaSS Score [61%]

Report Date/Time: 19/06/2017, 21:47

Report Details/Comments:

Unsolicited USB received through post. I am attending BlackHat this year so attack was very targeted. I did not plug the USB into any system I use, I instead put it though a sheepdip process on a standalone system, running the USB

Performance of the H score: In case study two, the HaaSS scores generated for sensor reports were less confident than those generated in case study one, with the probability of detection relatively proximal to the default classification threshold of 0.5 for the majority of reports. Further, analysis showed a low standard deviation of 0.07 from the H score sample mean of 0.56, with the most confident correct H score only reporting a 71% detection probability as the highest prediction result. Consequently, no reports qualified for the automatic H score report classification threshold and therefore all reports were sent for manual classification in the semantic attack sandbox. The lower H score confidence in case study two may be explained by a number of reasons. Firstly, in case study two, HaaSS sensors auditable (i.e., activity) features were only collected on one device and therefore a large proportion of their platform usage may not have been seen by the *Cogni-Sense* app. This would have resulted in less accurate platform frequency and duration features which may have been significantly different if the HaaSS sensors spent most of their time using mobile devices. Secondly, unlike case study one which computed scores for participants at one point in time, computer security training within case study two automatically decreased over time, unless it was updated by HaaSS sensors when they reported receiving training. As a result, the measurement for HaaSS sensors features in case study 2 was dynamic and thus dependent on participants keeping their computer security features up-to-date and limited by the visibility of the *Cogni-Sense* app for sensor activity on the device in which it was installed.

In Table 4.12 we evaluate the performance of the H score in case study 2, reporting the model performance using the default and optimal classification thresholds. Here, a true positive (TP) refers to a correctly predicted HaaSS sensor attack report, a true negative (TN) refers to an incorrect attack report (e.g., report was not a semantic attack) or correctly predicted HaaSS sensor exploitation, a false positive (FP) refers to an exploited HaaSS sensor who was predicted to have detected the attack and a false negative is an incorrect prediction of a HaaSS not reporting an attack.

We do not include in the analysis HaaSS sensor reports or exploitation records for which it was not possible to generate a H score as this would provide biased increase in the naive classifier by increasing the detection rate through more samples than were made available to the H score. HaaSS sensors who were not exploited by attacks by performing mitigating actions (orange box in Table 4.9), but who also failed to report them were not included in the analysis. In case study 2, it was not possible to generate H scores for HaaSS sensors' H5 and H7 exploitation records because their database files used to generate the frequency and duration features via the *Cogni-Sense* app was deleted by the participants at the end of the experiment and therefore not supplied for analysis.

The optimal H score classification threshold was reported at the probability cut-off value of probability of 52% for predicting HaaSS sensors attack reporting and

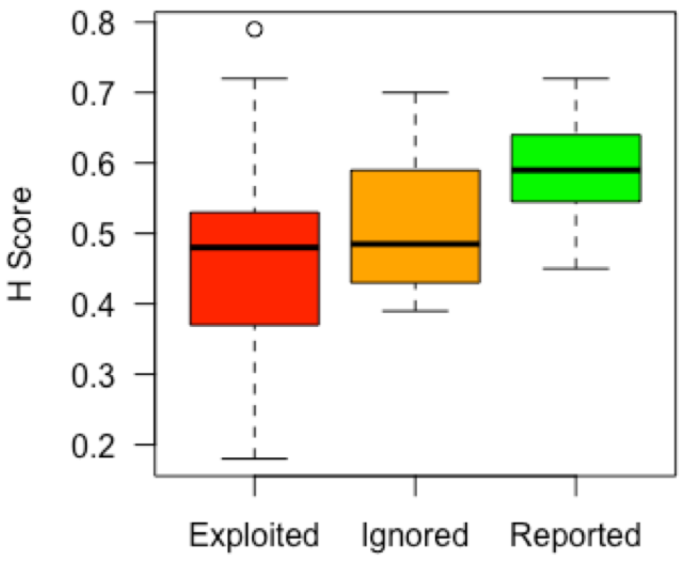
Class. Threshold	Accuracy	TP	TN	FP	FN	Precision
.50 (default)	.69	.32	.35	.25	.06	.57
.52 (optimal)	.74	.32	.42	.20	.06	.62
Naive classifier	.62	0	.62	0	.62	0

Table 4.12: *H* score detection efficacy classification performance for HaaSS sensor reports in case study 2 - Naive refers to the naive classifier performance as per chapter 3 and in this case refers to classifying HaaSS sensors as susceptible (i.e., unlikely to report semantic attack

exploitation, which demonstrated a 12% increase in prediction accuracy over the naive classifier (i.e., no information rate). In the case of the default classification threshold of 50%, the *H* score was only 7% more accurate than the naive classifier.

Arguably the most crucial validation of the *H* score performance (and its viability as a robust measure of sensor reliability) is observed by aggregating *H* scores and corresponding HaaSS sensor detection results. In Figure 4.19, a box-plot of the *H* scores in case study two shows that as the probability of detection efficacy increased (as predicted by the *H* score), semantic attack detection efficacy was indeed higher. Equally, as the prediction of probability lowered, sensor detection efficacy was lower and interestingly where sensors' were not exploited by a semantic attack, but also failed to report it, the *H* score was a median between the two.

Figure 4.19: *H* score predictions for HaaSS sensor responses in case study two



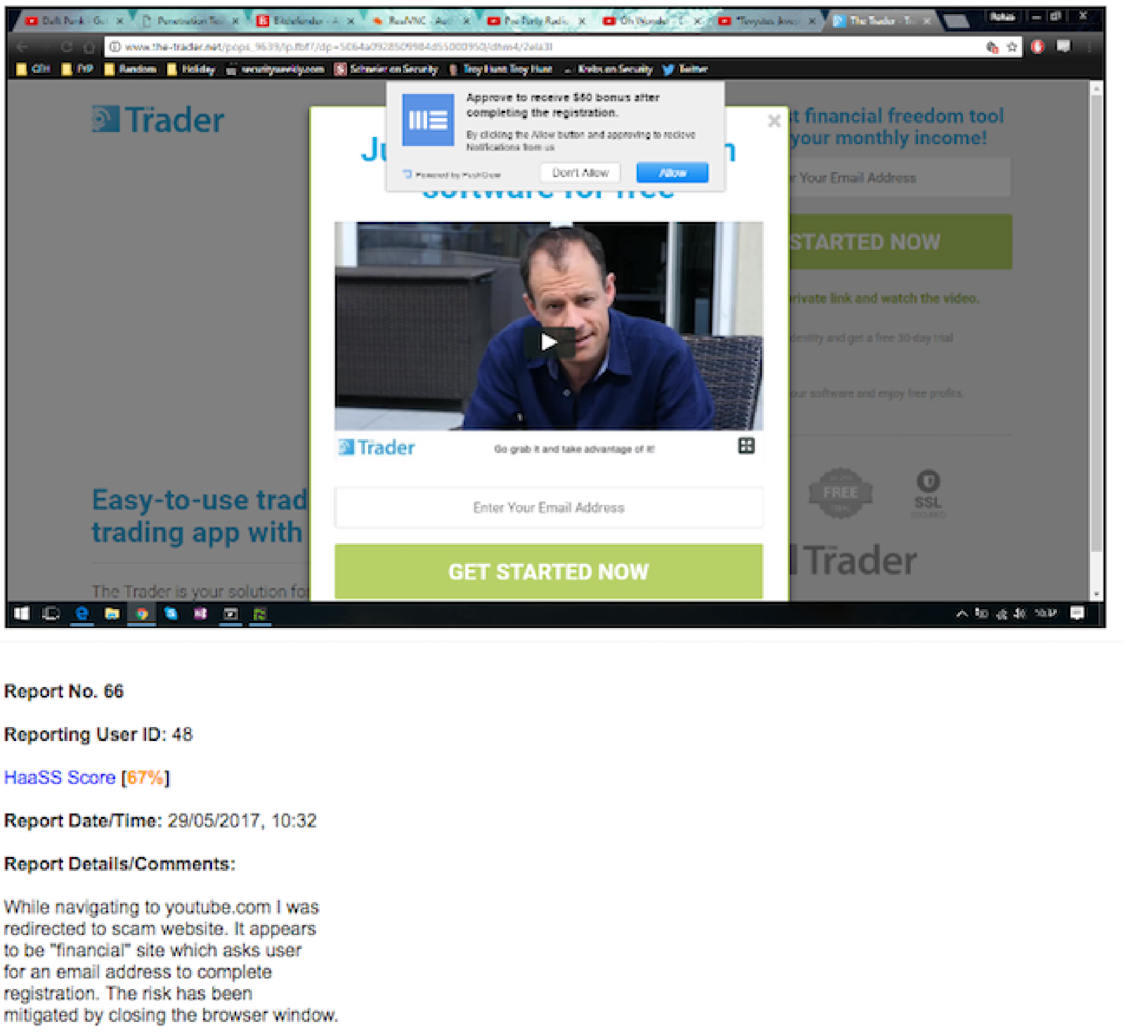
Overall, the H score's performance was consistent with the experimental results received in chapter 3 and therefore confirms the validity of the the H score prediction model within the context of empirical, real-world application.

HaaSS detection of further semantic attacks: During case study two a total of forty nine extra HaaSS sensors reports were received for suspected attacks that did not originate from the emulated attacks. In total, 26 of these reports were correctly detected semantic attacks and 23 were incorrectly detected as semantic attacks. Interestingly, multiple reports of typosquatting attacks were received, further validating the HaaSS concept for detecting and reporting an even wider range of semantic attacks than those evaluated within case study two. In Figure 4.20 an example typosquatting report is shown from HaaSS sensor 11, with a H score of 67% (who also reported the highest detection rate overall in the experiment).

Expert reviewer sandbox classification (Classification): Within the experiment, for each HaaSS sensor report made by participants, the computed H score was subject to automatic classification if the probability value hit the default defined upper ($>.85$) or lower ($<.1$) threshold in *Cogni-Sense*. However, unlike in case study one where two HaaSS sensor reports for attacks 1.1 and 3.1 were automatically classified based on a 92% H score, during the experiment for case study 2 no reports qualified for automatic classification and therefore all HaaSS sensors reports were sent to the semantic attack sandbox for manual classification.

In order to evaluate the practicality of the *Cogni-Sense* semantic attack sandbox, we invited an expert reviewer to manually classify each of the HaaSS reports received in case study 2. The expert reviewer recruited for this task was a lead security operations centre engineer, with over ten years experience working as an information security practitioner, working specifically with security event and information monitoring platforms. Here the aim is to evaluate experimentally whether information

Figure 4.20: HaaSS sensor report of typosquatting website received during case study 2



Report No. 66

Reporting User ID: 48

HaaSS Score [67%]

Report Date/Time: 29/05/2017, 10:32

Report Details/Comments:

While navigating to youtube.com I was redirected to scam website. It appears to be "financial" site which asks user for an email address to complete registration. The risk has been mitigated by closing the browser window.

supplied by the *Cogni-Sense* system and HaaSS sensors adequately informed accurate classification by report reviewers.

Overall, the expert reviewer classified each of the HaaSS reports with a high degree of accuracy and excellent precision to distinguish between HaaSS reports that were credible semantic attacks. According to the performance indicators, the semantic attack sandbox has proven its utility as an informative tool for manually classifying HaaSS reports in order to transform HaaSS attack detection into kinetic defence against the reported threats. In the case of the *Cogni-Sense* prototype, each correct detection would have resulted in an e-mail attack alert of the HaaSS report to HaaSS sensors, however in a production system the security enforcement module (SEM) could be expanded to blocking URLs, domains, file names for example, within an organisational security platform (e.g., proxy, anti-virus, firewall etc.).

Expert reviewer role	Accuracy	Precision	FP	FN
Security operations centre engineer	.87	.94	.04	.09

Table 4.13: Expert reviewer semantic attack sandbox report classification performance

HaaSS Score remodelling: During the course of both case study experiments the *Cogni-Sense* prototype collected new HaaSS feature data as a result of exploitation records and report classification received via the semantic attack sandbox. In total only 136 new observations were collected, which had no significant change in the HaaSS sensor detection rate and therefore H score remodelling was not carried out. In general, the data arrival frequency from HaaSS sensor reports is relatively low compared to that of network alerts from an IDS, but is likely to gradually increase as a HaaSS sensor-base expands. With this in mind, unlike in streaming prediction models, the H score RF rule-set will remain valid for longer until a significant change in the detection distribution, or until a significant amount of data has been collected with feature expansion (e.g., granularity of existing feature scale and entirely new features).

Conclusion from case study results: Overall, the HaaSS sensors comfortably outperformed all of the technical defences for the attacks evaluated in case study one and the new attacks in case study two. In both experiments every technical defence completely failed to identify the deception vectors in any of the semantic attacks. By exception, whilst the Comodo Antivirus failed to detect the PDF file masquerading on the spear USB as an attack (as it was unable to recognise the deception), it did run the file as part of default product configuration behaviour through a virtual machine sandbox and prevented the executable from opening a browser for the attack landing page. This behaviour effectively mitigated the threat of automatically opening a malicious URL via the system's browser. However, it remains untested whether the VM would prevent a sophisticated malware exploitation in practice, as the execution of the file allowed to run (inside the Comodo VM) and a zero-day exploit could potentially subvert this container. The Avast and AVG anti-virus products also scanned the PDF file masquerading executable, presumably because it was an unsigned executable and this is again default product configuration behaviour conducted in most modern-day anti-virus products. Although, in both cases the scanning found the file to be legitimate and therefore allowed the file to run; as the file contained a system command to directly open a browser to a random URL and domain (emulated attack landing page) this can and should be deemed as malicious code. More importantly, and as expected, was that the cosmetic deception vector of the PDF file masquerading attack was undetectable and could not be prevented by the host system and or any anti-virus product tested in the experiment. For attacks 2.1A, 4.1 and 4.2 none of the technical defences identified any malicious behaviour or blocked any of the attacks from being executed by susceptible HaaSS sensors.

Another important observation from the experiment results is further confirmation that detection efficacy for semantic attacks is influenced by security knowledge and experience with specific user platforms; exemplified by the fact that different HaaSS sensors were able to detect semantic attacks on some platforms, but not others. This result reinforces and confirms the contextual relevance of measuring HaaSS sensor features which are associated to the frequency, duration of activity and computer security training received for specific platforms. It is also equally important that these features do not depend on specific attack knowledge launched on a platform, but experience and knowledge of the platforms themselves and are therefore resilient to new attack vectors in future semantic attacks.

By evaluating the results from our laboratory experiment in case study one and our empirical, real-world experiment in case study two, the concept of the Human-as-a-Security-Sensor as a dynamic security mechanism for detecting semantic attacks, proved to be both viable and superior to a wide range of technical security solutions. Whilst the analysis of available security systems for detecting semantic attacks was not exhaustive, here we have shown how HaaSS sensors can dynamically and accurately detect semantic attacks across a range of disparate platforms, irrespective of the individual attacks vectors used, where typically technical systems tend to be incapable in operating in the same holistic manner. Crucially, we also showed how fairly accurate prediction can be made to determine whether a HaaSS sensor will report a credible attack, or indeed be vulnerable to an attack, and how this prediction in the form of the H score provides a metric to measure the reliability of sensors in a technical system.

4.4 Limitations

The interactive laboratory-based experiment in case study one has a few limitations that must be considered. As participants were primed to the purpose of the role-play experiment, they may have been more vigilant and sensitive to each of the semantic attacks deception vectors than they would have normally been; thus potentially weakening the attacks effect. Conversely, as the experiment involved role-play, this may have also reduced participants' ability to determine whether certain attack interactions were contextually anomalous (e.g., Amazon order confirmation, Job offer phishing e-mail). Nevertheless, in direct comparison with the operation of technical defence systems, a HaaSS sensor in practice can also be purposely employed to search for semantic social engineering attacks in a continuous "online" fashion as part of a security platform within an organisation (e.g., security operations centre) or remote paid service (e.g., cloud-based HaaSS reporting). From this perspective, it would be assumed that a HaaSS sensor is constantly vigilant and therefore actively searching for semantic attacks on platforms that they are accessing. In respect to the experiment itself, limitations were also shown in the evaluation of HaaSS against semantic attacks 2.1 and 2.2 (Facebook IM phishing and video masquerading), due to participants not being exposed to the attack or simply missing it. In practice, it is still unclear how effective such attacks would be, and therefore it is important to investigate this more robustly in future work.

For case study 2, it is possible that the *Cogni-Sense* did not capture the full footprint of participant user (i.e. HaaSS sensor) activity, because the HaaSS sensor activity collection tool was not installed on all platforms that the participants may have been using (e.g., smart phone, work devices etc.). By including all participant devices, this would increase the accuracy of the activity analysis of each HaaSS sensor and would have an impact on the H scores computed for each attack report; however this was highly impractical for the experiment due to ethics approval and substantial cross-platform development required for the prototype. Furthermore, as the activity learning period was for one month only, this could have also limited the ability to get a fuller and more accurate picture of user activity. For future experiments a wider footprint for user activity collection (by installing across a range of devices and synchronsing data through cloud storage, for example), as well a longer learning period would improve the accuracy of feature generation for user activity. In principle, the longer the learning period a HaaSS system has for each HaaSS, the more accurate measurements will be for HaaSS activity for feature generation and consequently more confident the H scores. Regarding the prototype HaaSS platform *Cogni-Sense*, implementation limitations were identified by the difficulty in reporting attack 5.1, which initiates a semantic attack distributed as hardware with software interaction (i.e., MD3, see 2.1). For HaaSS sensors that detected the USB as a semantic attack in the physical space, this required HaaSS sensors to take an actual picture of the USB and then send this image to their device to initiate a report; naming the image appropriately for the HaaSS score to determine the frequency and duration in which they used USB devices (specific user-computer interface/platform) and removable media (user-interface/platform type). Therefore, for future HaaSS system platforms it would be essential to allow for capturing reports more naturally in physical space through use of mobile devices video and image capture capabilities.

4.5 Expanding the User Susceptibility Feature-Set: Identification of future practical HaaSS features based on our work

Through our early experiments we identified high-level user susceptibility features that provided a reasonably accurate insight into whether users are likely to detect or be susceptible to social engineering attacks; where we have used these features to form the basis of the HaaSS (H) score. As a result of our empirical experiment, we identify further practical user features that can be measured ethically, in real-time and automatically, but which would also require further experimentation to determine their practical value; such as the ease of collection versus their measurable improvement to H score accuracy.

- **HaaSS report history**. Report history provides a performance-based review of a HaaSS sensor's reliability over a defined time period and across different user-interfaces, platforms and systems. This type of feature can be used to pinpoint a HaaSS sensor's accuracy for detecting different types of attacks, so as to establish which platforms and interfaces they are most suited for and also their typical susceptibility to such threats (for example if they have been subject to emulated attacks as part of HaaSS sensor validation training). As we have shown in case study two, HaaSS report history can be used to identify the performance of a sensor's detection across different platforms, which can be used to profile individual sensors.

- ***Cogni-Sense* SEM semantic attack alerts**. Whilst we have demonstrated the ability to generate semantic attack alerts using the *Cogni-Sense* security enforcement module (SEM) functionality, we have not tested the impact that attack alerts have on the detection efficacy of HaaSS sensors. As a feature, attack alerts can be measured in two ways, their direct effect on HaaSS sensors reporting the same or similar type of attack which has been included in an alert, and the general influence it has on HaaSS sensors' future detection of disparate semantic attacks. For example, were attack alerts (which are confirmed HaaSS reports of a semantic attack) attributed to HaaSS sensors as part of a reputational system, introduction of gamification and competition may have an influence on HaaSS sensors detection prevalence; irrespective of whether this would result in more true or false positive reports.

- **Attack Emulation**. Repeated exposure to semantic attacks and in particular exploitation has been shown to improve the ability for users to detect them [299]. In case study two, 82% of HaaSS sensors were exploited at least once by a semantic attacks, with 31% falling victim to three or more attacks. By recording a HaaSS sensor's specific attack exposure, for example through attack emulation campaigns (which amounts to active awareness training), this feature could be used as predictor of whether the sensor is likely to predict a specific type of attack on a specific type of platform. From an exposure perspective, HaaSS response to emulated attacks can be provided as part of mandatory/validation training or as serious gaming for subscribers (e.g., in the context of home users outside a controlled organisational setting).

- **Passive *H* Score Variance and distribution**. During the experiment, participant HaaSS scores for different platforms would have varied depending on the change in their activity and any updates that are supplied to the system for their computer security training and self-efficacy features. Passive H scores were not measured during the experiment, but in future work could be generated periodically for different platforms of interest and recorded substantively as they change based on a defined threshold (e.g., significant increase/decrease in H score) or trigger (e.g., received security training, updated self-efficacy features). Passive generation of H scores outside of attacks reports can be used to generate a new feature based on variable changes in a HaaSS sensor's H score, which in turn can be used as a forecast of the expected performance of a HaaSS sensor under specific conditions (e.g., attack exposure on specific platforms).

Furthermore, it could also be used to compare with the HaaSS report history feature to identify interactions that more accurately predict future detection performance.

- **Frequency and duration context time-lining**. In case study one and two we have only measured platform activity on one device, on a rolling window of 31 days as the frequency and duration threshold. However, nowadays HaaSS sensors are likely to switch their activity across a range of devices, applications and services (e.g., social media and apps), where activity recognition will fluctuate from month to month. For example, a HaaSS sensor may access a social media platform daily in one month, from their mobile and laptop devices, but in the next month only access the platform once on their laptop and continue to use it daily on their mobile. Therefore, generating multiple features associated to frequency and duration of platform access across different devices provides a more detailed insight into user activity, also providing insight as to whether the device context (and related user-interface) plays a role in detection performance. Aside from features, a cloud-based model to synchronize activity measurements could be utilised in this respect to store activity information on a secure storage environment that can be automatically analysed without requiring to access the device directly, or on demand when receiving data from attack reports; this approach would also aid passive H score analysis.

- **Validated security training and integration with HaaSS**. Whilst we have measured high-level features associated to computer security training within the prototype HaaSS platform developed, it is important to know more deeply how specific training material improves HaaSS sensor detection over other types of training. This is an on-going research problem within the field of semantic attack detection and prevention. Nevertheless, integration of computer security training platforms can be used as a robust way of validating security training and ensuring its measurement is kept up to date. In this way, validated security training would be a feature that is more "trusted" than self-reported training which is not validated or not-compliant with the HaaSS systems quality criteria; irrespective of whether it can be measured and confirmed externally. In fact, further potentially useful security features would be those that represent computer training on platforms proven to work for improving semantic attack detection efficacy [25, 26]. Using this concept, HaaSS sensor status could be validated as a time-based qualification if the user undertakes approved training and passes some mandatory testing.

- **Device / System cyber trustworthiness**. In this project we have focused on measuring the reliability of HaaSS sensor detection as a mechanism to automatically classify and prioritise responses to credible semantic attack threats. However, as HaaSS sensors are likely to use a range of computer devices and systems to report semantic attacks it is important to understand how the integrity of the reporting source, as a combination of HaaSS sensor and device, affects a report's overall credibility. For example, identifying whether a mobile device's "cyber trustworthiness" state differs to that of a desktop device, based on security posture and location may to serve as a factor of the overall reliability of a HaaSS report [337]. This is especially pertinent given the variable security controls that users install on their systems, where often mobile devices are not afforded as much attention.

- **Date, Time**. The date and time in which a HaaSS sensor report can help to indicate the likelihood of HaaSS report credibility, especially during specific times of the year that hackers and cyber criminals select as vulnerable, such as public holidays or during international crisis. It can also indicate whether or not the HaaSS sensor detection efficacy is affected by the date and time that an attack might be observed. In case study two, attacks 1.1 and 5.1 (spear phishing job advertisement and USB) were issued during a weekday in the morning, with attacks 2.1 and 3.1 issued on a weekday in the evening and attack 4.1 issued on a weekend in the evening. As all participants were based in the UK during the time of the experiment, reports were based on a London, UK GMT time zone.

Analysis of the results showed that 58% of HaaSS sensors were exploited in the weekday on an evening, with 82% being exploited on a weekday. Further, 78% of exploitation's on weekends also occurred in the evening. For HaaSS reports, the majority were received in the evening (61% of total reports), but were equally distributed between weekdays and weekends. However, HaaSS report accuracy for reports sent on the weekend and in the evening was 81% (i.e., classified as actual semantic attacks), whereas HaaSS report accuracy for reports received on a weekday in the evening was 68%. By comparison, only 38% of the total reports were received during the morning or afternoon, with the majority of reports received on a weekday (77%). For these weekday reports in the morning and afternoon the report accuracy was 90%, with afternoon reports reporting 93% accuracy.

The results do not represent a robust or rigorous analysis of the effects date and time have on HaaSS sensor detection efficacy, especially as attack exposure would have been determined by the specific HaaSS sensor's lifestyle and usage pattern (e.g., when they looked at the platforms in which the attacks were deployed). However, they do demonstrate the viability of measuring reporting (and exploitation) timestamps as a feature which could be used to highlight if specific HaaSS sensors' detection efficacy is influenced by the day and time.

To a large extent, these results will follow the HaaSS sensors' pattern of life, whereby exposure to attacks will be determined by a sensor typically accessing different computer platforms that they use. However, this is also a useful metric from date and time analysis as it allows for a HaaSS system to potentially predict the time period when it expects HaaSS sensors to be exposed to attacks, based on their individual usage.

Interdependent, time-sensitive features facilitate investigation of whether or not the accuracy and precision of a users' detection efficacy is transient in nature under specific conditions. For instance, were a user to detect an attack on specific time and day, on a specific platform and device, where the frequency and duration of their platform access and security training increased over this period of time compared to past measurements, this time-series data could contain patterns that reveal temporary changes in the user's susceptibility; especially if they were more successful in detecting attacks over this time period than others. Using this approach, susceptibility measurements can be truly dynamic based on users' behavioural and auditable history, rather than training a prediction model on independent sets of user susceptibility samples.

As we have discussed in chapter 3, it is without a doubt that a user's susceptibility to semantic social engineering attacks depends also on personality traits, social context, psychological state and other human and contextual factors, which are impractical to measure in a technical system. Once a HaaSS system is in place and has operated for a prolonged period of time, with access to a wider set of features, such as the temporal types mentioned above, it would be interesting to evaluate to what extent these features inherit traits of the former. It is also important to keep in mind the practicality of measuring different features, which can be limited by environmental deployment of the HaaSS system and its sensor-base. For example, a public crowd-sourced HaaSS system would likely have less integration into HaaSS sensors systems and activity (making some features unavailable or less practical to measure), whereas a HaaSS system deployed within an organisational environment for users would have access to a wider set of features through deeper system integration. At the same time, unlike a crowd-sourced deployment, there may be reduced visibility outside of the organisational environment for user activity feature collection (e.g., on users' personal devices).

4.6 Conclusion

In this chapter we have proposed a prototype technical framework for utilising Human-as-a-Security-Sensor as a novel means of detecting and mitigating semantic social engineering attacks. The framework proposed provides researchers and developers with an architectural blueprint for the design and development of a technical HaaSS system, where we have then employed it to build a prototype HaaSS system called *Cogni-Sense*; evaluating the prototype system's performance in a real-world experiment. The results have shown that the framework is both a practical and effective tool for developing a HaaSS defence system against semantic attacks, as well as reinforcing the wider applicability of the HaaSS concept of user-driven defence for semantic attacks.

We have put the HaaSS paradigm to the test with two case studies in semantic social engineering attacks, and compared against technical platforms that claim to provide defence against such attacks as well as technical defence systems designed to protect specifically from them.

In this respect, this first evaluation was successful, as the users performed considerably better than all technical defence systems, and the *Cogni-Sense* application developed for leveraging this ability of users proved fit for the purpose. In the case of the second evaluation, as an empirical experiment of HaaSS semantic attack detection, we showed that the application of HaaSS under real-world conditions is indeed viable and practically useful means to dynamically detect semantic attacks.

Chapter 5

Conclusion

5.1 Summary of the semantic social engineering problem in computer systems

Semantic social engineering attacks are a pervasive and existential threat to computer systems, because on any system the user-computer interface is always vulnerable to abuse by authorised users, with or without their knowledge. However, as attacks specifically target the user-computer interface it is particularly difficult for technical defences to identify them. This is because attacks primarily employ cosmetic or behavioural deception vectors and as a result often leave a very small technical footprint that can be analysed by technical systems. This leaves individual technical defences with an extremely limited view of semantic attack vectors through platform-specific implementations, which often mean the defence is only able to address a small number of attack vectors. To harness defence that addresses a wide range of semantic attacks requires extensive integration with many security platforms which is impractical, especially as this often is not feasible even to organisational technology platforms, let alone the home user. For technical defence systems to stand a chance in detecting a wide range of semantic attacks, defence mechanisms would require the ability to interpret both visual and behavioural information, contextually and across multiple user-interface platforms; a condition which makes the human user an attractive candidate to perform detection.

Furthermore, up until recently, semantic social engineering exploitation in computer systems has been limited to traditional Internet communications such as email and website platforms. However, in the Internet of Things the threat landscape includes vehicles, industrial control systems and even smart home appliances. As a result, the effects of a deception-based attack will no longer be limited to cyberspace (stealing information, compromising a system, crashing a web service ... etc.), but can also result in physical impact, ranging from manufacturing plants being damaged, trains and tram signalling disrupted causing death and injury, water treatment plants discharging sewage to damage to a nuclear power plants, or denial of service to a national power grid [48]. Therefore, given the ever expanding landscape, and the inability for technical defences to work dynamically across a wide range of user-interfaces, the impetus of designing semantic attack defence around the human user increases, especially as the user provides many of the primary physical interfaces and mediums between different systems, in a ubiquitously connected world.

5.2 Summary of our contributions

5.2.1 A Taxonomy of Attacks and Survey of Defenses for Semantic Social Engineering Attacks

We have designed a taxonomy of semantic social engineering attacks which introduces a structured baseline for classifying any semantic attack by breaking it down into its components and thus allowing to identify countermeasures that are applicable to a range of different attacks that share a subset of its characteristics. We have complemented this taxonomy with a survey of defence measures, highlighting their suitability against the taxonomy's categories. Our taxonomy provides researchers and developers with a tool that can help facilitate the design of more comprehensive defence mechanisms that can address each criterion of our classification rather than specific attack families. Using this approach, we have simplified the problem space and provided a systematic approach to breaking down a semantic attack to more easily identifying the means for mitigating their impact on a user-computer interface.

As the taxonomy relies on systematic components in semantic attacks, rather than platform specific attributes, the taxonomy is applicable to current and future attacks.

5.2.2 A model of user susceptibility to semantic attacks that can be integrated into technical defence systems

We have developed a model of susceptibility to semantic attacks, based on high-level predictors that can be measured ethically, automatically and in real-time. Unlike models produced in prior susceptibility research, this approach enables our model to be directly integrated into technical defence systems. The model provides a strong baseline with which to train a dynamic system for mitigating semantic attacks, where we have robustly validated its performance on a participant sample that spans a diverse geographical footprint.

5.2.3 A Human-as-a-Sensor technical framework for detecting Semantic Attacks

We have designed a HaaSS framework for building a technical system around computer users as physical sensors of semantic attacks. The framework provides an architectural blueprint to design and develop a fully-functional HaaSS platform, based on a core set of procedural and functional components.

5.2.4 Development of *Cogni-Sense*: a prototype HaaSS platform implementation

We have developed and implemented a prototype HaaSS platform *Cogni-Sense*, based on our HaaSS framework, which we have evaluated experimentally within two case studies. The *Cogni-Sense* prototype system demonstrated the feasibility of capturing and scoring HaaSS attacks reports, through a device-based report interface and cloud-based platform for report classification and response; providing a practical means to evaluate the overall concept of HaaSS and prove its viability in an empirical context.

5.3 Future Work

Human-as-a-Security-Sensor is a relatively new concept in computer security and we have made progress towards the first technical framework in order to enable the design and implementation of HaaSS platforms. However, despite the demonstrable feasibility of the utilising HaaSS as a dynamic means to defend against semantic attacks, there is still a lot of room for further development to make HaaSS a viable option for defence against attacks in the era of the Internet of Everything.

1. We have proposed a technical HaaSS framework and used it to develop a functioning HaaSS system to demonstrate the feasibility of the concept as a defence against semantic attacks. Experimentally, the concept and prototype have been shown to work well in combating the threat, however both remain at a prototype stage and require further development to reach a stage of technology readiness that is suitable for integration into a production system. For example, for a user to report a WiFi Evil Twin attack using Cogni-Sense, it would probably be safer for the user to make a report that is stored offline until they are connected to a secure access point, before sending the attack report. So, developing an offline attack reporting function when no network connectivity is available would be highly useful as the loss of reports would be minimised; instead stored and queued for batch upload when network connectivity is established/restored. Furthermore, in order to maintain the integrity of HaaSS reporting cryptographic watermarking and supplementing HaaSS scoring with both user and device susceptibility posture offers options for better guaranteeing the source and identity of the reporting sensor.

2. Whilst we have proven the concept of HaaSS for conventional desktop systems, in the space-constrained interfaces of smartphones and embedded systems, the user is afforded a lot less information to spot suspicious activity. For instance, it is difficult to see the full address of a website. Also, SMS messages in modern smartphones are automatically grouped within one's existing conversations

based on the phone number of the sender. So, anyone using a freely available mobile number spoofer can insert themselves into an existing SMS discussion between two smartphone users and masquerade as one of the two. As a result, Cogni-Sense HaaSS sensor reporting mechanisms should be expanded to include a range of system and device interfaces; especially as future semantic attacks will be designed to exploit IoT devices that interface with users through both physical and cyber means. By providing a richer facility for users to report suspected threats across a multitude of mediums (e.g., pictures, videos, files, audio, text ... etc.), Cogni-Sense can be expanded to address an even wider range of cyber-physical semantic attack threats that are expected to emerge in the near future.

3. The trustworthiness of HaaSS reporting information has been studied in relation to the reliability of human sensors of semantic attacks in the context of this work. However, malicious modification, prevention or delay of HaaSS reports can also be the result of cyber security breaches affecting the mobile devices and network infrastructure used to deliver HaaSS reports. Examples of these can be denial of service attacks, where the timely delivery of reports is important, and location spoofing attacks, where the accuracy of the location of an incident is important. Future work should aim to introduce the cyber-trustworthiness aspect in HaaSS and propose a mechanism for scoring reports in terms of their cyber-trustworthiness based on features of the HaaSS reporting device. Whilst we have conducted some preliminary research in a mobile device "cyber trustworthiness" in [337], combining both approaches explores the concept of a unified measure of trust and reliability for HaaSS reports based on both the reliability of the HaaSS user and cyber-trustworthiness of their system.

4. Our primary aim in this work has been to evaluate the concept of Human-as-a-Security-Sensor for detecting and mitigating semantic attacks. However, more generally, the concept of Human-as-a-Sensor (HaaS) has practical uses beyond semantic attacks. For example, human as a sensor has been used extensively and

successfully for the detection of threats and adverse conditions in physical space [338, 339, 295]. It would be interesting to investigate to what degree human-as-a-sensor can be used for detection of other threats in computer science such as denial of service, reporting service interruption and degradation to determine user experience for improving a systems quality of service in real-time or even further exploring the concept of the H score as a measurement criteria for "cyber insurance" premiums.

5.4 Final Remark

Semantic social engineering attacks are an existential threat to computer systems. As technology evolves, the landscape for attacks constantly shifts and containing the threat is becoming a greater and more complex task than ever before. The emergence of the Internet of Things will enable semantic attacks to thrive, further exacerbating the threat as it bridges into a cyber-physical domain - where deception in cyber space has direct impact in physical space. Fundamentally, building lasting and practical defenses against semantic attacks is a perpetual challenge, for which technical defences are simply not equipped to solve on their own.

The purpose of this thesis was to identify a dynamic and holistic approach to the challenge of detecting semantic social engineering attacks, by involving users as human sensors at the heart of a technical defence platform. In doing so, we have challenged the concept that users are the weakest link against semantic attacks, and instead empowered them to become a human firewall against cyber deception in the Internet age.

Appendix A

Chapter 3 Experiment Surveys and Exhibit Tests

A.1 Experiment 1

Susceptibility to Social Engineering Attacks

P1 PARTICIPANT CONSENT & INFORMATION

This survey is part of a research project on the identification of user susceptibility to social engineering attacks on computer systems. By clicking "Yes" the you agree to the following terms:

I understand that I am free to withdraw from this study:

at any time (until 31/09/2015 as this will no longer be possible, which I have been told) without giving a reason for withdrawing (for students:) without affecting my future with the University and without affecting my grades. I understand that my research data may be used for a further project in anonymous form, but I am able to opt out of this if I so wish, by clicking "Opt Out" below. I can confirm that I am 18 years of age or older.

- ❍ Opt Out

- ❍ Yes
- ❍ No

PARTICIPANT INFORMATION

Information technology is used so pervasively in our society that it is increasingly difficult to identify when and where it is possible for a social engineering attack to occur and more importantly what one would look like. With this survey, we will try to identify the parameters that are the most relevant in predicting what makes a user susceptibility to social engineering attacks, in front of different, often unfamiliar user interfaces, on desktop computers and smartphones. The survey starts with a few questions about yourself and continues with a test containing a number of exhibit based questions. It should take no longer than 10 minutes or so to complete. Please ensure you are using a Desktop or Laptop, or 8inch or larger tablet device when taking this survey, as some images may appear distorted or obscured from view.   Researcher's contact details (including telephone number and e-mail address):   Room QM365, University of Greenwich, Old Royal Naval College, SE10 9LS, UK E-mail: hr07@gre.ac.uk   Tel: +44 (0)20 8331 8531

Q1 Age: What is your age?
❍ 18-24 years old
❍ 25-34 years old
❍ 35-44 years old
❍ 45-54 years old
❍ 55-64 years old
❍ 65+

Q2 Gender: What is your sex?
❍ Male
❍ Female

Q3 Education: What is the highest level of education you have completed (This should NOT include what you are currently studying)?
❍ Less than High School
❍ High School / GED
❍ Some College
❍ Trade/technical/vocational training (2 year)
❍ Associate degree
❍ Bachelor's degree
❍ Master's degree
❍ Doctorate degree

Q4 Where 0=Beginner and 100=Expert, for each category what is your approximate skill level?
______ Computer Literacy
______ Computer Security Awareness

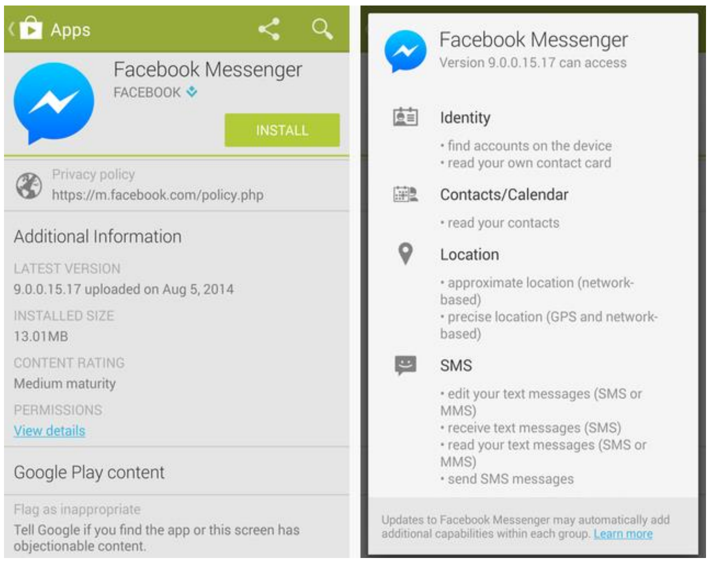
Q5 Have you ever received any computer security training?

	No	Yes
As part of formal education	❍	❍
As part of training at work	❍	❍
As part of self study	❍	❍

The next section is a test of your susceptibility to social engineering attacks. You will be shown 12 cases and you will be asked to determine whether they are examples of computer security attacks or not.This should take no longer than 10 minutes. Incorrectly marking an exhibit as an attack that is NOT an attack will negatively affect your score.Your responses are timed.Scores are displayed at the end of the survey.

E1 EXHIBIT 1: You click on "INSTALL" to download this app from the Googleplay store. Once downloaded, the following permissions are granted to the app.

❍ Most likely not an attack
❍ Most likely an attack



Facebook Messenger
FACEBOOK

Privacy policy
https://m.facebook.com/policy.php

Additional Information

LATEST VERSION
9.0.0.15.17 uploaded on Aug 5, 2014

INSTALLED SIZE
13.01MB

CONTENT RATING
Medium maturity

PERMISSIONS
View details

Google Play content

Flag as inappropriate
Tell Google if you find the app or this screen has objectionable content.

Facebook Messenger
Version 9.0.0.15.17 can access

Identity
• find accounts on the device
• read your own contact card

Contacts/Calendar
• read your contacts

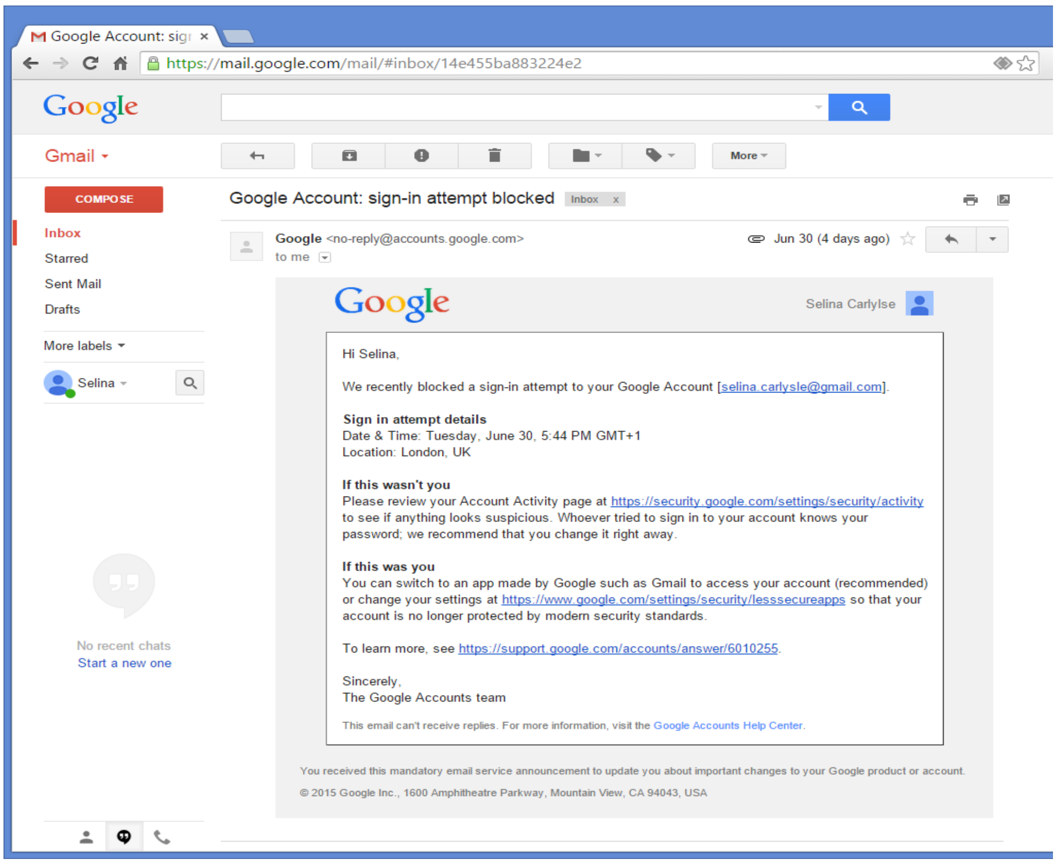
Location
• approximate location (network-based)
• precise location (GPS and network-based)

SMS
• edit your text messages (SMS or MMS)
• receive text messages (SMS)
• read your text messages (SMS or MMS)
• send SMS messages

Updates to Facebook Messenger may automatically add additional capabilities within each group. Learn more

E2 EXHIBIT 2: When logging into GMAIL to check some emails, you see that the following email has been received.

❍ Most likely not an attack
❍ Most likely an attack



Google Account: sign-in attempt blocked Inbox x

Google <no-reply@accounts.google.com>
to me

Jun 30 (4 days ago)

Google

Selina Carlylse

Hi Selina,

We recently blocked a sign-in attempt to your Google Account [selina.carlysle@gmail.com].

Sign in attempt details
Date & Time: Tuesday, June 30, 5:44 PM GMT+1
Location: London, UK

If this wasn't you
Please review your Account Activity page at https://security.google.com/settings/security/activity to see if anything looks suspicious. Whoever tried to sign in to your account knows your password; we recommend that you change it right away.

If this was you
You can switch to an app made by Google such as Gmail to access your account (recommended) or change your settings at https://www.google.com/settings/security/lesssecureapps so that your account is no longer protected by modern security standards.

To learn more, see https://support.google.com/accounts/answer/6010255.

Sincerely,
The Google Accounts team

This email can't receive replies. For more information, visit the Google Accounts Help Center.

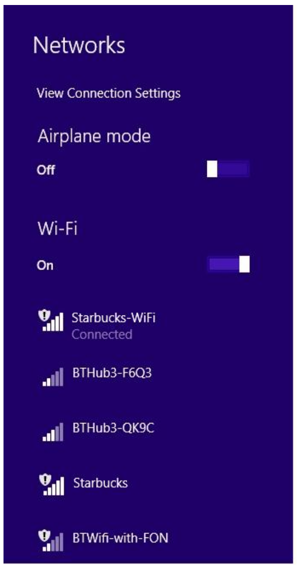
You received this mandatory email service announcement to update you about important changes to your Google product or account.

© 2015 Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA

EXHIBIT 3: Whilst sat inside a Starbucks cafe, you connect to WiFi access point "Starbucks-WiFi" which opens the following login page when attempting to browse the Internet.

❍ Most likely not an attack

❍ Most likely an attack



Networks

View Connection Settings

Airplane mode

Off

Wi-Fi

On

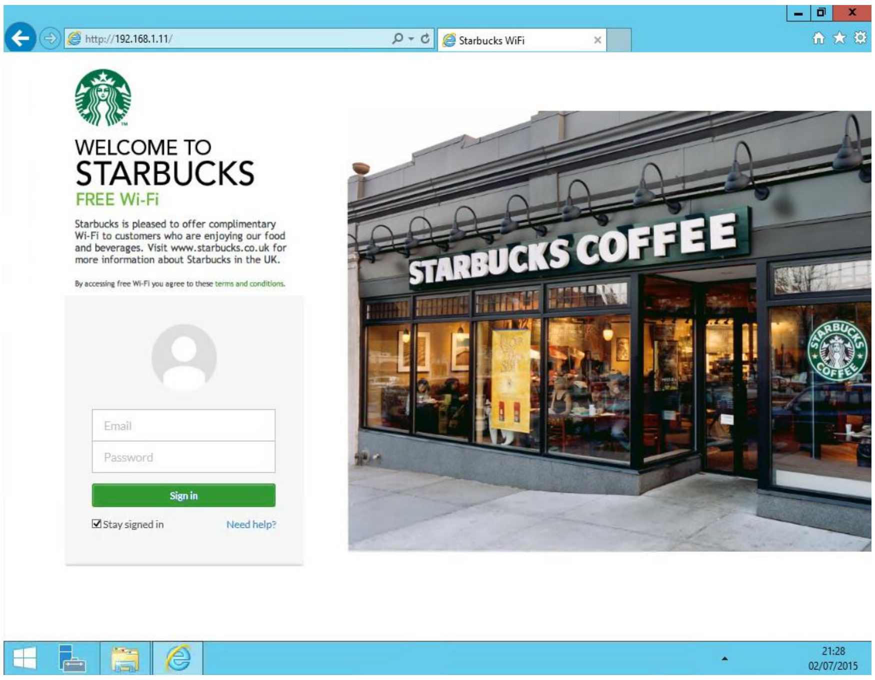
Starbucks-WiFi
Connected

BTHub3-F6Q3

BTHub3-QK9C

Starbucks

BTWifi-with-FON



http://192.168.1.11/

Starbucks WiFi

WELCOME TO
STARBUCKS
FREE Wi-Fi

Starbucks is pleased to offer complimentary Wi-Fi to customers who are enjoying our food and beverages. Visit www.starbucks.co.uk for more information about Starbucks in the UK.

By accessing free Wi-Fi you agree to these terms and conditions.

Email

Password

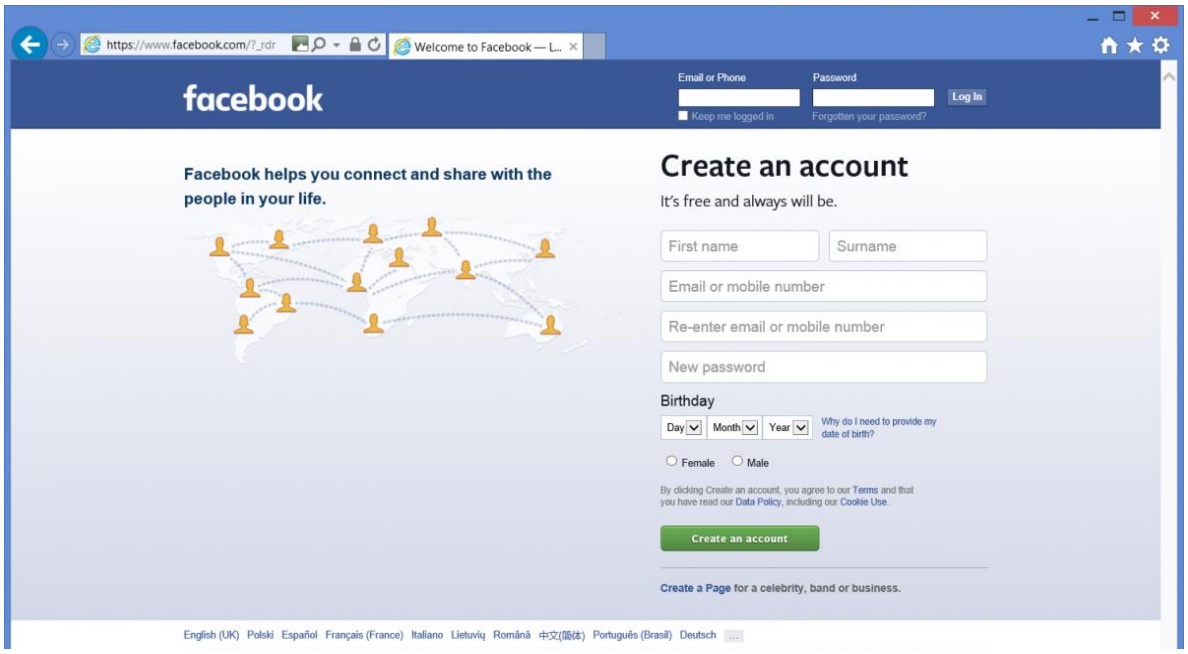
Sign in

Stay signed in   Need help?

EXHIBIT 4: You enter "facebok.com" into Internet Explorer address bar, press enter and the following webpage appears.

❍ Most likely not an attack

❍ Most likely an attack



https://www.facebook.com/?_rdr

Welcome to Facebook — L...

facebook

Email or Phone   Password   Log In

Keep me logged in   Forgotten your password?

Facebook helps you connect and share with the people in your life.

Create an account

It's free and always will be.

First name   Surname

Email or mobile number

Re-enter email or mobile number

New password

Birthday

Day   Month   Year   Why do I need to provide my date of birth?

Female   Male

By clicking Create an account, you agree to our Terms and that you have read our Data Policy, including our Cookie Use.

Create an account

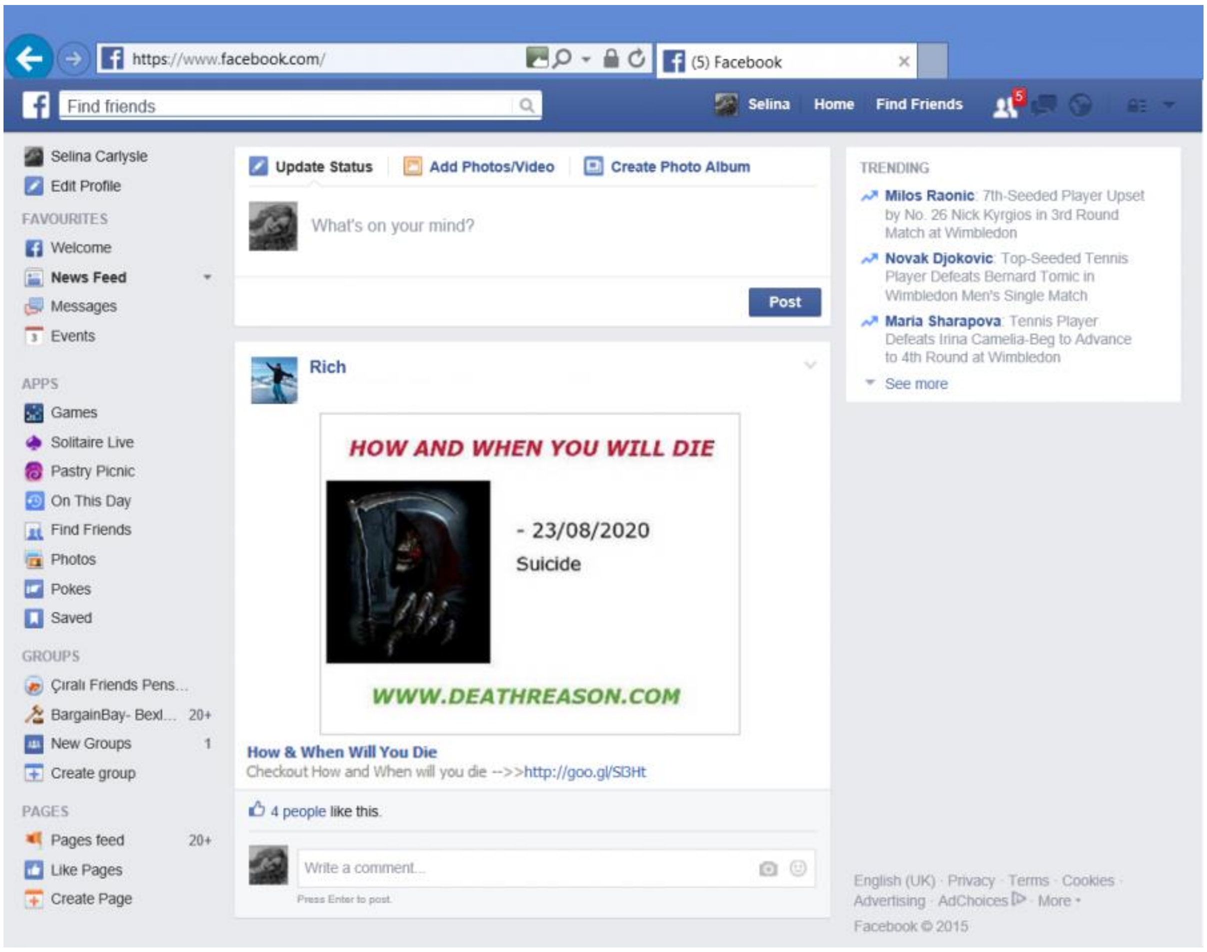
Create a Page for a celebrity, band or business.

English (UK) Polski Español Français (France) Italiano Lietuvių Română 中文(简体) Português (Brasil) Deutsch ...

EXHIBIT 5: You log onto Facebook and you see the following post on your news feed. On clicking on the link in the post the following request appears.

❍ Most likely not an attack

❍ Most likely an attack



Rich

HOW AND WHEN YOU WILL DIE

- 23/08/2020

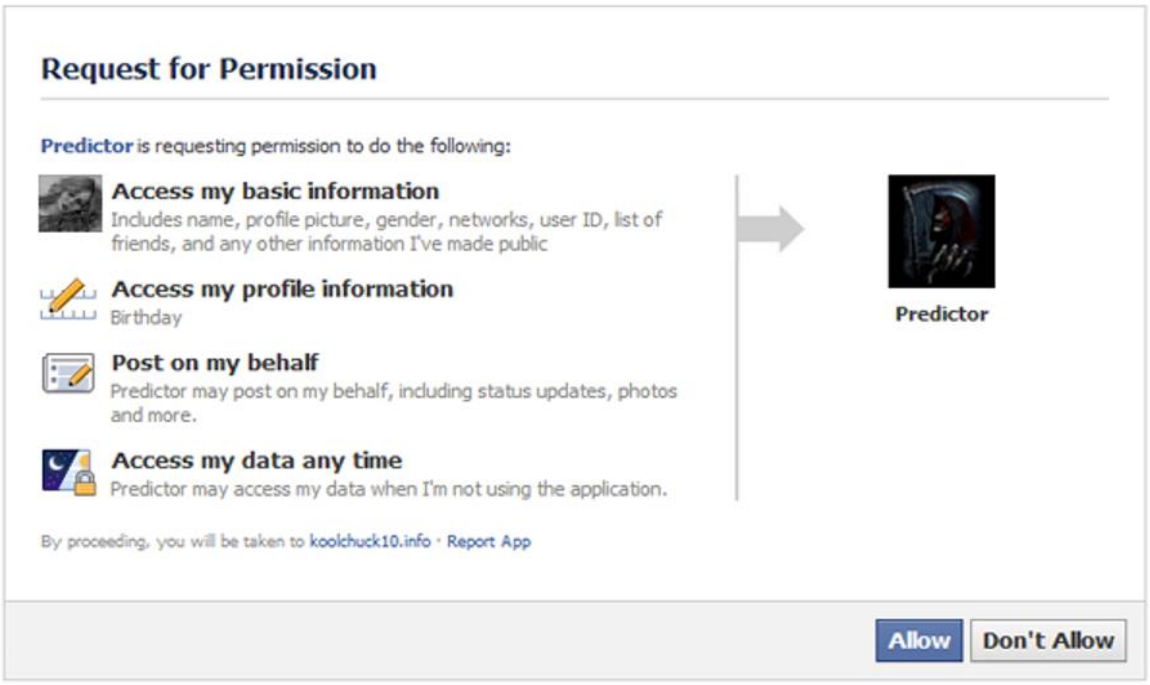
Suicide

WWW.DEATHREASON.COM

How & When Will You Die

Checkout How and When will you die -->>http://goo.gl/SI3Ht

4 people like this.



Request for Permission

Predictor is requesting permission to do the following:

Access my basic information
Includes name, profile picture, gender, networks, user ID, list of friends, and any other information I've made public

Access my profile information
Birthday

Post on my behalf
Predictor may post on my behalf, including status updates, photos and more.

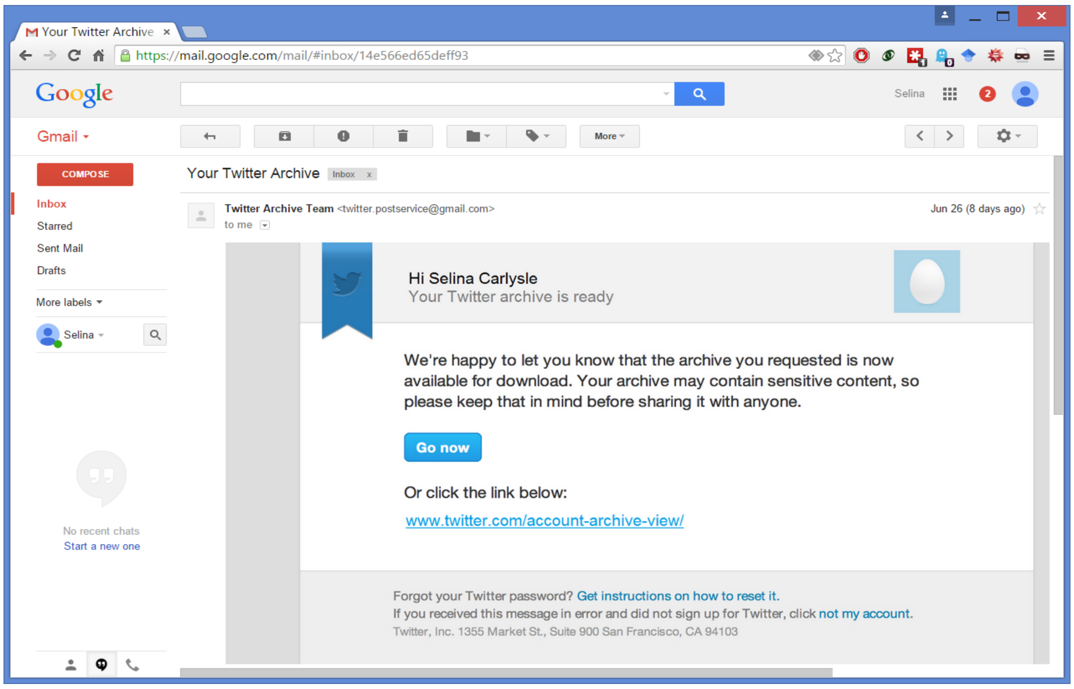
Access my data any time
Predictor may access my data when I'm not using the application.

By proceeding, you will be taken to koolchuck10.info · Report App

EXHIBIT 6: When logging into GMAIL to check some emails, you see that the following email has been received.

❍ Most likely not an attack

❍ Most likely an attack


Your Twitter Archive

https://mail.google.com/mail/#inbox/14e566ed65deff93

Your Twitter Archive Inbox

Twitter Archive Team <twitter.postservice@gmail.com>
to me

Jun 26 (8 days ago)

Hi Selina Carlyle
Your Twitter archive is ready

We're happy to let you know that the archive you requested is now available for download. Your archive may contain sensitive content, so please keep that in mind before sharing it with anyone.

Go now

Or click the link below:

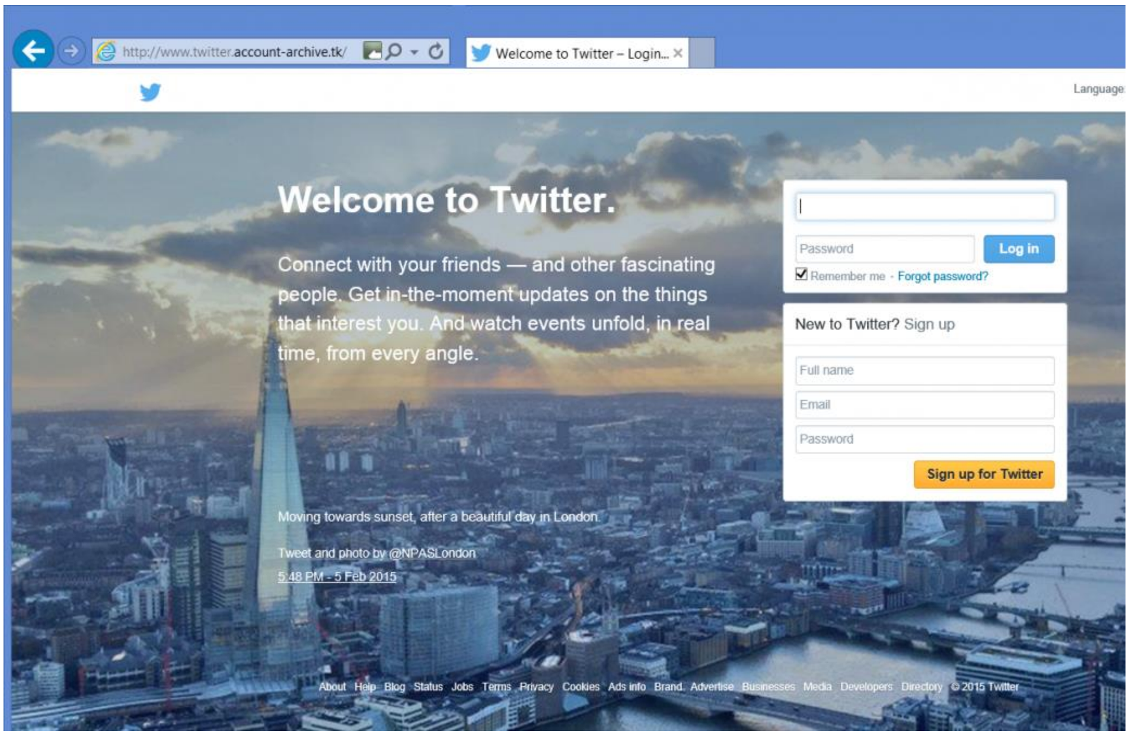
www.twitter.com/account-archive-view/

Forgot your Twitter password? Get instructions on how to reset it.
If you received this message in error and did not sign up for Twitter, click not my account.
Twitter, Inc. 1355 Market St., Suite 900 San Francisco, CA 94103

EXHIBIT 7: By clicking on the link in the previous email, the following webpage appears.

❍ Most likely not an attack

❍ Most likely an attack


http://www.twitter.account-archive.tk/

Welcome to Twitter – Login...

Welcome to Twitter.

Connect with your friends — and other fascinating people. Get in-the-moment updates on the things that interest you. And watch events unfold, in real time, from every angle.

Password
Log in
Remember me · Forgot password?

New to Twitter? Sign up

Full name
Email
Password
Sign up for Twitter

Moving towards sunset, after a beautiful day in London.

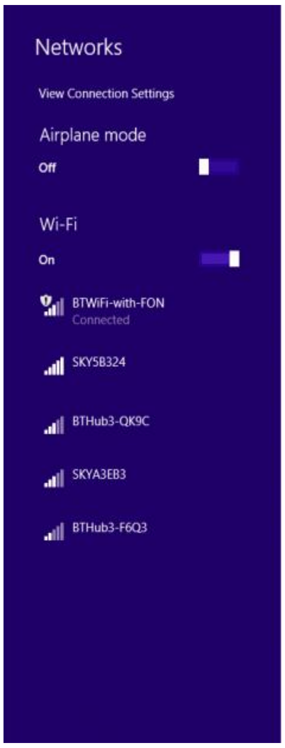
Tweet and photo by @NPASLondon

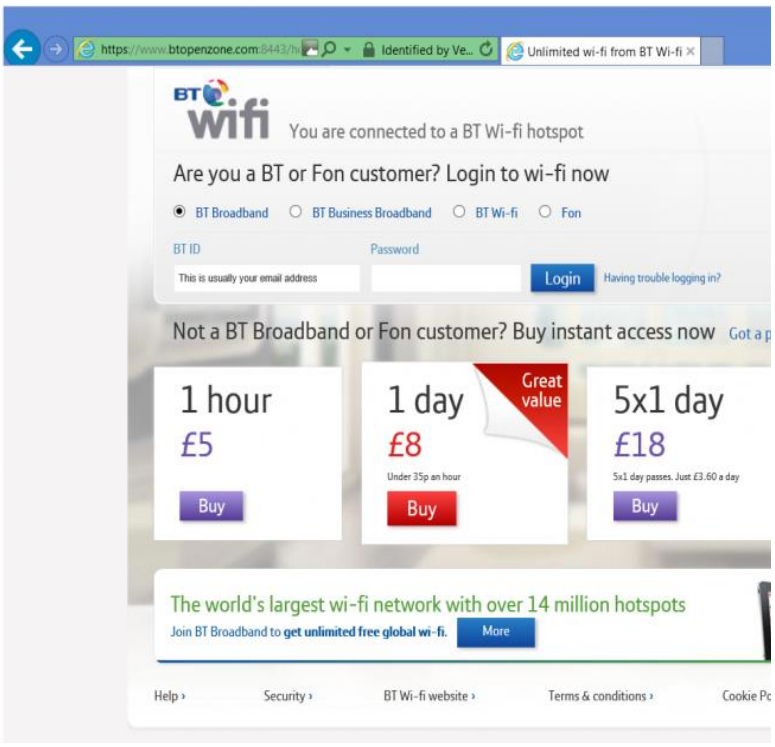
5:48 PM - 5 Feb 2015

Language
About Help Blog Status Jobs Terms Privacy Cookies Ads info Brand Advertise Businesses Media Developers Directory © 2015 Twitter

EXHIBIT 8: Whilst waiting at a train station, you connect to WiFi access point "BTWiFi-with-FON" which opens the following login page when attempting to browse the Internet.

- ❍ Most likely not an attack
- ❍ Most likely an attack




https://www.btopenzone.com:8443/h

Identified by Ve...

Unlimited wi-fi from BT Wi-fi

BT wifi You are connected to a BT Wi-fi hotspot

Are you a BT or Fon customer? Login to wi-fi now

● BT Broadband ○ BT Business Broadband ○ BT Wi-fi ○ Fon

BT ID: This is usually your email address

Password:

Login Having trouble logging in?

Not a BT Broadband or Fon customer? Buy instant access now

1 hour £5 Buy

1 day £8 Under 35p an hour Buy — Great value

5x1 day £18 5x1 day passes. Just £3.60 a day Buy

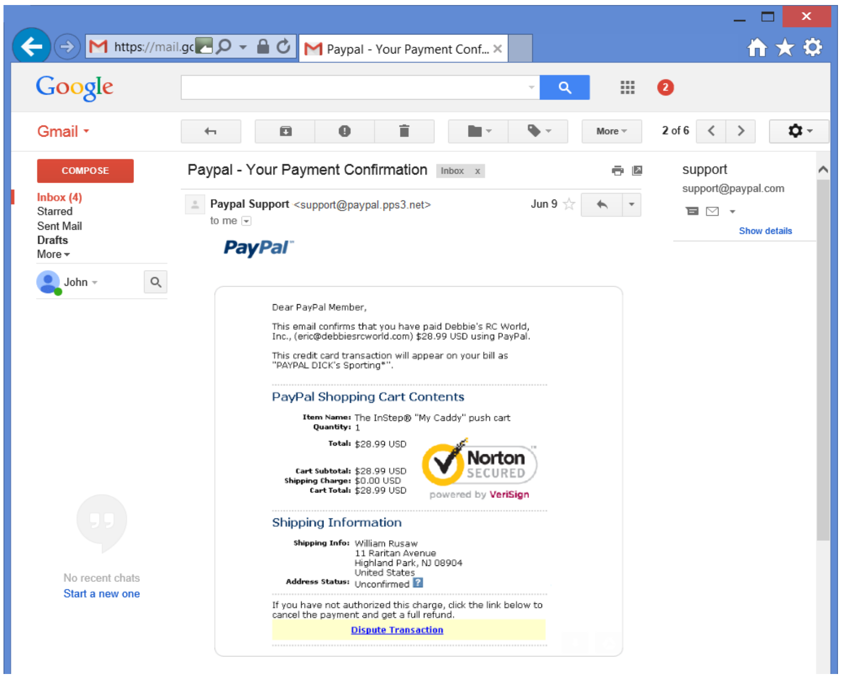
The world's largest wi-fi network with over 14 million hotspots

Join BT Broadband to get unlimited free global wi-fi. More

Help › Security › BT Wi-fi website › Terms & conditions › Cookie Pc

EXHIBIT 9: When logging into GMAIL to check some emails, you see that the following email has been received.

- ❍ Most likely not an attack
- ❍ Most likely an attack


https://mail.gc

Paypal - Your Payment Conf...

Gmail: COMPOSE, Inbox (4), Starred, Sent Mail, Drafts, More, John

No recent chats. Start a new one

2 of 6

Paypal - Your Payment Confirmation (Inbox)

Paypal Support <support@paypal.pps3.net> to me — Jun 9

support — support@paypal.com — Show details

PayPal

Dear PayPal Member,

This email confirms that you have paid Debbie's RC World, Inc., (eric@debbiesrcworld.com) $28.99 USD using PayPal.

This credit card transaction will appear on your bill as "PAYPAL DICK's Sporting*".

PayPal Shopping Cart Contents

Item Name: The InStep® "My Caddy" push cart
Quantity: 1
Total: $28.99 USD

Cart Subtotal: $28.99 USD
Shipping Charge: $0.00 USD
Cart Total: $28.99 USD

Norton SECURED powered by VeriSign

Shipping Information

Shipping Info: William Rusaw
11 Raritan Avenue
Highland Park, NJ 08904
United States
Address Status: Unconfirmed

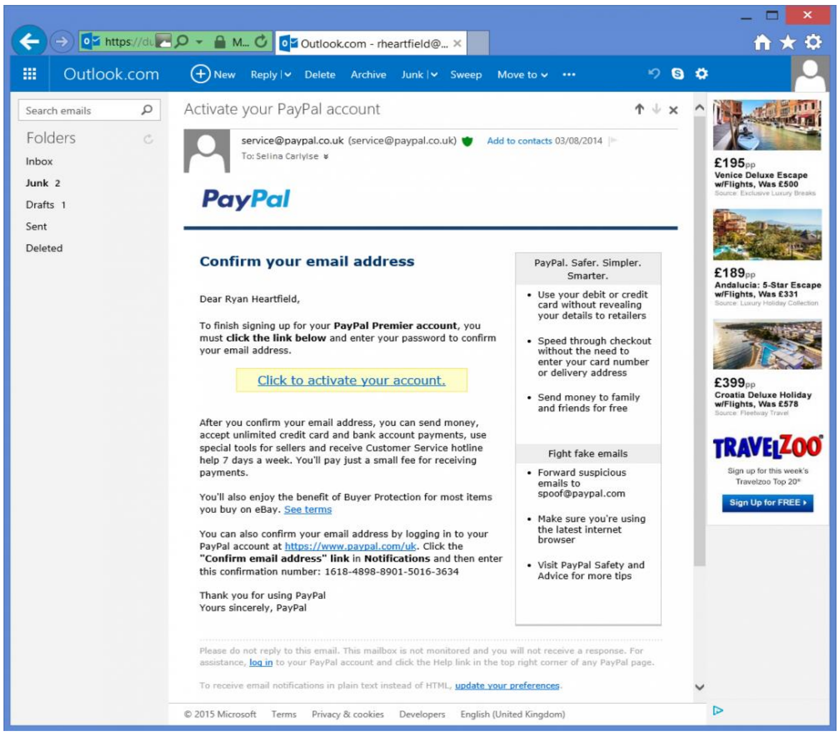
If you have not authorized this charge, click the link below to cancel the payment and get a full refund.

Dispute Transaction

EXHIBIT 10: When logging into OUTLOOK to check some emails, you see that the following email has been received.

❍ Most likely not an attack

❍ Most likely an attack



Outlook.com - rheartfield@...

Activate your PayPal account

service@paypal.co.uk (service@paypal.co.uk) Add to contacts 03/08/2014

To: Selina Carlyise

PayPal

Confirm your email address

Dear Ryan Heartfield,

To finish signing up for your **PayPal Premier account**, you must **click the link below** and enter your password to confirm your email address.

Click to activate your account.

After you confirm your email address, you can send money, accept unlimited credit card and bank account payments, use special tools for sellers and receive Customer Service hotline help 7 days a week. You'll pay just a small fee for receiving payments.

You'll also enjoy the benefit of Buyer Protection for most items you buy on eBay. See terms

You can also confirm your email address by logging in to your PayPal account at https://www.paypal.com/uk. Click the **"Confirm email address" link** in **Notifications** and then enter this confirmation number: 1618-4898-8901-5016-3634

Thank you for using PayPal
Yours sincerely, PayPal

PayPal. Safer. Simpler. Smarter.

- Use your debit or credit card without revealing your details to retailers
- Speed through checkout without the need to enter your card number or delivery address
- Send money to family and friends for free

Fight fake emails

- Forward suspicious emails to spoof@paypal.com
- Make sure you're using the latest internet browser
- Visit PayPal Safety and Advice for more tips

Please do not reply to this email. This mailbox is not monitored and you will not receive a response. For assistance, log in to your PayPal account and click the Help link in the top right corner of any PayPal page.

To receive email notifications in plain text instead of HTML, update your preferences.

© 2015 Microsoft Terms Privacy & cookies Developers English (United Kingdom)

Outlook.com
New
Reply
Delete
Archive
Junk
Sweep
Move to
Search emails
Folders
Inbox
Junk 2
Drafts 1
Sent
Deleted
£195pp
Venice Deluxe Escape w/Flights, Was £500
£189pp
Andalucia: 5-Star Escape w/Flights, Was £331
£399pp
Croatia Deluxe Holiday w/Flights, Was £578
TRAVELZOO
Sign up for this week's Travelzoo Top 20
Sign Up for FREE

EXHIBIT 11: Scanning the following QR code opens the following website.

❍ Most likely not an attack

❍ Most likely an attack

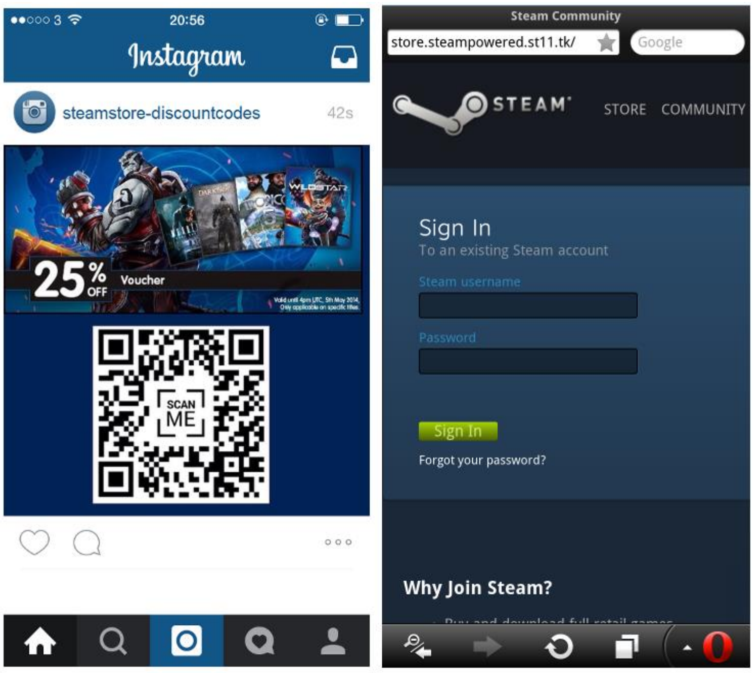
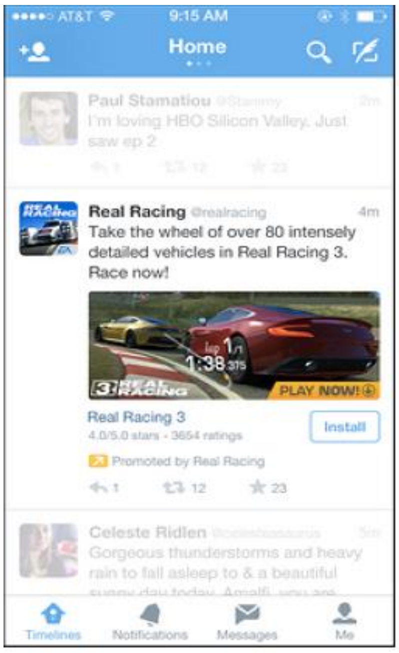


EXHIBIT 12: The following Twitter post is clicked on.

❍ Most likely not an attack

❍ Most likely an attack



AT&T 9:15 AM

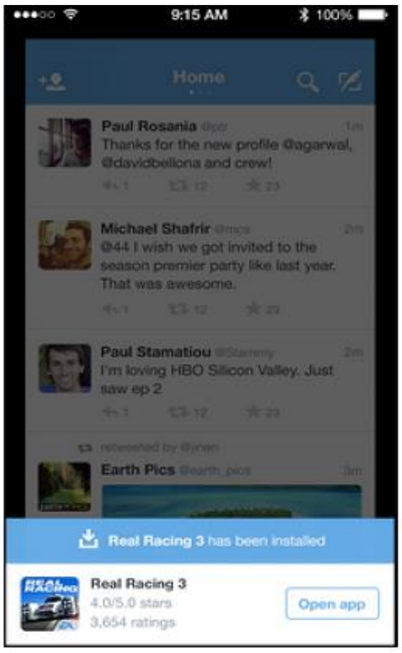
Home

Paul Stamatiou @Stammy 2m
I'm loving HBO Silicon Valley. Just saw ep 2
1 12 23

Real Racing @realracing 4m
Take the wheel of over 80 intensely detailed vehicles in Real Racing 3. Race now!

Real Racing 3
4.0/5.0 stars - 3654 ratings
Install
Promoted by Real Racing
1 12 23

Celeste Ridlen 5m
Gorgeous thunderstorms and heavy rain to fall asleep to & a beautiful sunny day today. Amalfi, you are



9:15 AM 100%

Home

Paul Rosania 1m
Thanks for the new profile @agarwal, @davidbellona and crew!
1 12 23

Michael Shafrir 2m
@44 I wish we got invited to the season premier party like last year. That was awesome.
1 12 23

Paul Stamatiou @Stammy 2m
I'm loving HBO Silicon Valley. Just saw ep 2
1 12 23

Earth Pics @earth_pics 3m

Real Racing 3 has been installed

Real Racing 3
4.0/5.0 stars
3,654 ratings
Open app

Q6 Frequency: How often do you use the following?

	Never	Less than once a month	Once a month	Weekly	Daily
Email	❍	❍	❍	❍	❍
Instant Messaging	❍	❍	❍	❍	❍
Public WiFi	❍	❍	❍	❍	❍
Social Media	❍	❍	❍	❍	❍
Search Engine	❍	❍	❍	❍	❍

Q7 Duration: For each frequency interval you chose in the previous question, on average what is the cumulative duration of time you spend using the following (e.g. Instant Messaging: Weekly, 1 to 2 hours)?

	None	Less than 30 mins	30 mins to 1 hour	1 to 2 hours	2 to 4 hours	4+ hours
Email	❍	❍	❍	❍	❍	❍
Instant Messaging	❍	❍	❍	❍	❍	❍
Public WiFi	❍	❍	❍	❍	❍	❍
Social Media	❍	❍	❍	❍	❍	❍
Search Engine	❍	❍	❍	❍	❍	❍

Q8 Familiarity: How familiar are you with using the following?

	Not Very	Somewhat	Very
Facebook	❍	❍	❍
Twitter	❍	❍	❍
WhatsApp	❍	❍	❍
Skype	❍	❍	❍
Gmail	❍	❍	❍
Googleplay	❍	❍	❍
Paypal	❍	❍	❍
Starbucks WiFi	❍	❍	❍
BT WiFi Hotspot	❍	❍	❍
Steam	❍	❍	❍
Google Search	❍	❍	❍
Youtube	❍	❍	❍

Q9 Feedback (Optional): Please provide any thoughts, suggestions or observations you may have after taking this survey.

Share Challenge your friends to beat your score by inviting them to take the survey or sharing the following
link: https://greenwichuniversity.eu.qualtrics.com/SE/?SID=SV_dbcX1BR4kQOH6sJ

A.2 Experiment 2

Susceptibility to Social Engineering Attacks

PARTICIPANT CONSENT & INFORMATION

This survey is part of a research project for testing user susceptibility to social engineering attacks on computer systems. It starts with a few questions about yourself and continues with a test containing a number of exhibit based questions. It should take no longer than 10 minutes or so to complete. By clicking continuing with this survey you agree to the following terms:

I understand that I am free to withdraw from this study at any time without giving a reason for withdrawing. I understand that my research data may be used for a further project in anonymous form, but I am able to opt out of this if I so wish by existing the survey at any time. I can confirm that I am 18 years of age or older.

IMPORTANT

This survey must be taken on a device with a browser that is capable of playing flash videos e.g. Chrome, Firefox, Internet Explorer. Mobile browsers will NOT work. Test scores are displayed at the end of the survey.

Age: What is your age?
______ Years of Age

G What is your gender?
❍ Male
❍ Female
❍ Other

Education: What is the highest level of education you have completed (This should NOT include what you are currently studying)?
❍ Less than High School
❍ High School / GED
❍ Some College
❍ Trade/technical/vocational training (2 year)
❍ Associate degree
❍ Bachelor's degree
❍ Master's degree
❍ Doctorate degree

S1 Please answer the following question as accurately as possible.

	Have you ever received computer security training through the following		When did you last receive this training								How was the training delivered? Tick all that apply				
	No	Yes	NA	1 year +	Up to 1 year ago	Up to 6 months ago	Up to 3 months ago	Up to 1 month ago	Up to 2 weeks ago	In the last week	Lectures	Tests	Projects	Course work	N/A
Formal Education	○	○	○	○	○	○	○	○	○	○	☐	☐	☐	☐	☐

S2 Please answer the following question as accurately as possible.

	Have you ever received computer security training through the following		When did you last receive this training								How was the training delivered? Tick all that apply				
	No	Yes	NA	1 year +	Up to 1 year ago	Up to 6 months ago	Up to 3 months ago	Up to 1 month ago	Up to 2 weeks ago	In the last week	Survey	Videos	Tests	Game	N/A
At work	○	○	○	○	○	○	○	○	○	○	☐	☐	☐	☐	☐

S3 Please answer the following question as accurately as possible.

	Have you ever received computer security training through the following		When did you last receive this training								How was the training delivered? Tick all that apply				
	No	Yes	NA	1 year +	Up to 1year ago	Up to 6 months ago	Up to 3 months ago	Up to 1 month ago	Up to 2 weeks ago	In the last week	Books	Videos	Websites	Game	N/A
Self-study	○	○	○	○	○	○	○	○	○	○	☐	☐	☐	☐	☐

Please answer the following question as accurately as possible. Please answer all columns with NA if no options apply.

	How often do you use the following?					If you use any of the above on a particular day, for how long would it be throughout the day?						When did you last receive computer security training for this platform?								How familiar are you with this type of platform		
	N/A	Less than once a month	Once a month	Weekly	Daily	N/A	Less than 30 mins	30 mins to 1 hour	1 to 2 hours	2 to 4 hours	4 hours +	N/A	1 year +	Up to 1 year ago	Up to 6 months ago	Up to 3 months ago	Up to 1 month ago	2 weeks ago	Within the last week	Not very	Somewhat	Very
Email	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Instant Messaging	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Public WiFi	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Social Media	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Web browsers	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
E-commerce / classifieds websites (e.g., ebay, amazon, gumtree ...etc.)	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

Please answer the following question as accurately as possible.

	How often do you use the following?					If you use any of the above on a particular day, for how long would it be throughout the day?						When did you last receive computer security training for this platform?								How familiar are you with this particular platform		
	N/A	Less than once a month	Once a month	Weekly	Daily	N/A	Less than 30 mins	30 mins to 1 hour	1 to 2 hours	2 to 4 hours	4 hours +	N/A	1 year +	Up to 1 year ago	Up to 6 months ago	Up to 3 months ago	Up to 1 month ago	2 weeks ago	Within the last week	Not very	Somewhat	Very
Gmail	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Steam Community	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Starbucks WiFi	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Facebook	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Twitter	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Youtube	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Microsoft Edge Browser	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍
Gumtree	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍	❍

SA Where 0=Beginner and 100=Expert, for each category what is your approximate skill level?
______ Computer Security Awareness

CL LAST Question before the test! Where 0=Beginner and 100=Expert, for each category what is your approximate skill level?
______ Computer Literacy

The next section is a test of your security awareness to computer-based social engineering attacks

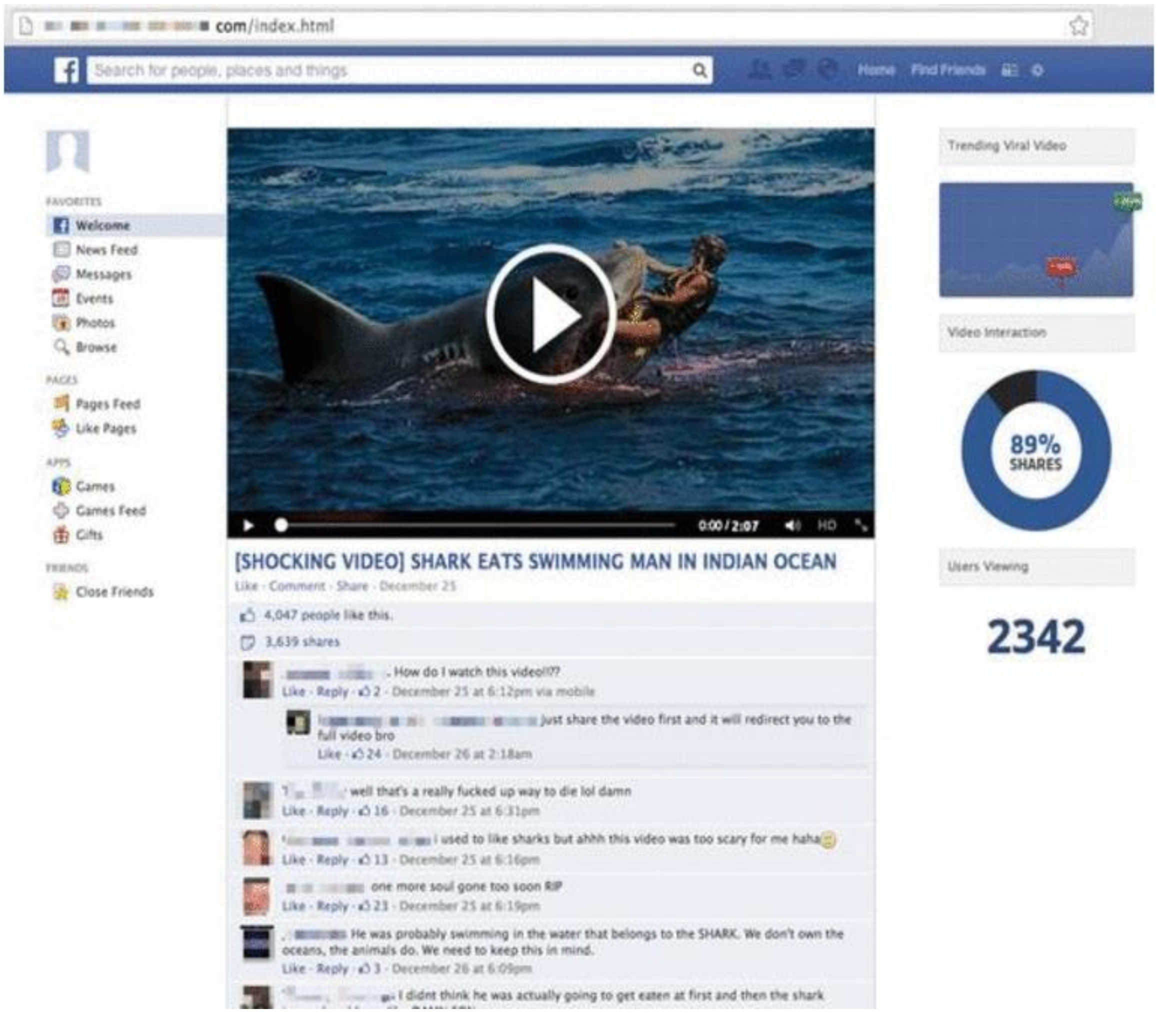
You will be shown 12 cases and you will be asked to determine whether they are examples of computer security attacks or not.

This should take no longer than 10 minutes.

E1 You are logged into Facebook. The following slideshow of images shows a post on your news feed that you have clicked on.

❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence


[redacted] com/index.html

[SHOCKING VIDEO] SHARK EATS SWIMMING MAN IN INDIAN OCEAN
Like · Comment · Share · December 25

4,047 people like this.
3,639 shares

[redacted] How do I watch this video!!??
Like · Reply · 2 · December 25 at 6:12pm via mobile

[redacted] just share the video first and it will redirect you to the full video bro
Like · 24 · December 26 at 2:18am

[redacted] well that's a really fucked up way to die lol damn
Like · Reply · 16 · December 25 at 6:31pm

[redacted] i used to like sharks but ahhh this video was too scary for me haha
Like · Reply · 13 · December 25 at 6:16pm

[redacted] one more soul gone too soon RIP
Like · Reply · 23 · December 25 at 6:19pm

[redacted] He was probably swimming in the water that belongs to the SHARK. We don't own the oceans, the animals do. We need to keep this in mind.
Like · Reply · 3 · December 26 at 6:09pm

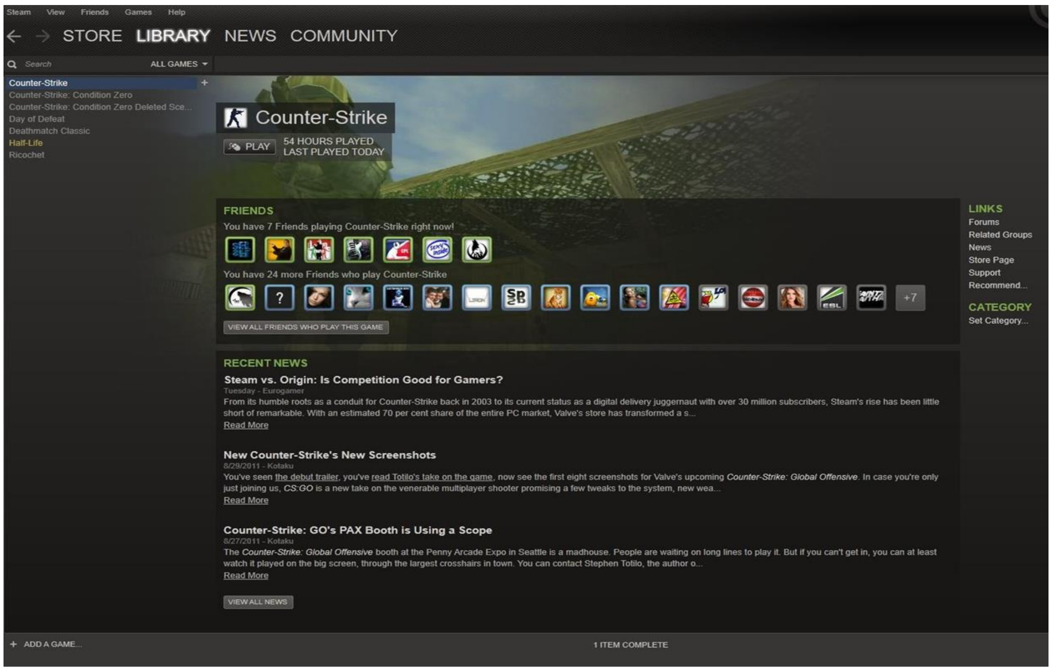
[redacted] I didnt think he was actually going to get eaten at first and then the shark

E2 You are logged into the Steam community website after recently playing Counter-Strike. The following message has been received on your steam instant messaging account.

❍ Most likely "NOT" an attack

❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence



FRIENDS

You have 7 Friends playing Counter-Strike right now!

You have 24 more Friends who play Counter-Strike

VIEW ALL FRIENDS WHO PLAY THIS GAME

RECENT NEWS

Steam vs. Origin: Is Competition Good for Gamers?

Tuesday - Eurogamer

From its humble roots as a conduit for Counter-Strike back in 2003 to its current status as a digital delivery juggernaut with over 30 million subscribers, Steam's rise has been little short of remarkable. With an estimated 70 per cent share of the entire PC market, Valve's store has transformed a s...

Read More

New Counter-Strike's New Screenshots

8/29/2011 - Kotaku

You've seen the debut trailer, you've read Totilo's take on the game, now see the first eight screenshots for Valve's upcoming *Counter-Strike: Global Offensive*. In case you're only just joining us, *CS:GO* is a new take on the venerable multiplayer shooter promising a few tweaks to the system, new wea...

Read More

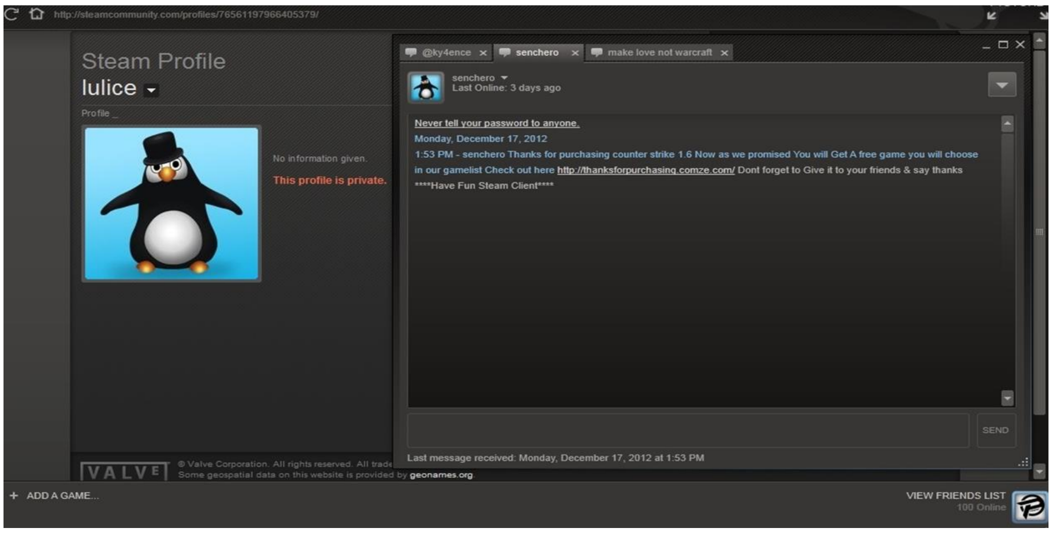
Counter-Strike: GO's PAX Booth is Using a Scope

8/27/2011 - Kotaku

The *Counter-Strike: Global Offensive* booth at the Penny Arcade Expo in Seattle is a madhouse. People are waiting on long lines to play it. But if you can't get in, you can at least watch it played on the big screen, through the largest crosshairs in town. You can contact Stephen Totilo, the author o...

Read More

VIEW ALL NEWS



http://steamcommunity.com/profiles/76561197966405379/
Steam Profile
lulice
Profile
No information given.
This profile is private.
@ky4ence
senchero
make love not warcraft
SEND
VALVE
© Valve Corporation. All rights reserved. All trade
Some geospatial data on this website is provided by geonames.org
ADD A GAME...
VIEW FRIENDS LIST
100 Online

senchero
Last Online: 3 days ago

Never tell your password to anyone.

Monday, December 17, 2012

1:53 PM - senchero Thanks for purchasing counter strike 1.6 Now as we promised You will Get A free game you will choose in our gamelist Check out here http://thanksforpurchasing.comze.com/ Dont forget to Give it to your friends & say thanks

****Have Fun Steam Client****

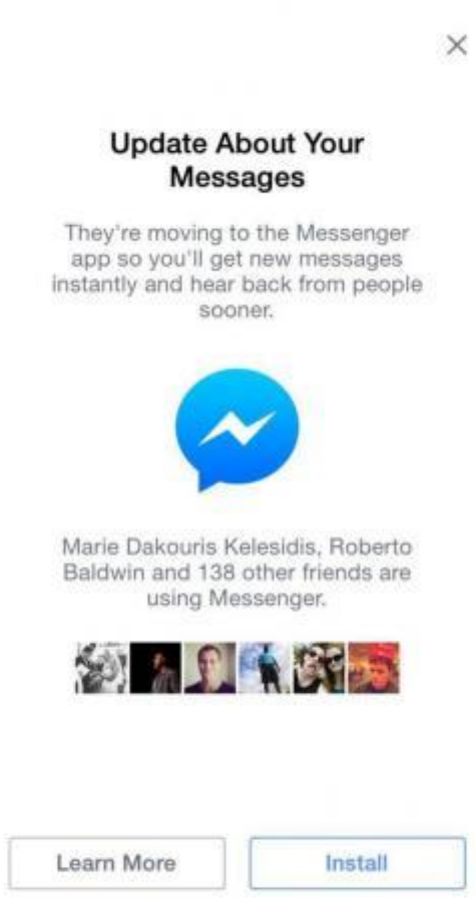
Last message received: Monday, December 17, 2012 at 1:53 PM

E3 You have logged in to Facebook on your mobile and received the following message.

❍ Most likely "NOT" an attack

❍ Most likely "IS" an attack

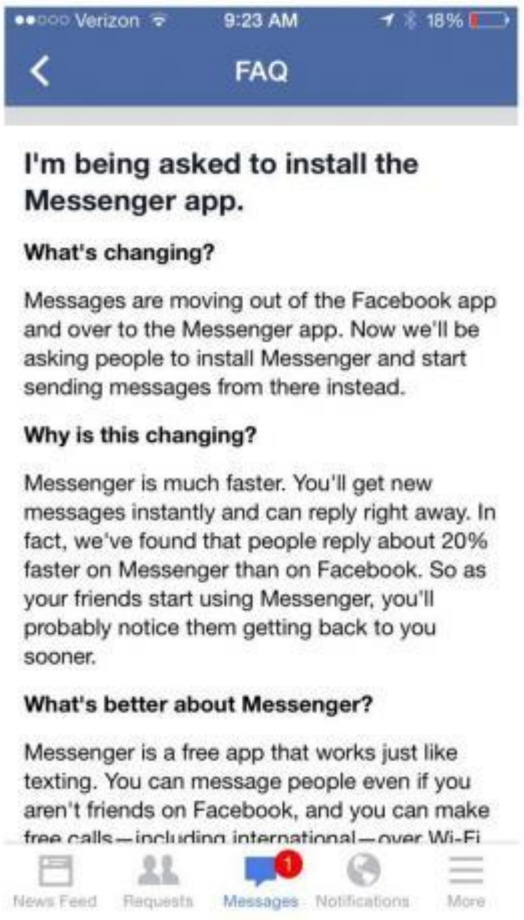
If you think this is an attack, please explain why, in one sentence



Update About Your Messages

They're moving to the Messenger app so you'll get new messages instantly and hear back from people sooner.

Marie Dakouris Kelesidis, Roberto Baldwin and 138 other friends are using Messenger.



I'm being asked to install the Messenger app.

What's changing?

Messages are moving out of the Facebook app and over to the Messenger app. Now we'll be asking people to install Messenger and start sending messages from there instead.

Why is this changing?

Messenger is much faster. You'll get new messages instantly and can reply right away. In fact, we've found that people reply about 20% faster on Messenger than on Facebook. So as your friends start using Messenger, you'll probably notice them getting back to you sooner.

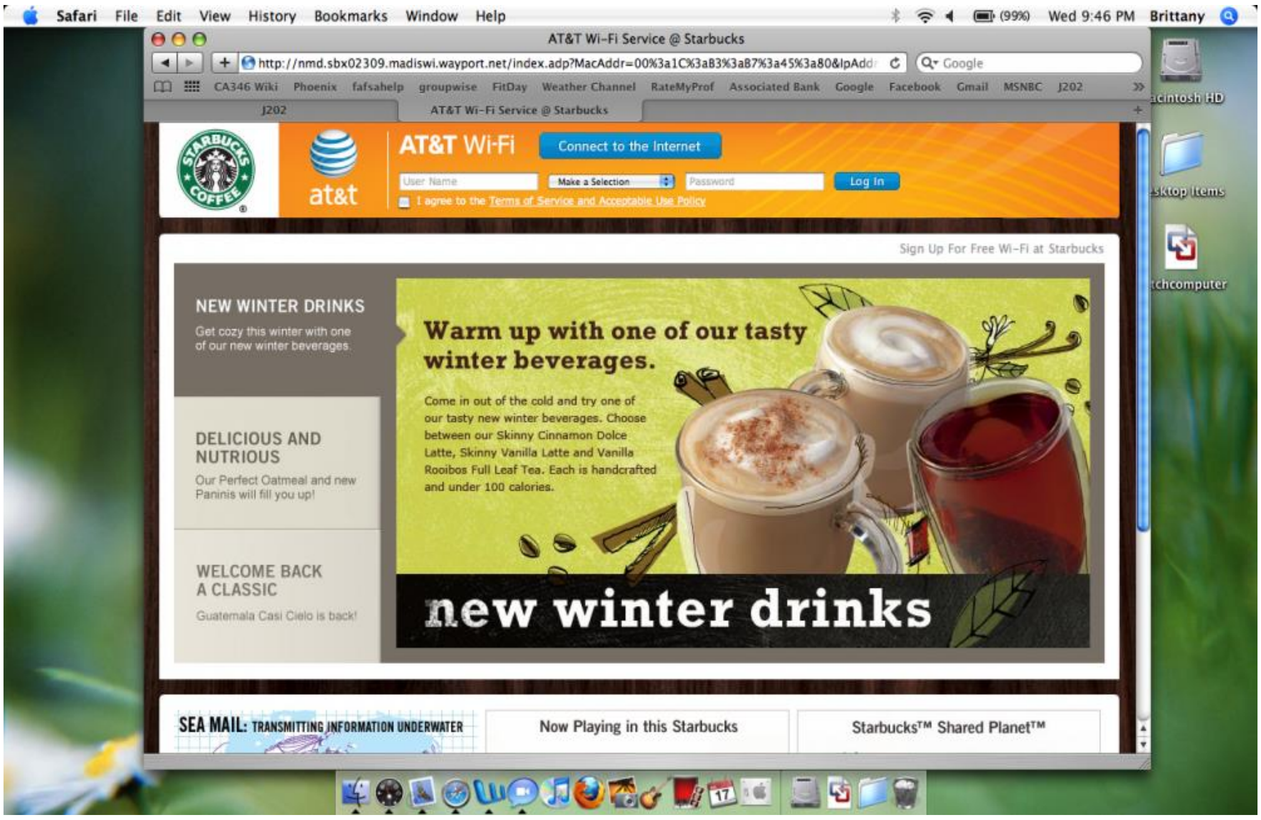
What's better about Messenger?

Messenger is a free app that works just like texting. You can message people even if you aren't friends on Facebook, and you can make free calls—including international—over Wi-Fi

E4 You are on holiday in California and connect to a WiFi access point called "Starbucks" at a Starbucks cafe, on attempting to browse the internet you are redirected to the following webpage.

❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence


Safari File Edit View History Bookmarks Window Help
(99%) Wed 9:46 PM Brittany
CA346 Wiki Phoenix fafsahelp groupwise FitDay Weather Channel RateMyProf Associated Bank Google Facebook Gmail MSNBC J202
J202
Google
Connect to the Internet
User Name
Make a Selection
Password
Log In
Sign Up For Free Wi-Fi at Starbucks
SEA MAIL: TRANSMITTING INFORMATION UNDERWATER
Now Playing in this Starbucks
Starbucks™ Shared Planet™
acintosh HD
sktop Items
tchcomputer

AT&T Wi-Fi Service @ Starbucks

http://nmd.sbx02309.madiswi.wayport.net/index.adp?MacAddr=00%3a1C%3aB3%3aB7%3a45%3a80&IpAdd

AT&T Wi-Fi

I agree to the Terms of Service and Acceptable Use Policy

NEW WINTER DRINKS
Get cozy this winter with one of our new winter beverages.

DELICIOUS AND NUTRIOUS
Our Perfect Oatmeal and new Paninis will fill you up!

WELCOME BACK A CLASSIC
Guatemala Casi Cielo is back!

Warm up with one of our tasty winter beverages.

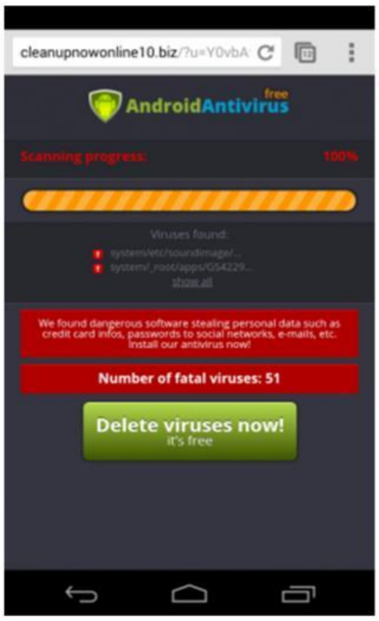
Come in out of the cold and try one of our tasty new winter beverages. Choose between our Skinny Cinnamon Dolce Latte, Skinny Vanilla Latte and Vanilla Rooibos Full Leaf Tea. Each is handcrafted and under 100 calories.

new winter drinks

E5 Whilst web browsing on your mobile the following warning appears. Clicking "Delete viruses now" downloads the app circled in red. You run the app "Android Defender". Do you think the app "Android Defender" is:

- ❍ Most likely "NOT" an attack
- ❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence



cleanupnowonline10.biz/?u=Y0vbA

AndroidAntivirus free

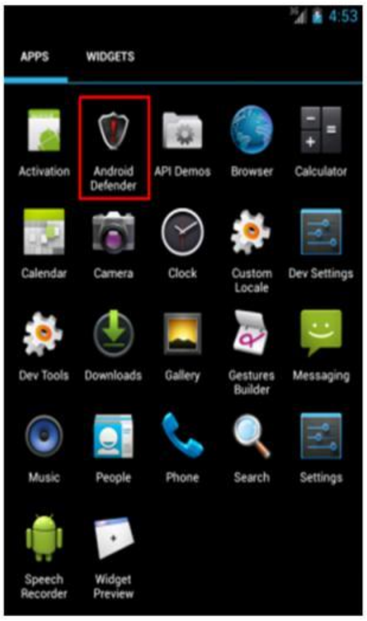
Scanning progress: 100%

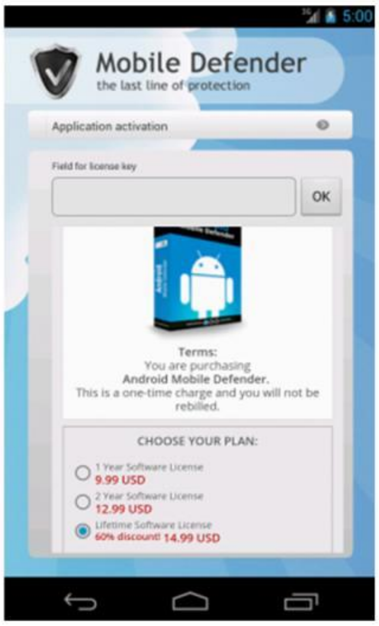
Viruses found:
system/etc/soundimage/...
system/_root/apps/G54229...
show all

We found dangerous software stealing personal data such as credit card infos, passwords to social networks, e-mails, etc. Install our antivirus now!

Number of fatal viruses: 51

Delete viruses now!
it's free





5:00

Mobile Defender
the last line of protection

Application activation

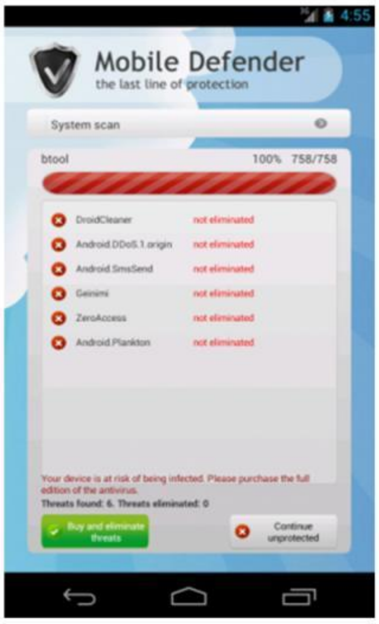
Field for license key

OK

Terms:
You are purchasing
Android Mobile Defender.
This is a one-time charge and you will not be rebilled.

CHOOSE YOUR PLAN:

- ❍ 1 Year Software License 9.99 USD
- ❍ 2 Year Software License 12.99 USD
- ◉ Lifetime Software License 60% discount! 14.99 USD



4:55

Mobile Defender
the last line of protection

System scan

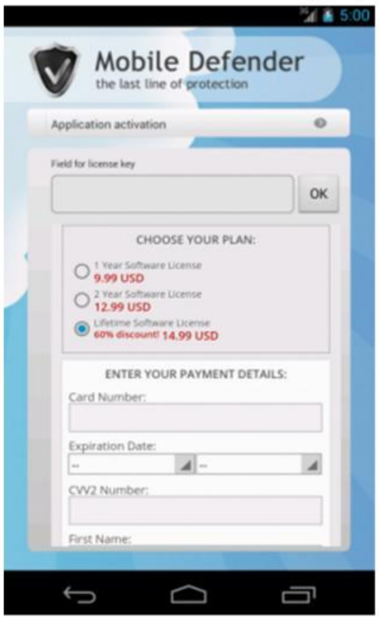
btool 100% 758/758

DroidCleaner	not eliminated
Android.DDoS.1.origin	not eliminated
Android.SmsSend	not eliminated
Geinimi	not eliminated
ZeroAccess	not eliminated
Android.Plankton	not eliminated

Your device is at risk of being infected. Please purchase the full edition of the antivirus.
Threats found: 6. Threats eliminated: 0

Buy and eliminate threats

Continue unprotected



5:00

Mobile Defender
the last line of protection

Application activation

Field for license key

OK

CHOOSE YOUR PLAN:

- ❍ 1 Year Software License 9.99 USD
- ❍ 2 Year Software License 12.99 USD
- ◉ Lifetime Software License 60% discount! 14.99 USD

ENTER YOUR PAYMENT DETAILS:

Card Number:

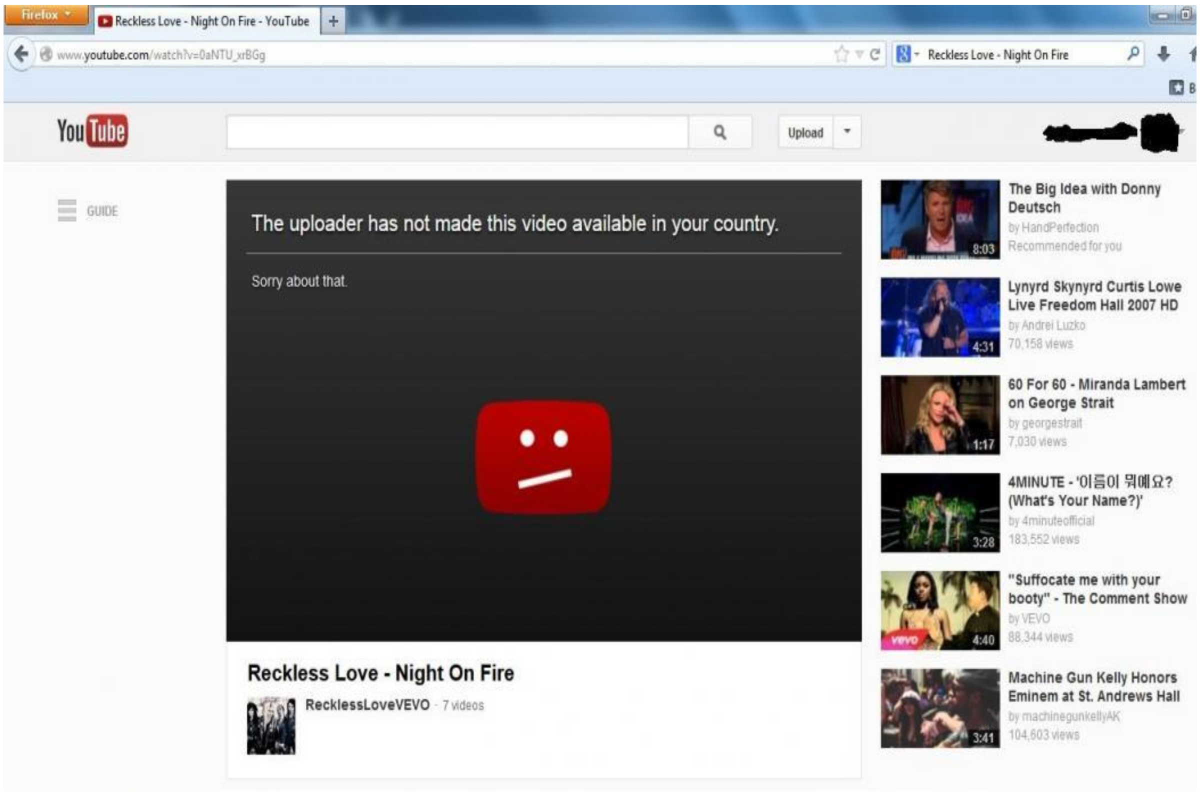
Expiration Date: -- --

CVV2 Number:

First Name:

E6 You are browsing Youtube and search for the video "Reckless love", which you find and attempt to play.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence


Reckless Love - Night On Fire - YouTube

www.youtube.com/watch?v=0aNTU_xrBGg

[redacted]

The uploader has not made this video available in your country.

Sorry about that.

Reckless Love - Night On Fire

RecklessLoveVEVO · 7 videos

Firefox
Reckless Love - Night On Fire
Upload
GUIDE
The Big Idea with Donny Deutsch
by HandPerfection
Recommended for you
8:03
Lynyrd Skynyrd Curtis Lowe Live Freedom Hall 2007 HD
by Andrei Luzko
70,158 views
4:31
60 For 60 - Miranda Lambert on George Strait
by georgestrait
7,030 views
1:17
4MINUTE - '이름이 뭐예요? (What's Your Name?)'
by 4minuteofficial
183,552 views
3:28
"Suffocate me with your booty" - The Comment Show
by VEVO
88,344 views
4:40
Machine Gun Kelly Honors Eminem at St. Andrews Hall
by machinegunkellyAK
104,603 views
3:41

E7 The following video shows Selina Carlylse web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

https://www.dropbox.com/s/jak2in8k7q3824q/E8.webm?dl=0

E8 The following video shows Selina Carlylse web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

https://www.dropbox.com/s/3d0vi7dxyy4022s/E9.webm?dl=0

E9 The following videos shows Selina Carlysle web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

Video: https://www.dropbox.com/s/3d0vi7dxyy4022s/E9.webm?dl=0

E10 The following video shows Selina Carlysle web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

Video: https://www.dropbox.com/s/7aose6bolyvtpl3/E10.webm?dl=0

E11 The following video shows Selina Carlysle web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

Video: https://www.dropbox.com/s/l05u6akwk3ra4os/E11.webm?dl=0

E12 The following video shows Selina Carlysle access web browsing.
❍ Most likely "NOT" an attack
❍ Most likely "IS" an attack

If you think this is an attack, please explain why, in one sentence

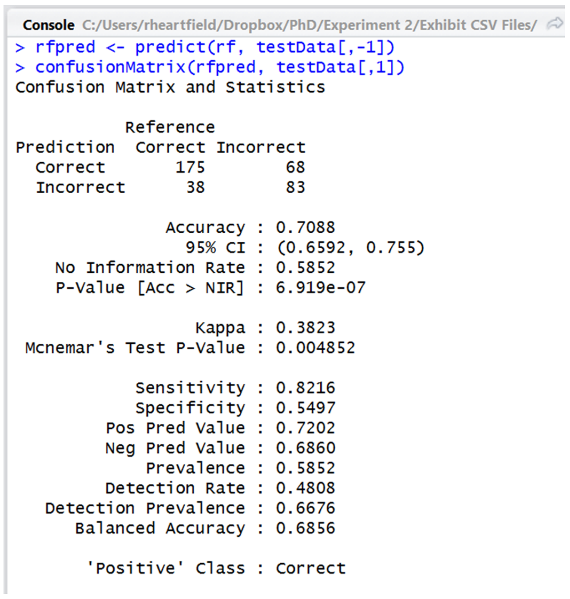
Video: https://www.dropbox.com/s/5023ureohok2w0b/E12.webm?dl=0

Appendix B

R Modelling Output and Cogni-Sense Technical Details

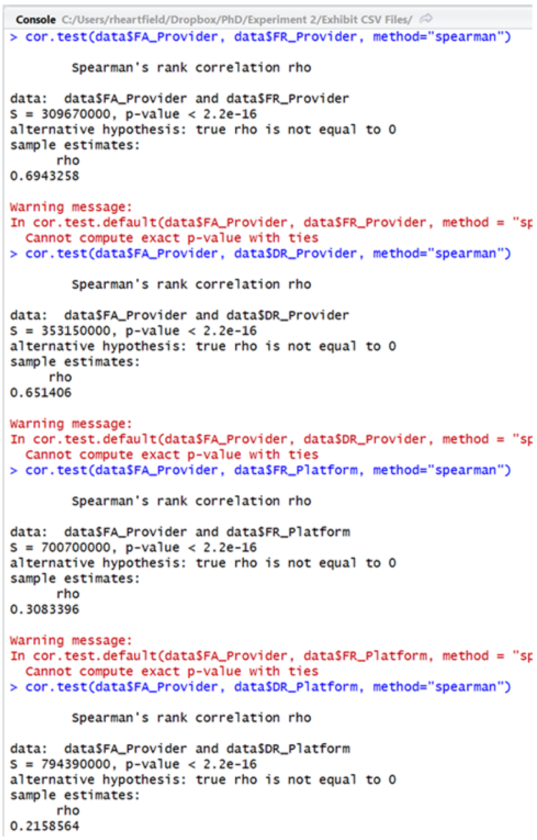
B.1 Random Forest Susceptibility Prediction Modelling R results

Figure B.1: Random Forest Test Accuracy R Model Console Output

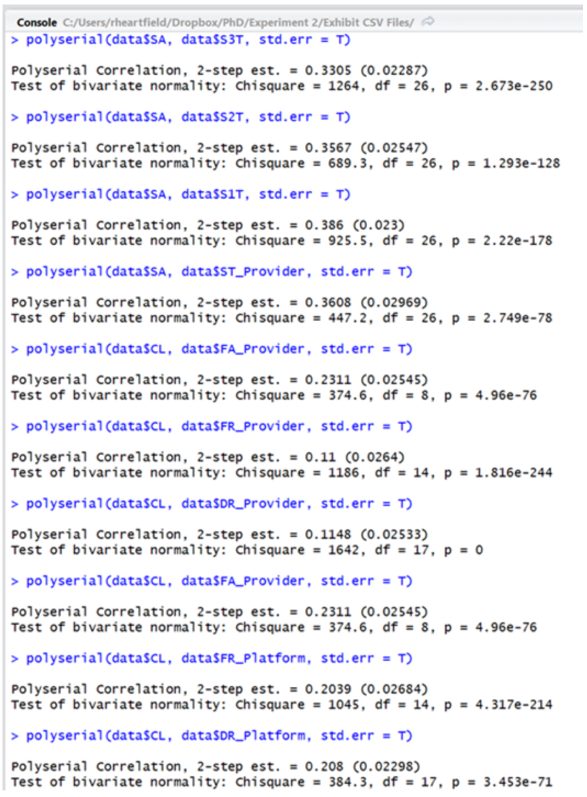


```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> rfpred <- predict(rf, testData[,-1])
> confusionMatrix(rfpred, testData[,1])
Confusion Matrix and Statistics

           Reference
Prediction  Correct Incorrect
  Correct       175        68
  Incorrect      38        83

               Accuracy : 0.7088
                 95% CI : (0.6592, 0.755)
    No Information Rate : 0.5852
    P-Value [Acc > NIR] : 6.919e-07

                  Kappa : 0.3823
 Mcnemar's Test P-Value : 0.004852

            Sensitivity : 0.8216
            Specificity : 0.5497
         Pos Pred Value : 0.7202
         Neg Pred Value : 0.6860
             Prevalence : 0.5852
         Detection Rate : 0.4808
   Detection Prevalence : 0.6676
      Balanced Accuracy : 0.6856

       'Positive' Class : Correct
```

B.2 Self-efficacy Feature Evaluation and Modelling R results

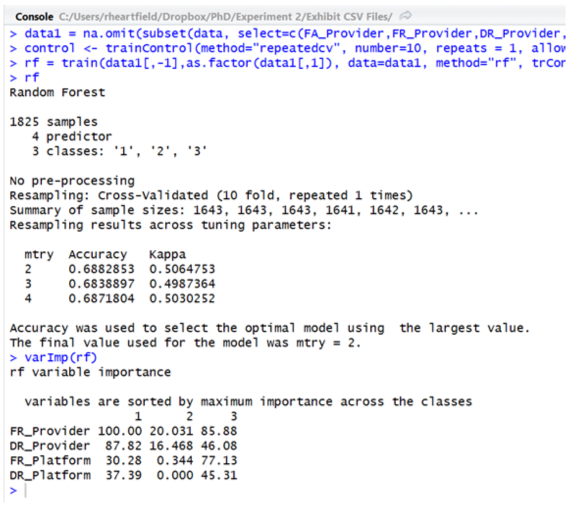


```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> cor.test(data$FA_Provider, data$FR_Provider, method="spearman")

        Spearman's rank correlation rho

data:  data$FA_Provider and data$FR_Provider
S = 309670000, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.6943258

Warning message:
In cor.test.default(data$FA_Provider, data$FR_Provider, method = "sp
  Cannot compute exact p-value with ties
> cor.test(data$FA_Provider, data$DR_Provider, method="spearman")

        Spearman's rank correlation rho

data:  data$FA_Provider and data$DR_Provider
S = 353150000, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
     rho
0.651406

Warning message:
In cor.test.default(data$FA_Provider, data$DR_Provider, method = "sp
  Cannot compute exact p-value with ties
> cor.test(data$FA_Provider, data$FR_Platform, method="spearman")

        Spearman's rank correlation rho

data:  data$FA_Provider and data$FR_Platform
S = 700700000, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.3083396

Warning message:
In cor.test.default(data$FA_Provider, data$FR_Platform, method = "sp
  Cannot compute exact p-value with ties
> cor.test(data$FA_Provider, data$DR_Platform, method="spearman")

        Spearman's rank correlation rho

data:  data$FA_Provider and data$DR_Platform
S = 794390000, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.2158564
```

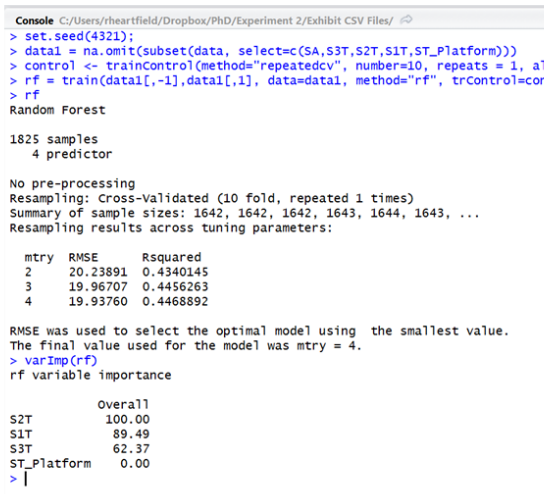


```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> polyserial(data$SA, data$S3T, std.err = T)

Polyserial Correlation, 2-step est. = 0.3305 (0.02287)
Test of bivariate normality: Chisquare = 1264, df = 26, p = 2.673e-250

> polyserial(data$SA, data$S2T, std.err = T)

Polyserial Correlation, 2-step est. = 0.3567 (0.02547)
Test of bivariate normality: Chisquare = 689.3, df = 26, p = 1.293e-128

> polyserial(data$SA, data$S1T, std.err = T)

Polyserial Correlation, 2-step est. = 0.386 (0.023)
Test of bivariate normality: Chisquare = 925.5, df = 26, p = 2.22e-178

> polyserial(data$SA, data$ST_Provider, std.err = T)

Polyserial Correlation, 2-step est. = 0.3608 (0.02969)
Test of bivariate normality: Chisquare = 447.2, df = 26, p = 2.749e-78

> polyserial(data$CL, data$FA_Provider, std.err = T)

Polyserial Correlation, 2-step est. = 0.2311 (0.02545)
Test of bivariate normality: Chisquare = 374.6, df = 8, p = 4.96e-76

> polyserial(data$CL, data$FR_Provider, std.err = T)

Polyserial Correlation, 2-step est. = 0.11 (0.0264)
Test of bivariate normality: Chisquare = 1186, df = 14, p = 1.816e-244

> polyserial(data$CL, data$DR_Provider, std.err = T)

Polyserial Correlation, 2-step est. = 0.1148 (0.02533)
Test of bivariate normality: Chisquare = 1642, df = 17, p = 0

> polyserial(data$CL, data$FA_Provider, std.err = T)

Polyserial Correlation, 2-step est. = 0.2311 (0.02545)
Test of bivariate normality: Chisquare = 374.6, df = 8, p = 4.96e-76

> polyserial(data$CL, data$FR_Platform, std.err = T)

Polyserial Correlation, 2-step est. = 0.2039 (0.02684)
Test of bivariate normality: Chisquare = 1045, df = 14, p = 4.317e-214

> polyserial(data$CL, data$DR_Platform, std.err = T)

Polyserial Correlation, 2-step est. = 0.208 (0.02298)
Test of bivariate normality: Chisquare = 384.3, df = 17, p = 3.453e-71
```

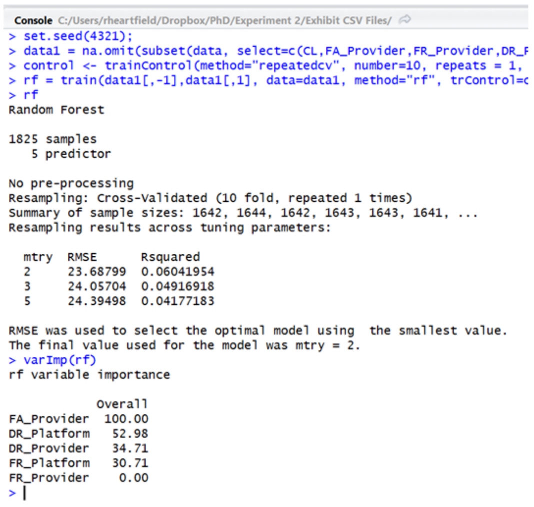
Figure B.2: Correlation results between self-efficacy features and individual auditable features



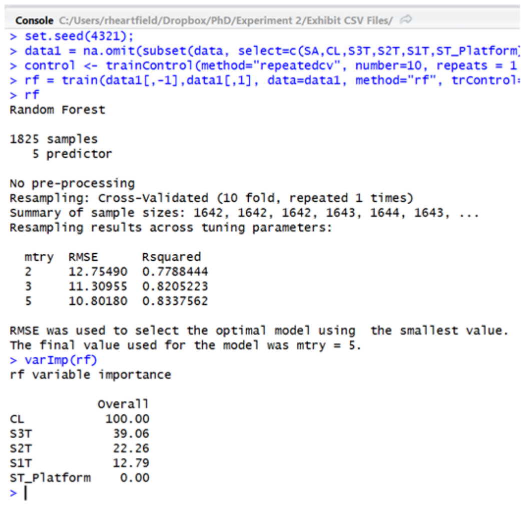
```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> data1 = na.omit(subset(data, select=c(FA_Provider,FR_Provider,DR_Provider,
> control <- trainControl(method="repeatedcv", number=10, repeats = 1, allow
> rf = train(data1[,-1],as.factor(data1[,1]), data=data1, method="rf", trCor
> rf
Random Forest

1825 samples
   4 predictor
   3 classes: '1', '2', '3'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 1643, 1643, 1643, 1641, 1642, 1643, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.6882853  0.5064753
  3     0.6838897  0.4987364
  4     0.6871804  0.5030252

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 2.
> varImp(rf)
rf variable importance

  variables are sorted by maximum importance across the classes
                 1      2     3
FR_Provider 100.00 20.031 85.88
DR_Provider  87.82 16.468 46.08
FR_Platform  30.28  0.344 77.13
DR_Platform  37.39  0.000 45.31
> |
```



```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> set.seed(4321);
> data1 = na.omit(subset(data, select=c(SA,S3T,S2T,S1T,ST_Platform)))
> control <- trainControl(method="repeatedcv", number=10, repeats = 1, al
> rf = train(data1[,-1],data1[,1], data=data1, method="rf", trControl=cor
> rf
Random Forest

1825 samples
   4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 1642, 1642, 1642, 1643, 1644, 1643, ...
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared
  2     20.23891  0.4340145
  3     19.96707  0.4456263
  4     19.93760  0.4468892

RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was mtry = 4.
> varImp(rf)
rf variable importance

            Overall
S2T          100.00
S1T           89.49
S3T           62.37
ST_Platform    0.00
> |
```



```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> set.seed(4321);
> data1 = na.omit(subset(data, select=c(CL,FA_Provider,FR_Provider,DR_F
> control <- trainControl(method="repeatedcv", number=10, repeats = 1,
> rf = train(data1[,-1],data1[,1], data=data1, method="rf", trControl=c
> rf
Random Forest

1825 samples
   5 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 1642, 1644, 1642, 1643, 1643, 1641, ...
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared
  2     23.68799  0.06041954
  3     24.05704  0.04916918
  5     24.39498  0.04177183

RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was mtry = 2.
> varImp(rf)
rf variable importance

            Overall
FA_Provider  100.00
DR_Platform   52.98
DR_Provider   34.71
FR_Platform   30.71
FR_Provider    0.00
> |
```



```
Console C:/Users/rheartfield/Dropbox/PhD/Experiment 2/Exhibit CSV Files/
> set.seed(4321);
> data1 = na.omit(subset(data, select=c(SA,CL,S3T,S2T,S1T,ST_Platform]
> control <- trainControl(method="repeatedcv", number=10, repeats = 1
> rf = train(data1[,-1],data1[,1], data=data1, method="rf", trControl:
> rf
Random Forest

1825 samples
   5 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 1642, 1642, 1642, 1643, 1644, 1643, ...
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared
  2     12.75490  0.7788444
  3     11.30955  0.8205223
  5     10.80180  0.8337562

RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was mtry = 5.
> varImp(rf)
rf variable importance

            Overall
CL           100.00
S3T           39.06
S2T           22.26
S1T           12.79
ST_Platform    0.00
> |
```

Figure B.3: Random Forest prediction accuracy for self-efficacy features using auditable features

Algorithm 1: Algorithm for generating platform/application/system frequency (FR) feature

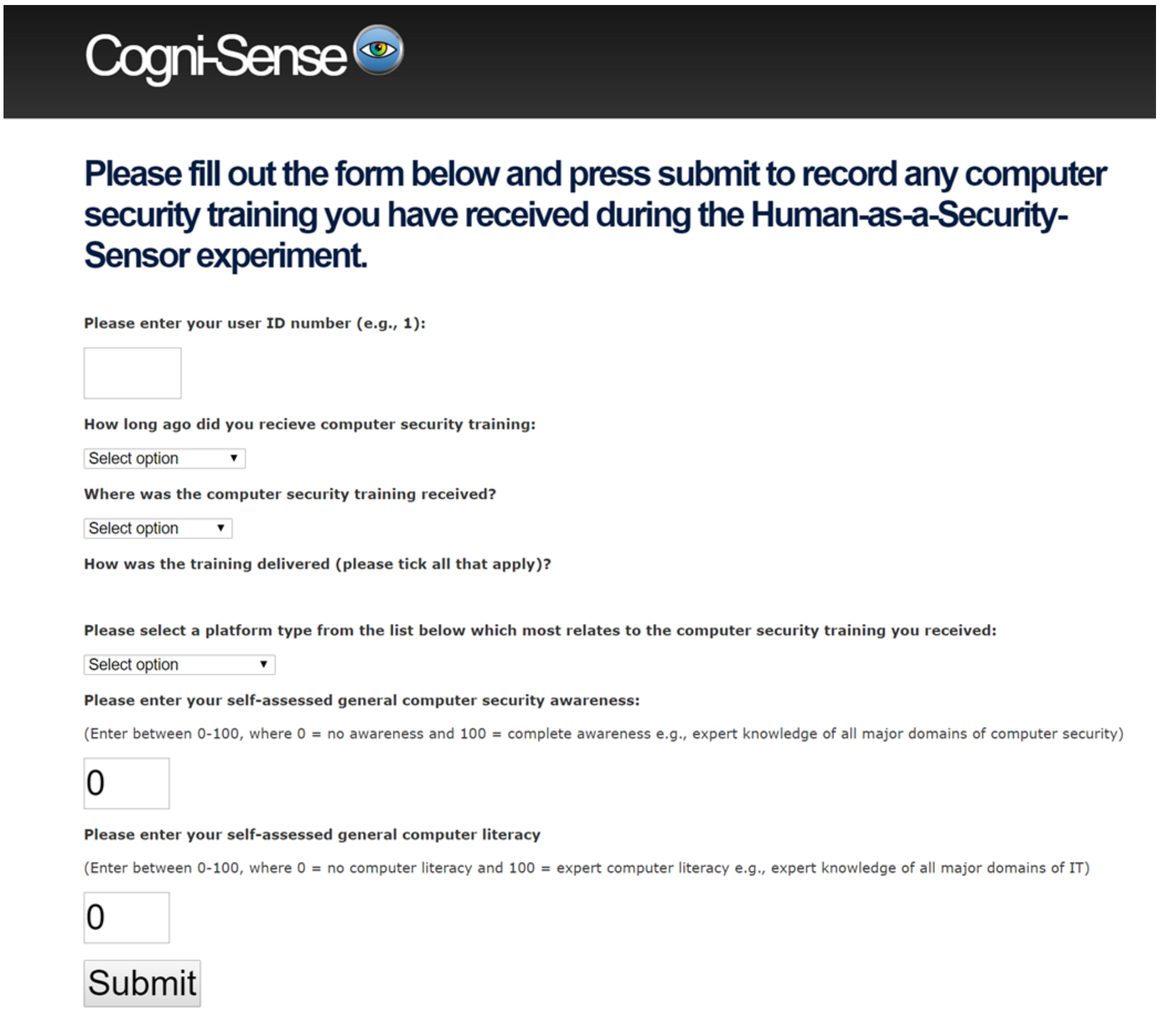
```
function Frequency $(\cdot v)$;
Input  : Array of dates $\cdot v = (v_1, v_2, ..., v_n)$
Output: $FR$
if $\cdot v = 0$ then
    return $FR = Never$;
else
    for $i \in \cdot v$ do
        $j = i + 1$;
        $E_i = |v_i - v_j|$;
        $i + +$
    end
end
for $i \in E$ do
    $FR = \begin{cases} Less\ than\ once\ a\ month, & \text{if } E_i \geq 31 \\ Monthly, & \text{if } E_i \leq 31 \textbf{ and } > 7 \\ Weekly, & \text{if } E_i \leq 7 \textbf{ and } > 1 \\ Daily, & \text{if } E_i = 1 \end{cases}$
    if $FR = Less\ than\ once\ a\ month$ then
        return $FR$;
    else
        if $i = len(E)$ then
            $FR = \begin{cases} Monthly, & \text{if } FR = Monthly \\ Weekly, & \text{if } FR = Weekly \\ Daily, & \text{if } FR = Daily \textbf{ and } len(E) \geq 31 \end{cases}$
            return $FR$;
        else
            $i + +$
        end
    end
end
```

Algorithm 2: Algorithm for generating platform/application/system duration (DR) feature

```
1 function Duration ($\cdot v$);
  Input  : Array $\cdot v = (v_{1_1}, v_{1_2}, ..., v_{n_j})$
  Output: $DR$
2 for $i \in v$ do
3     $sec_i = \sum_{j=0}^{j} v_{i_j}$;
4 end
5 $D = len(sec_i)$;
6 $T_{sec} = \frac{\sum_{i=0}^{i} sec_i}{D}$;
7 $DR = \begin{cases} 1, & \text{if } T_{sec} = Never \\ 2, & \text{if } T_{sec} \leq Less\ than\ 30mins \\ 3, & \text{if } T_{sec} \leq 30mins\ to\ 1hour \\ 4, & \text{if } T_{sec} \leq 1\ to\ 2hours \\ 5, & \text{if } T_{sec} \leq 2\ to\ 4hours \\ 6, & \text{if } T_{sec} > 4hours+ \end{cases}$
8 return $DR$;
```

B.3 User activity monitoring algorithms

B.4 User computer security training form (includes CL and SA self-efficacy


Cogni-Sense

Please fill out the form below and press submit to record any computer security training you have received during the Human-as-a-Security-Sensor experiment.

Please enter your user ID number (e.g., 1):

How long ago did you recieve computer security training:

Select option ▾

Where was the computer security training received?

Select option ▾

How was the training delivered (please tick all that apply)?

Please select a platform type from the list below which most relates to the computer security training you received:

Select option ▾

Please enter your self-assessed general computer security awareness:

(Enter between 0-100, where 0 = no awareness and 100 = complete awareness e.g., expert knowledge of all major domains of computer security)

0

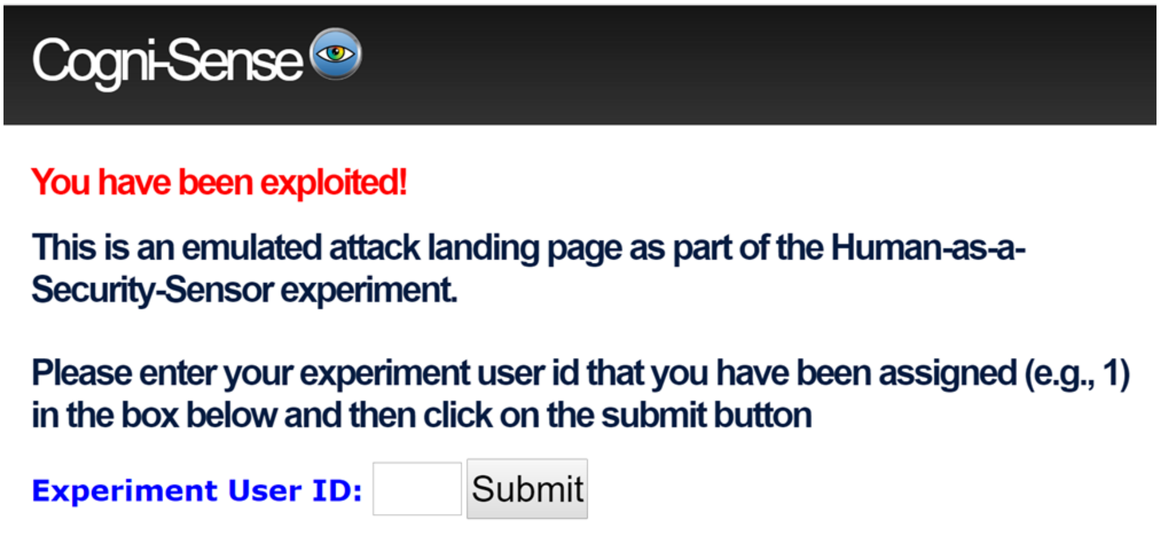
Please enter your self-assessed general computer literacy

(Enter between 0-100, where 0 = no computer literacy and 100 = expert computer literacy e.g., expert knowledge of all major domains of IT)

0

Submit

Figure B.4: Cogni-Sense user form for updating computer security record



You have been exploited!

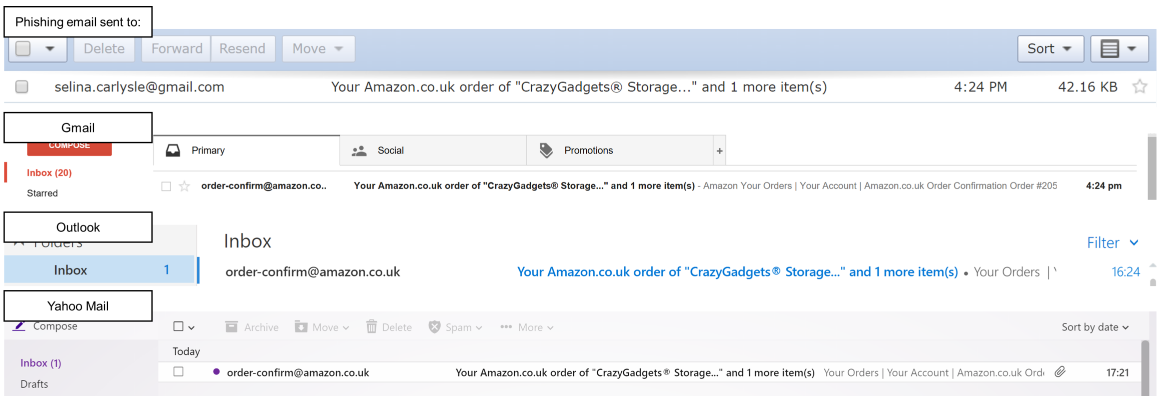
This is an emulated attack landing page as part of the Human-as-a-Security-Sensor experiment.

Please enter your experiment user id that you have been assigned (e.g., 1) in the box below and then click on the submit button

Experiment User ID:

Figure B.5: Cogni-Sense attack landing page for users exploited by experiments emulated semantic attacks

B.5 Technical Defence Evaluation against Experiment 3's semantic attacks



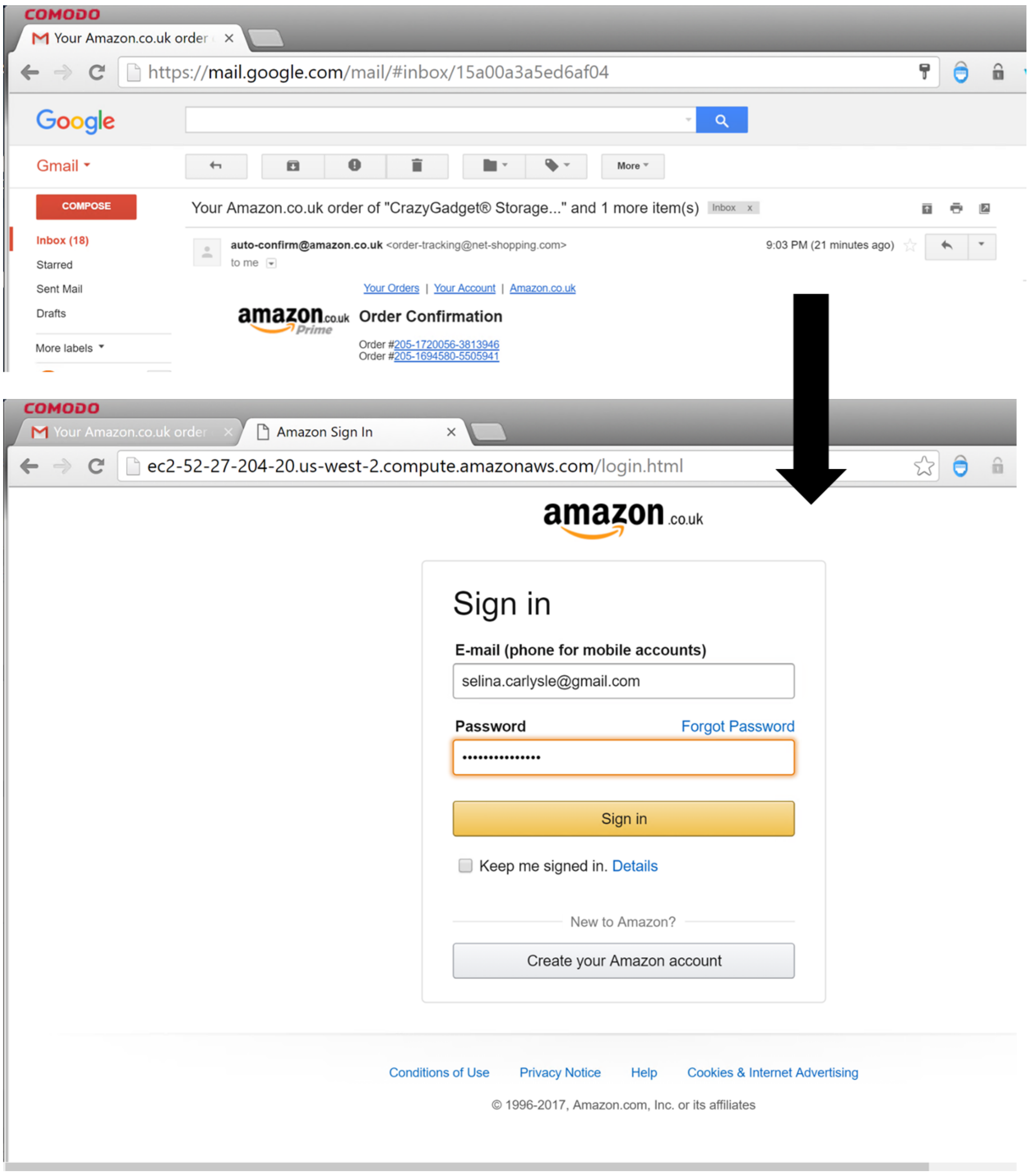
selina.carlysle@gmail.com Your Amazon.co.uk order of "CrazyGadgets® Storage..." and 1 more item(s) 4:24 PM 42.16 KB

order-confirm@amazon.co.. Your Amazon.co.uk order of "CrazyGadgets® Storage..." and 1 more item(s) - Amazon Your Orders | Your Account | Amazon.co.uk Order Confirmation Order #205 4:24 pm

order-confirm@amazon.co.uk Your Amazon.co.uk order of "CrazyGadgets® Storage..." and 1 more item(s) • Your Orders | 16:24

order-confirm@amazon.co.uk Your Amazon.co.uk order of "CrazyGadgets® Storage..." and 1 more item(s) Your Orders | Your Account | Amazon.co.uk Ord 17:21

Figure B.6: All major email providers with anti-phishing technology failure to detect amazon phishing email or generate any warnings of suspected phishing


https://mail.google.com/mail/#inbox/15a00a3a5ed6af04

Your Amazon.co.uk order of "CrazyGadget® Storage..." and 1 more item(s)

auto-confirm@amazon.co.uk <order-tracking@net-shopping.com>
to me
9:03 PM (21 minutes ago)

Your Orders | Your Account | Amazon.co.uk

Order Confirmation

Order #205-1720056-3813946
Order #205-1694580-5505941

ec2-52-27-204-20.us-west-2.compute.amazonaws.com/login.html

Sign in

E-mail (phone for mobile accounts)
selina.carlysle@gmail.com

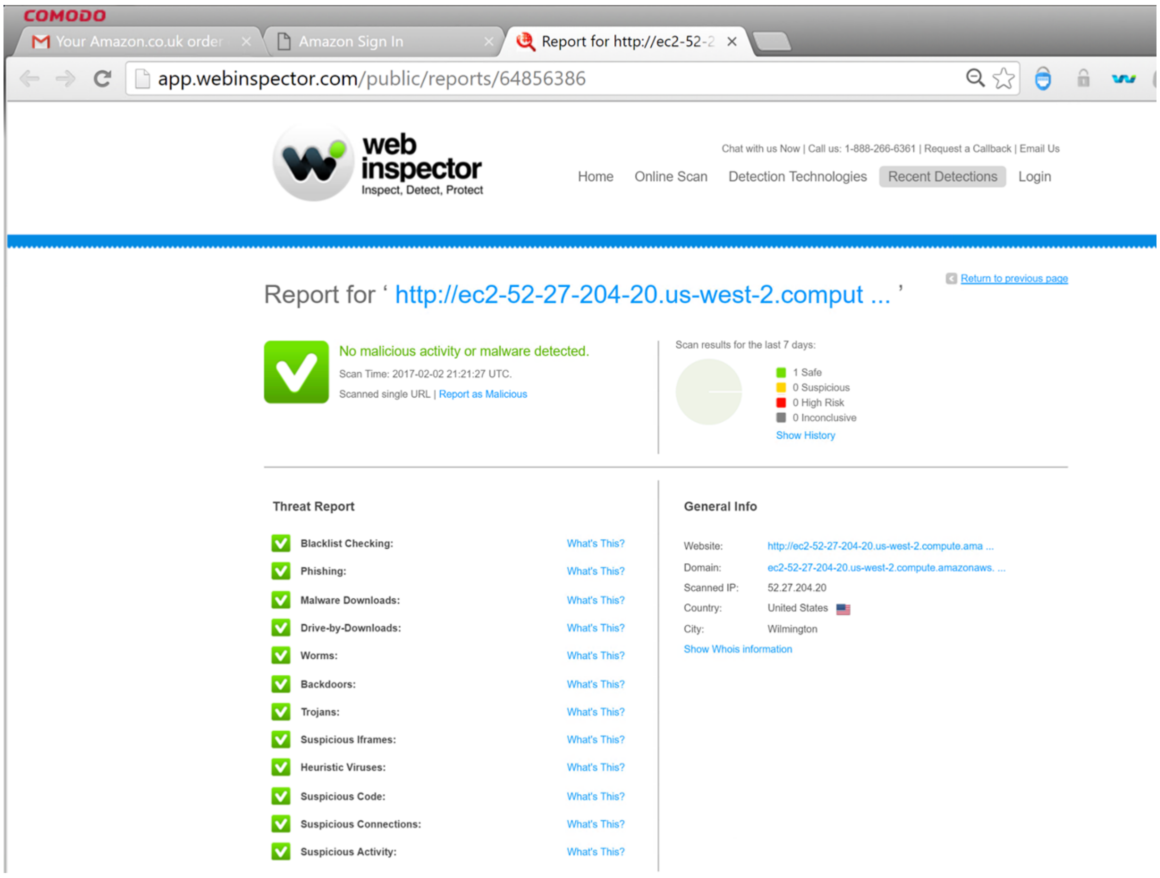
Password
•••••••••••••••

Keep me signed in. Details

New to Amazon?

© 1996-2017, Amazon.com, Inc. or its affiliates

Figure B.7: Comodo Dragon browser with ant-phishing and anti-malware and sandbox technology (based on Google Chromium) fails to generate warnings or block links in the phishing email that lead to the Amazon login phishing website; which also not detected as malicious.



Report for ' http://ec2-52-27-204-20.us-west-2.comput ... '

No malicious activity or malware detected.

Scan Time: 2017-02-02 21:21:27 UTC.

Scanned single URL | Report as Malicious

Scan results for the last 7 days:

- 🟩 1 Safe
- 🟨 0 Suspicious
- 🟥 0 High Risk
- ⬛ 0 Inconclusive

Show History

Threat Report

✓	Blacklist Checking:	What's This?
✓	Phishing:	What's This?
✓	Malware Downloads:	What's This?
✓	Drive-by-Downloads:	What's This?
✓	Worms:	What's This?
✓	Backdoors:	What's This?
✓	Trojans:	What's This?
✓	Suspicious Iframes:	What's This?
✓	Heuristic Viruses:	What's This?
✓	Suspicious Code:	What's This?
✓	Suspicious Connections:	What's This?
✓	Suspicious Activity:	What's This?

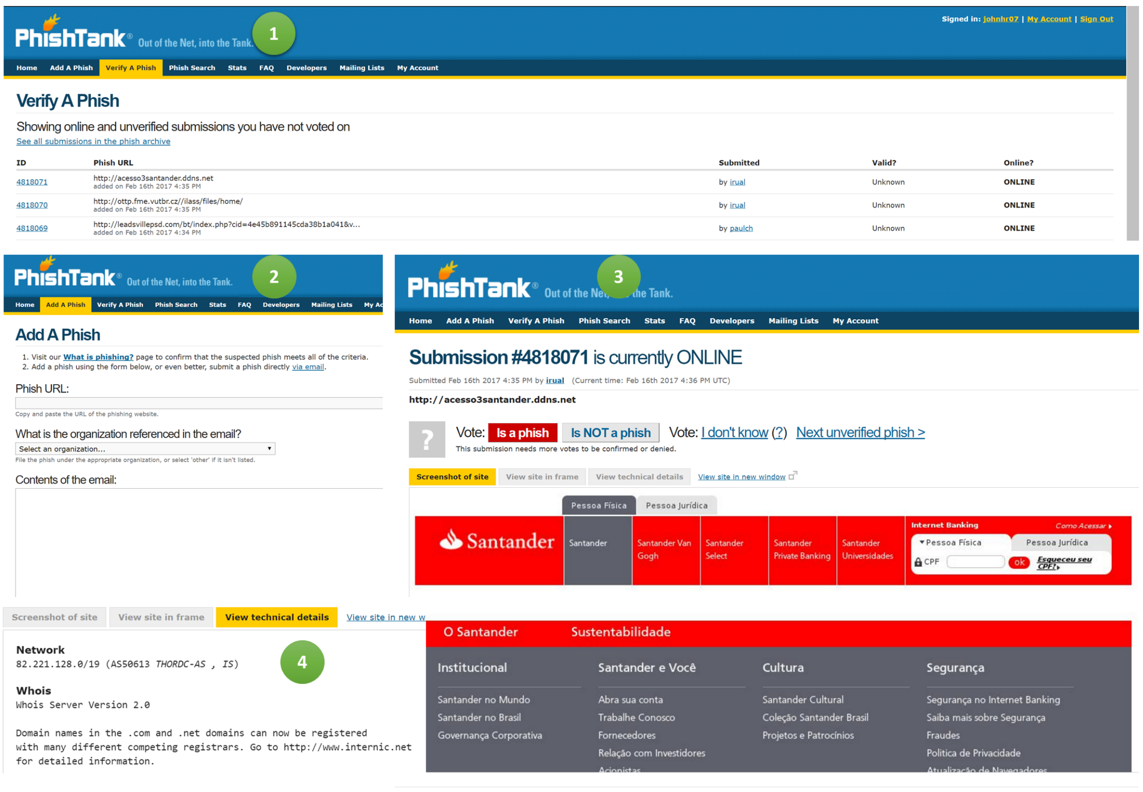
General Info

Website:	http://ec2-52-27-204-20.us-west-2.compute.ama ...
Domain:	ec2-52-27-204-20.us-west-2.compute.amazonaws. ...
Scanned IP:	52.27.204.20
Country:	United States
City:	Wilmington

Show Whois information

Figure B.8: Comodo Dragon browser webinspector application - evaluates the legitimacy and rating of website based on blacklist and heuristic factors. The website check fails to identify the spoof Amazon login webpage.

B.6 User Phishing Report Repositories



Verify A Phish

Showing online and unverified submissions you have not voted on

See all submissions in the phish archive

ID	Phish URL	Submitted	Valid?	Online?
4818071	http://acesso3santander.ddns.net added on Feb 16th 2017 4:35 PM	by irual	Unknown	ONLINE
4818070	http://ottp.fme.vutbr.cz//ilass/files/home/ added on Feb 16th 2017 4:35 PM	by irual	Unknown	ONLINE
4818069	http://leadsvillepsd.com/bt/index.php?cid=4e45b891145cda38b1a041&v... added on Feb 16th 2017 4:34 PM	by paulch	Unknown	ONLINE

Add A Phish

1. Visit our What is phishing? page to confirm that the suspected phish meets all of the criteria.
2. Add a phish using the form below, or even better, submit a phish directly via email.

Phish URL:

Copy and paste the URL of the phishing website.

What is the organization referenced in the email?

Select an organization...

File the phish under the appropriate organization, or select 'other' if it isn't listed.

Contents of the email:

Submission #4818071 is currently ONLINE

Submitted Feb 16th 2017 4:35 PM by irual (Current time: Feb 16th 2017 4:36 PM UTC)

http://acesso3santander.ddns.net

This submission needs more votes to be confirmed or denied.

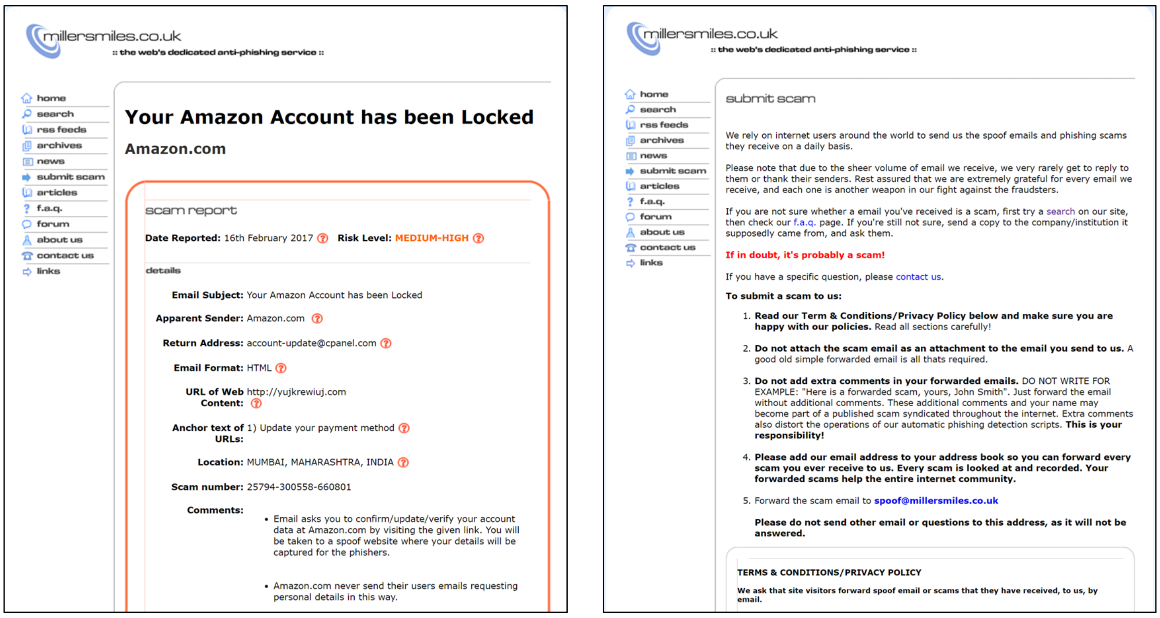
Network

```
82.221.128.0/19 (AS50613 THORDC-AS , IS)
```

Whois

```
Whois Server Version 2.0

Domain names in the .com and .net domains can now be registered
with many different competing registrars. Go to http://www.internic.net
for detailed information.
```

Figure B.9: Phishtank phishing report repository, 1) Live report of most recent reported phishing attacks, 2) Phishing report form, 3) Review and classification (votes) of phishing report in PhishTank sandbox, 4) Phishing technical details



Your Amazon Account has been Locked

Amazon.com

scam report

Date Reported: 16th February 2017 **Risk Level:** MEDIUM-HIGH

details

Email Subject: Your Amazon Account has been Locked

Apparent Sender: Amazon.com

Return Address: account-update@cpanel.com

Email Format: HTML

URL of Web Content: http://yujkrewiuj.com

Anchor text of URLs: 1) Update your payment method

Location: MUMBAI, MAHARASHTRA, INDIA

Scam number: 25794-300558-660801

Comments:

- Email asks you to confirm/update/verify your account data at Amazon.com by visiting the given link. You will be taken to a spoof website where your details will be captured for the phishers.
- Amazon.com never send their users emails requesting personal details in this way.

submit scam

We rely on internet users around the world to send us the spoof emails and phishing scams they receive on a daily basis.

Please note that due to the sheer volume of email we receive, we very rarely get to reply to them or thank their senders. Rest assured that we are extremely grateful for every email we receive, and each one is another weapon in our fight against the fraudsters.

If you are not sure whether a email you've received is a scam, first try a search on our site, then check our f.a.q. page. If you're still not sure, send a copy to the company/institution it supposedly came from, and ask them.

If in doubt, it's probably a scam!

If you have a specific question, please contact us.

To submit a scam to us:

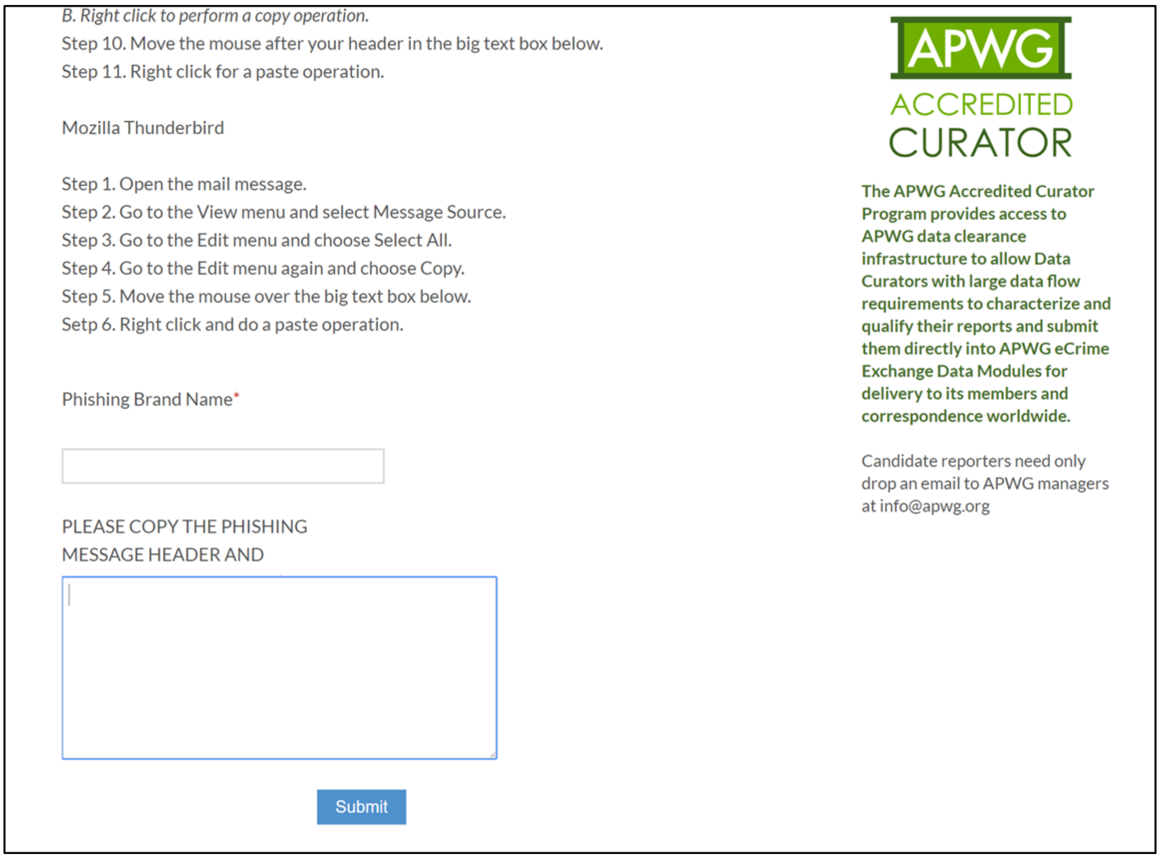
1. **Read our Term & Conditions/Privacy Policy below and make sure you are happy with our policies.** Read all sections carefully!
2. **Do not attach the scam email as an attachment to the email you send to us.** A good old simple forwarded email is all thats required.
3. **Do not add extra comments in your forwarded emails.** DO NOT WRITE FOR EXAMPLE: "Here is a forwarded scam, yours, John Smith". Just forward the email without additional comments. These additional comments and your name may become part of a published scam syndicated throughout the internet. Extra comments also distort the operations of our automatic phishing detection scripts. **This is your responsibility!**
4. **Please add our email address to your address book so you can forward every scam you ever receive to us. Every scam is looked at and recorded. Your forwarded scams help the entire internet community.**
5. Forward the scam email to **spoof@millersmiles.co.uk**

   Please do not send other email or questions to this address, as it will not be answered.

TERMS & CONDITIONS/PRIVACY POLICY

We ask that site visitors forward spoof email or scams that they have received, to us, by email.

Figure B.10: Millersmiles spam and phishing report repository, phishing report view (left) and phishing report instructions (right)



B. Right click to perform a copy operation.

Step 10. Move the mouse after your header in the big text box below.

Step 11. Right click for a paste operation.

Mozilla Thunderbird

Step 1. Open the mail message.

Step 2. Go to the View menu and select Message Source.

Step 3. Go to the Edit menu and choose Select All.

Step 4. Go to the Edit menu again and choose Copy.

Step 5. Move the mouse over the big text box below.

Setp 6. Right click and do a paste operation.

Phishing Brand Name*

PLEASE COPY THE PHISHING MESSAGE HEADER AND

The APWG Accredited Curator Program provides access to APWG data clearance infrastructure to allow Data Curators with large data flow requirements to characterize and qualify their reports and submit them directly into APWG eCrime Exchange Data Modules for delivery to its members and correspondence worldwide.

Candidate reporters need only drop an email to APWG managers at info@apwg.org

Figure B.11: Anti-phishing working group (APWG) phishing report form

References

[1] Anti-Phishing Working Group. Apwg phishing trends reports, 2015. URL `http://www.antiphishing.org/resources/apwg-reports/`.

[2] Symantec. Istr2: Internet security threat report, 2016. URL `https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf`.

[3] B. Schneier. *Secrets and lies: digital security in a networked world.* Wiley, 2011.

[4] G. Cluley. A 419 scam via snail mail. Naked Security, 2011. URL `http://nakedsecurity.sophos.com/2011/05/30/a-419-scam-via-snail-mail`.

[5] SocialEngineer. The power of the uniform in social engineering. Naked Security, 2013. URL `https://www.social-engineer.com/the-power-of-the-uniform-in-social-engineering/`.

[6] B. Schneier. Inside risks: semantic network attacks. *Communications of the ACM*, 12(168), 2000.

[7] M. Jordan and G. Heather. The signs, signifiers and semiotics of the successful semantic attack. In *14th Annual EICAR Conference*, pages 344–364, 2005.

[8] P. Thompson and A. Kott. *Deception as a semantic attack*, chapter 2.2, pages 125–144. Chapman and Hall/CRC, 2007.

[9] R. Heartfield and G. Loukas. A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks. *ACM Computing Surveys*, 48(3), 2016.

[10] M. Huber, M. Mulazzani, E. Weippl, G. Kitzler, and S. Goluch. Friend-in-the-middle attacks: Exploiting social networking sites for spam. *Internet Computing*, 3:28–34, 2011.

[11] R. Heartfield and G. Loukas. On the feasibility of automated semantic attacks in the cloud. In *Computer and Information Sciences III*, pages 343–351. Springer London, 2013.

[12] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger. Nfc devices: Security and privacy. In *Availability, Reliability and Security*, ARES 08, pages 642–647. IEEE, 2008.

[13] R. H. Weber. Internet of things: New security and privacy challenges. *Computer Law and Security Review*, 1:23–30, 2010.

[14] McAfee. Social engineering in the internet of things (iot), 2015. URL `https://blogs.mcafee.com/executive-perspectives/social-engineering-internet-things-iot`.

[15] G. Loukas. *Cyber-Physical Attacks: A Growing Invisible Threat.* Butterworth-Heinemann (Elsevier), 2015.

[16] S. Abraham and C.S. UnduShobha. An overview of social engineering malware: Trends, tactics, and implications. *Technology in Society*, 3(183-196), 2010.

[17] R. Dhamija, D. J. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems.* ACM, 2006.

[18] C.E. Drake, J. O. Jonathan, and J.K. Eugene. Anatomy of a phishing email. In *CEAS*, 2004.

[19] M. Huber, M. Mulazzani, and E. Weipp. Who on earth is mr. cypher: Automated friend injection attacks on social networking sites. In *Security and Privacy Silver Linings in the Cloud*, pages 80–89. Springer Berlin Heidelberg, 2010.

[20] C. Marforio, F. Aurelien, and S. Capkun. Application collusion attack on the permission-based security model and its implications for modern smartphone systems. report 724. Technical report, 2011.

[21] K. Selvaraj and N. F. Gutierrez. The rise of pdf malware. Symantec Security Response, 2010.

[22] M. Aburrous, M. A. Hossain, F. Thabatah, and K. Dahal. Intelligent phishing website detection system using fuzzy techniques. In *Information and Communication Technologies: From Theory to Applications*, ICTTA 2008. 3rd International Conference on. IEEE, 2008.

[23] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell. Client-side defense against web-based identity theft. In *NDSS*, 2004.

[24] H. Huang, S. Zhong, and J. Tan. Browser-side countermeasures for deceptive phishing attack. In *Information Assurance and Security*, volume 1 of *IAS'09. Fifth International Conference on.* IEEE, 2009.

[25] A. Acquisti L. F. Cranor J. Hong P. Kumaraguru, Y. Rhee and E. Nunge. Protecting people from phishing: The design and evaluation of an embedded training email system. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* ACM, 2007.

[26] P. Kumaraguru. *PhishGuru: a system for educating users about Semantic Attacks.* PhD thesis, Carnegie Mellon University, 2009.

[27] FirstCyberSecurity. Protecting your brand online and creating customer confidence, 2009. URL `http://www.firstcybersecurity.com/main/IPRiskMReview.pdf`.

[28] Webroot. Webroot real-time anti-phishing service, 2013. URL `http://www.webroot.com/shared/pdf/WAP-Anti-Phishing-102013.pdf`.

[29] J. Bates. Trojan horse: Aids information introductory diskette version 2.0. In *Virus Bulletin*, pages 3–6, 1990.

[30] A. Young and M. Yung. Cryptovirology: Extortion-based security threats and countermeasures. In *Security and Privacy, 1996. Proceedings*, pages 129–140. IEEE, 1996.

[31] Amiga Fish-Disk Database. Fish-disk 448 content: Nightmare, 1990. URL `http://amiga-fish.erkan.se/amiga-fish-disk-448-content-NightMare/`.

[32] J. Giles. Scareware the inside story. *New Scientist*, 2753:38–41, 2010.

[33] K. Rekouche. Early phishing. *arXiv preprint arXiv:1106.4692*, 2011.

[34] Anti-phishing Working Group. Phishing activity trends report. Technical report, 2005.

[35] Cisco. Viruses, worms, trojans, and bots, 2017. URL `https://www.cisco.com/c/en/us/about/security-center/virus-differences.html`.

[36] L. J Tidy, K. Shahzad, A. Muhammad, and S. Woodhead. An assessment of the contemporary threat posed by network worm malware. In *The Ninth International Conference on Systems and Networks Communications (ICSNC 2014)*, October 2014.

[37] M. Bishop. Analysis of the iloveyou worm, 2000. URL `Internet:http://nob.cs.ucdavis.edu/classes/ecs155-2005-04/handouts/iloveyou.pdf`.

[38] M. E. Kabay. Viruses and worms: more than a technical problem. *Ubiquity*, 2, 2001.

[39] Financial Cryptography. Gp4.3 - growth and fraud - case 3 - phishing, 2005. URL `http://financialcryptography.com/mt/archives/000609.html`.

[40] N. Leavitt. Mobile phones: the next frontier for hackers? *Computer*, 4:20–23, 2005.

[41] M. Dornseif. 0wned by an ipod, 2004. Presentation.

[42] J. Kong, W. Cai, and L. Wang. The evaluation of index poisoning in bittorrent. In *Communication Software and Networks 2010. ICCSN'10. Second International Conference on*, pages 382–386. IEEE, 2010.

[43] F. Howard and O. Komili. Poisoned search results: How hackers have automated search engine poisoning attacks to distribute malware. Technical report, 2010.

[44] S. Doherty, J. Gegeny, B. Spasojevic, and J. Baltazar. Hidden lynxprofessional hackers for hire. Symantec Security Response, 2013.

[45] D. Irani, M. Balduzzi, D. Balzarotti, E. Kirda, and C. Pu. Reverse social engineering attacks in online social networks. *Detection of intrusions and malware, and vulnerability assessment*, pages 55–74, 2011.

[46] G. Cluley. Osama bin laden death video scam spreads virally on facebook, 2011. URL `https://nakedsecurity.sophos.com/2011/05/02/osama-bin-laden-death-video-scam-spreads-virally-on-facebook/`.

[47] K. Zetter. A cyberattack has caused confirmed physical damage for the second time ever, 2015. URL `https://www.wired.com/2015/01/german-steel-mill-hack-destruction/`.

[48] K. Zetter. Inside the cunning, unprecedented hack of ukraines power grid, 2016. URL `https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/`.

[49] I. S. Winkler. The non-technical threat to computing systems. *Computer Systems*, 9(1):3–14, 1996.

[50] D. Harley. Re-floating the titanic: Dealing with social engineering attacks. *EICAR*, 13, 1998.

[51] T. Reeve. Even security experts fail to spot phishing emails, finds report, 2015. URL `http://www.scmagazineuk.com/even-security-experts-fail-to-spot-phishing-emails-finds-report/article/415453/`.

[52] TrendLabs. Spear-phishing email: Most favored apt attack bait. Technical report, TrendLabs - APT Research Team, 2012. URL `http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-spear-phishing-email-most-favored-apt-attack-bait.pdf`.

[53] Social Engineer. The social engineering infographic, 2014. URL `http://www.social-engineer.org/social-engineering/social-engineering-infographic/`.

[54] Statista. Types of cyber attacks experienced by companies worldwide as of august 2015, 2015. URL `http://www.statista.com/statistics/474937/cyber-crime-attacks-experienced-by-global-companies/`.

[55] Statista. Average number of days to resolve a cyber attack on companies in the united states as of august 2015, 2015. URL `http://www.statista.com/statistics/193463/average-days-to-resolve-a-cyber-attack-in-us-companies-by-attack/`.

[56] Statista. Share of cyber crime damages caused to u.s. companies through phishing and social engineering in 2015, 2015. URL `http://www.statista.com/statistics/193465/financial-damage-caused-by-phishing-for-us-companies/`.

[57] B. Atkins and W. Huang. A study of social engineering in online frauds. *Open Journal of Social Sciences*, 3:23–32, 2013.

[58] L. Corrons. The business of rogueware. In *Web Application Security*, volume 10 of *72*, pages 7–7, 2010.

[59] M. Christodorescu and S. Jha. Testing malware detectors. *ACM SIGSOFT Software Engineering Notes*, 4:34–44, 2004.

[60] L. Bilge and T. Dumitras. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security*, volume 10, pages 833–944. ACM, 2012.

[61] T. Blasing, L. Batyuk, A. D. Schmidt, S. A. Camtepe, and S. Albayrak. An android application sandbox system for suspicious software detection. In *Malicious and Unwanted Software (MALWARE)*, 5th International Conference on, pages 55–62. IEEE, 2010.

[62] C. Greamo and A.Ghosh. Sandboxing and virtualisation - modern tools for combating malware. In *Security and Privacy.*, volume 2 of *9*, pages 79–82. IEEE, 2011.

[63] M. Tavallaee, N. Stakhanova, and A. A. Ghorbani. Toward credible evaluation of anomaly-based intrusion-detection methods. In *Systems, Man, and Cybernetics, Part C: Applications and Reviews*, volume 5 of *IEEE Transactions on 40*, pages 516–524. IEEE, 2010.

[64] C. Foozy, R. Ahmad, M. Abdollah, R. Yusof, and M. Zaki. Generic taxonomy of social engineering attack. In *Malaysian Technical Universities International Conference on Engineering and Technology*, pages 527–533, 2011.

[65] K. Ivaturi and L. Janczewski. A taxonomy for social engineering attacks. In *CONF-IRM Proceedings.*, 2011.

[66] P. Tetri and J. Vuorinen. Dissecting social engineering. In *Behaviour and Information Technology*, volume 10 of *32*, pages 1014–1023, 2013.

[67] A. Algarni, Y. Xu, T. Chan, and Y. C. Tian. Social engineering in social networking sites: Affect-based model. In *Internet Technology and Secured Transactions (ICITST) 2013 8th International Conference for*, pages 508–515. IEEE, 2013.

[68] A. Sood and R. Enbody. *Targeted Cyber Attacks: Multi-staged Attacks Driven by Exploits and Malware*. Syngress, 2014.

[69] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium*, 2010.

[70] CESG. Common cyber attacks: Reducing the impact, 2015. URL https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/400106/Common_Cyber_Attacks-Reducing_The_Impact.pdf.

[71] H. Orman. The compleat story of phish. *IEEE Internet Computing*, 1:87–91, 2009.

[72] RSA. Lions at the watering hole the voho affair, 2012. URL https://blogs.rsa.com/lions-at-the-watering-hole-the-voho-affair/.

[73] B. Chaffin. someone targets hong kong protesters using jailbroken iphones with malware, 2014. URL http://www.macobserver.com/tmo/article/someone-targets-hong-kong-protesters-using-jailbroken-iphones-with-malware.

[74] M. T. Banday, J. A. Qadri, and N. A. Shah. Study of botnets and their threats to internet security, 2009. URL http://sprouts.aisnet.org/594/1/Botnet_Sprotus.pdf.

[75] V. Sharma. An analytical survey of recent worm attacks. In *IJCSNS*, volume 11, pages 99–103, 2011.

[76] H. Hwang, G. Jung, K. Sohn, and S. Park. A study on mitm (man in the middle) vulnerability in wireless network using 802.1 x and eap. In *Information Science and Security (ICISS)*, pages 164–170. IEEE, 2008.

[77] J. M. Briones, M. A. Coronel, and P. Chavez-Burbano. Case of study: Identity theft in a university wlan evil twin and cloned authentication web interface. In *Computer and Information Technology (WCCIT), 2013 World Congress on*, pages 1–4. IEEE, 2013.

[78] D. Emm. The changing face of malware. In *Proceedings of the IWWST*, 2005.

[79] G. S. Bindra. Masquerading as a trustworthy entity through portable document file (pdf) format. In *Privacy, security, risk and trust (passat)*, pages 784–789. IEEE, 2011.

[80] N. Leavitt. Instant messaging: A new target for hackers. *Computer*, 7:20–23, 2005.

[81] T. Lauinger, V. Pankakoski, D. alzarotti, and E. Kirda. Honeybot, your man in the middle for automated social engineering. In *LEET10, 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2010.

[82] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang. Knowing your enemy: understanding and detecting malicious web advertising. In *Proceedings of the 2012 ACM conference on Computer and communications security.* ACM, 2012.

[83] Y. Boshmaf, I. Muslukhov, K. Beznosov, and M. Ripeanu. The socialbot network: when bots socialize for fame and money. In *Proceedings of the 27th Annual Computer Security Applications Conference*, pages 93–102. ACM, 2011.

[84] Z. Coburn and G. Marra. Realboy believable twitter bots, 2008. URL `http://ca.olin.edu/2008/realboy/`.

[85] A. Podhradsky, R. DOvidio, P. Engebretson, and C. Casey. Xbox 360 hoaxes, social engineering, and gamertag exploits. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 3239–3250. IEEE, 2013.

[86] A. Boileau. Hit by a bus: Physical access attacks with firewire, 2006. URL `http://www.security-assessment.com/files/presentations/ab_firewire_rux2k6-final.pdf`.

[87] L. Duflot, Y. A. Perez, and B. Morin. What if you cant trust your network card? In *Recent Advances in Intrusion Detection*, pages 378–397. Springer Berlin Heidelberg, 2011.

[88] Keelog. Keelog key grabber internal module ps2 2gb, 2015. URL `https://www.keelog.com/`.

[89] B. Anderson and B. Anderson. *Seven deadliest USB attacks*. Syngress, 2010.

[90] Dennis Fisher. Massive, decades-long cyber espionage framework uncovered, 2015. URL `http://threatpost.com/massive-decades-long-cyberespionage-framework-uncovered/111080d`.

[91] T. M. Chen. Trends in viruses and worms. *The Internet Protocol Journal*, 3: 23–33, 2003.

[92] P. Ducklin. Return of the android sms virus - self-spreading selfmite worm comes back for more, 2014. URL `https://nakedsecurity.sophos.com/2014/10/10/return-of-the-android-sms-virus-self-spreading-selfmite-worm-comes-back-for-more/`.

[93] N. P. P. Mavromatis and M. A. R. F. Monrose. All your iframes point to us. In *USENIX Security Symposium*, pages 1–16. USENIX, 2008.

[94] N. Provos, M. A. Rajab, and P. Mavrommatis. Cybercrime 2.0: when the cloud turns dark. *Communications of the ACM*, 4:42–47, 2009.

[95] M. E. Johnson, D. McGuire, and N. D. Willey. The evolution of the peer-to-peer file sharing industry and the security risks for users. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences.*, pages 383–383. IEEE, 2008.

[96] S. Shin, J. Jung, and H. Balakrishnan. Malware prevalence in the kazaa file-sharing network. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM, 2006.

[97] K. P. Yee. *Guidelines and strategies for secure interaction design*, chapter 13, pages 247–273. 2005. URL `http://sid.toolness.org/ch13yee.pdf`.

[98] P. Agten, W. Joosen, F. Piessens, and N. Nikiforakis. Seven months' worth of mistakes: A longitudinal study of typosquatting abuse. In *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS 2015)*. NDSS, 2015.

[99] TrendMicro. Malaysia airlines flight 370 news used to spread online threats, 2014. URL `http://blog.trendmicro.com/trendlabs-security-intelligence/malaysia-airlines-flight-370-news-used-to-spread-online-threats/`.

[100] Y. Song, C. Yang, and G. Gu. Who is peeping at your passwords at starbucks?to catch an evil twin access point. In *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, pages 323–332. IEEE, 2010.

[101] A. P. Felt and D. Wagner. *Phishing on mobile devices*. W2SP, 2011.

[102] J.Corbetta, L. Invernizzi, C. Kruegel, and G. Vigna. Eyes of a human, eyes of a program: Leveraging different views of the web for analysis and detection. In *17th International Symposium, RAID 2014*, pages 130–149. Springer, 2014.

[103] H. Xu N. Wang and J. Grossklags. Third-party apps on facebook: privacy and the illusion of control. In *Proceedings of the 5th ACM Symposium on Computer Human Interaction for Management of Information Technology*. ACM, 2011.

[104] Pierluigi Paganini. Phishing goes mobile with cloned banking app into google play, 2014. URL `http://securityaffairs.co/wordpress/26134/cyber-crime/phishing-goes-mobile-cloned-banking-app-google-play.html`.

[105] K. Nohl and J. Lehl. Badusbon accessories that turn evil. In *Black Hat USA*, 2014.

[106] D. Sullivan. What is search engine spam? the video edition, 2008. URL `http://searchengineland.com/what-is-search-engine-spam-the-video-edition-15202`.

[107] M. Egele, P. Wurzinger, C. Kruegel, and E. Kirda. Defending browsers against drive-by downloads: Mitigating heap-spraying code injection attacks. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 88–106. Springer Berlin Heidelberg, 2010.

[108] M. Cova, C. Kruegel, and G. Vigna. Detection and analysis of drive-by-download attacks and malicious javascript code. In *Proceedings of the 19th international conference on World wide web*, pages 281–290. ACM, 2010.

[109] B. Krishna. Malicious emails masquerade as office printer messages. Symantec Connect Blog - Symantec Intelligence.ONLINE, 2011. URL `http://www.symantec.com/connect/blogs/malicious-emails-masquerade-office-printer-messages-0`.

[110] C. Dhinakaran, J. K. Lee, and D. Nagamalai. 'reminder: please update your details': Phishing trends. In *Networks and Communications*, NETCOM'09. First International Conference on, pages 295–300. IEEE, 2009.

[111] J. Szurdi, B. Kocso, G. Cseh, J. Spring, M. Felegyhazi, and C.e Kanich. The long taile of typosquatting domain names. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 191–206. USENIX, 2014.

[112] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer. Social phishing. *Communications of the ACM*, 10:49–51, 2007.

[113] S. Abu-Nimeh and S. Nair. Phishing attacks in a mobile environment. In *SMU HACNet Lab Southern Methodist University Dallas*, 2006.

[114] B. Laurie and A. Laurie. Serious flaws in bluetooth security lead to disclosure of personal data. Technical report, 2004.

[115] Symantec. Trojan.ransomcrypt.i, 2014. URL `http://www.symantec.com/security_response/writeup.jsp?docid=2014-051514-5659-99`.

[116] F. Callegati, W. Cerroni, and M. Ramilli. Man-in-the-middle attack to the https protocol. *IEEE Security and Privacy*, 1:78–81, 2009.

[117] US-CERT. Lenovo computers vulnerable to https spoofing, 2015. URL `https://www.us-cert.gov/ncas/current-activity/2015/02/20/Lenovo-Computers-Vulnerable-HTTPS-Spoofing`.

[118] M. Mannan and P. C. van Oorschot. On instant messaging worms, analysis and countermeasures. In *Proceedings of the 2005 ACM workshop on Rapid malcode*, pages 2–11. ACM, 2005.

[119] S. Ford, M. Cova, C. Kruegel, and G. Vigna. Analyzing and detecting malicious flash advertisements. In *Proceedings of the Annual Computer Security Applications Conference*, ACSAC'09, pages 363–372. IEEE, 2009.

[120] A. Kalafut, A. Acharya, and M. Gupta. A study of malware in peer-to-peer networks. In *Proceedings of 6th ACM SIGCOMM conference on Internet measurement*, pages 327–332. ACM, 2006.

[121] J. R. Jacobs. Measuring the effectiveness of the usb flash drive as a vector for social engineering attacks on commercial and residential computer systems. Master's thesis, Embry-Riddle Aeronautical University, 2011.

[122] A. Gazet. Comparative analysis of various ransomware virii. *Journal in computer virology*, 1:77–90, 2010.

[123] P. Ducklin. Anatomy of an android sms virus - watch out for text messages, even from your friends!, 2014. URL `https://nakedsecurity.sophos.com/2014/06/29/anatomy-of-an-android-sms-virus-watch-out-for-text-messages-even-from-your-friends/`.

[124] J. Hong. The state of phishing attacks. *Communications of the ACM*, 55(1): 74–81, 2012.

[125] Arstechnica. Phishing scam that penetrated wall street just might work against you, too, 2014. URL `http://arstechnica.com/security/2014/12/phishing-scam-that-penetrated-wall-street-just-might-work-against-you-too/`.

[126] A. Raskin. Tabnabbing: A new type of phishing attack, 2011. URL `http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/`.

[127] A. Adelsbach, S. Gajek, and J. Schwenk. Visual spoofing of ssl protected web sites and effective countermeasures. In *Information Security Practice and Experience*, pages 204–216. Springer Berlin Heidelberg, 2005.

[128] L. Phifer. Top ten wi-fi security threats, 2000. URL `http://www.esecurityplanet.com/views/article.php/3869221/Top-Ten-WiFi-Security-Threats.htm`.

[129] SensePost. Snoopy, 2014. URL `https://github.com/sensepost/Snoopy`.

[130] M. Eeckhaut and N. Vanhecke. De standaard: Belgian professor in cryptography hacked, 2014. URL `http://www.standaard.be/cnt/dmf20140201_011`.

[131] M. Hasan and N.B. Prajapati. An attack vector for deception through persuasion used by hackers and crackers. In *Networks and Communications*, NETCOM'09. First International Conference on, pages 254–258. IEEE, 2009.

[132] K. Mitnick and W. L. Simon. *The art of deception: controlling the human element of security*. Wiley, 2001.

[133] G. Hinson. Social engineer techniques, risks and controls. *The EDP Audit, Control and Security Newsletter*, 4-5:32–46, 2008.

[134] R. Kuipers, E. Starck, and H. Heikkinen. Smart tv hacking: Crash testing your home entertainment, 2010. URL `http://www.codenomicon.com/resources/whitepapers/codenomicon-wp-smart-tv-fuzzing.pdf`.

[135] C. Colwill. Human factors in information security: The insider threat: Who can you trust these days? *Information security technical report*, 4:186–196, 2009.

[136] A. Calder and S. Watkins. *IT Governance: An International Guide to Data Security and ISO27001/ISO27002*. Kogan Page Publishers, 2010.

[137] ISACA. *COBIT 5 for Information Security*. ITGI, 2012.

[138] S. M. Ali. Integration of information security essential controls into information technology infrastructure library - a proposed framework. *International Journal of Applied Science and Technology*, 1, 2014.

[139] GOVUK. 10 steps to cyber security, 2015. URL `https://www.gov.uk/government/publications/cyber-risk-management-a-board-level-responsibility/10-steps-summary`.

[140] CPNI. Social engineering: Understanding the threat, 2013. URL `http://www.cpni.gov.uk/documents/publications/2013/2013065-social-engineering.pdf?epslanguage=en-gb`.

[141] A. Calder and S. Watkins. *Framework for Improving Critical Infrastructure Cybersecurity*. National Institute of Standards and Technology (NIST) and United States of America, 2014.

[142] W. Jansen and T. Grance. Guidelines on security and privacy in public cloud computing. Technical report, 2011.

[143] R. T. Peltier. *Information Security Policies, Procedures, and Standards: guidelines for effective information security management*. CRC PRess, 2013.

[144] E.D. Frauenstein and R.von Solms. An enterprise anti-phishing framework. In *Information Assurance and Security Education and Training (WISE 2009), IFIP Advances in Information and Communication Technology*, pages 196–203. Springer Berlin Heidelberg, 2013.

[145] B. D Cone, C. E Irvine, M. F. Thompson, and T. D. Nguyen. A video game for cyber security training and awareness. *Computer and Security*, 1:63–72, 2007.

[146] N. A. G. Arachchilage, S. Love, and M. Scott. Designing a mobile game to teach conceptual knowledge of avoiding phishing attacks. *International Journal for e-Learning Security*, 2:127–132, 2012.

[147] S. Sheng, B. Magnien, P. Kumaraguru, A. Acquisti L. F. Cranor, J. Hong, and E. Nunge. Anti-phishing phil: the design and evaluation of a game that teaches people not to fall for phish. In *Proceedings of the 3rd symposium on Usable privacy and security*, pages 88–99. ACM, 2007.

[148] A. H. Kruger and D. K. Wayne. A prototype for assessing information security awareness. *Computers and Security*, 4:289–296, 2006.

[149] I. Kirlappos and M.A Sasse. Security education against phishing: A modest proposal for a major rethink. *IEEE Security and Privacy Magazine*, 2:24–32, 2012.

[150] M. Wu R. C. Miller and S. L. Garfinkel. Do security toolbars actually prevent phishing attacks? In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 601–610. ACM, 2006.

[151] S. E. Schechter, R. Dhamija, A. Ozment, and I. Fischer. The emperor's new security indicators. In *IEEE Symposium on Security and Privacy, 2007*, pages 51–65. IEEE, 2007.

[152] E. Lin, S. Greenberg, E. Trotter, D. Ma, and J. Aycock. Does domain highlighting help people identify phishing sites? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2075–2084. ACM, 2011.

[153] J. Lee, L. Bauer, and M. L. Mazurek. The effectiveness of security images in internet banking. *Internet Computing*, 19(1):54–62, 2015.

[154] S. Sheng, M. Holbrook, P. Kumaraguru, L.F Cranor, and J. Downs. Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of interventions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 373–382, Cambridge, United Kingdom, April 2010. ACM.

[155] A. Doupe, M. Egele, B. Caillat, G. Stringhini, G. Yakin, A. Zand, and G. Vigna. Hit'em where it hurts: a live security exercise on cyber situational awareness. In *Proceedings of the 27th Annual Computer Security Applications Conference*, pages 51–61. ACM, 2011.

[156] I. Burke W. A. Labuschagne, N. Veerasamy and M. M. Eloff. Design of cyber security awareness game utilizing a social media framework. In *Information Security South Africa (ISSA)*. IEEE, 2011.

[157] I. Gulenko. Social against social engineering: Concept and development of a facebook application to raise security and risk awareness. *Information Management and Computer Security*, 2:91–101, 2013.

[158] E. Kritzinger and S. H. von Solms. Cyber security for home users: A new way of protection through awareness enforcement. *Computer and Security*, 8:840–847, 2010.

[159] K. E. Stewart, J. W. Humphries, and T. R. Andel. Developing a virtualization platform for courses in networking, systems administration and cyber security education. In *Proceedings of the 2009 Spring Simulation Multiconference. Society for Computer Simulation International*, 2009.

[160] A. Konak and M. Bartolacci. Broadening e-commerce information security education using virtual computing technologies. In *2012 Networking and Electronic Commerce Research Conference*, 2012.

[161] B. B. Anderson, C. B. Kirwan, J. L. Jenkins, D. Eargle, S. Howard, and A. Vance. How polymorphic warnings reduce habituation in the braininsights from an fmri study. In *Proceedings of of CHI15*, 2013.

[162] A. Neupane, N. Saxena, K. Kuruvilla, and M. Georgescu R. Kana. Neural signatures of user-centered security: An fmri study of phishing, and malware warnings. In *Proceedings of the Network and Distributed System Security Symposium*, pages 1–16. NDSS, 2014.

[163] G. Schaff, C. Harpes, R. Martin, and M. Junger. An application to estimate the cyber-risk detection skill of mobile device users (idea), 2013. URL `http://doc.utwente.nl/87117/1/SCHAFF_itrust-scientific_article_GSC_(3).pdf`.

[164] S. Guillaume, H. Carlo, A. Matthieu, J. Marianne, and M. Romain. Risk-det: Ict security awareness aspect combining education and cognitive sciences. In *ICCGI 2014, The Ninth International Multi-Conference on Computing in the Global Information Technology*, pages 51–53, June 2014.

[165] T. Bakhshi, M. Papadaki, and S. Furnell. Social engineering: assessing vulnerabilities in practice. *Information management and computer security,*, 1:53–63, 2009.

[166] M. Eminagaoglu, E. Ucar, and S. Eren. The positive outcomes of information security awareness training in companies a case study. *Information security technical report*, 4:223–229, 2009.

[167] K. F. McCrohan, K. Engel, and J. W. Harvey. Influence of awareness and training on cyber security. *Journal of Internet Commerce*, 1:23–41, 2010.

[168] X. Leroy. Java bytecode verification: an overview. In *Computer Aided Verification*. Springer Berlin Heidelberg, 2001.

[169] A. Barth, C. Jackso, C. Reis, and TGC Team. The security architecture of the chromium browser, 2008. URL `http://seclah.stanford.edu/websec/chromium`.

[170] H. Xiao and B. Zhao. Analysis on sandbox technology of adobe reader x. In *Computational and Information Sciences (ICCIS)*, 5th International Conference on. IEEE, 2013.

[171] D. S. Peterson, M. Bishop, and R. Pandey. A flexible containment mechanism for executing untrusted code. In *Proceedings of the 11th USENIX Security Symposium*, 5th International Conference on, pages 207–225. IEEE, 2002.

[172] E. F. Brickell, J. F. Cihula, C. D. Hall, and R. Uhlig. Method of improving computer security through sandboxing. US patent No. 7,908,653, 2011.

[173] Chromium. The chromium projects - sandbox, 2015. URL `http://www.chromium.org/developers/design-documents/sandbox`.

[174] Mozilla Firefox. Mozilla wiki - security/sandbox, 2015. URL `https://wiki.mozilla.org/Security/Sandbox`.

[175] Comodo. Demo of a url-bar spoofing attack, 2015. URL `http://www.contentverification.com/graphic-attacks/demo/`.

[176] G. W. Romney, J. K. Jones, B. L. Rogers, and P. MacCabe. It security education is enhanced by analyzing honeynet data. In *Information Technology Based Higher Education and Training*, ITHET 2005. 6th International Conference on. IEEE, 2005.

[177] Microsoft. The windows vista and windows server 2008 developer story: Windows vista application development requirements for user account control, 2007. URL `https://msdn.microsoft.com/en-us/library/aa905330.aspx`.

[178] L. Lu, V. Yegneswaran, P. Porras, and W. Lee. Blade: an attack-agnostic approach for preventing drive-by malware infections. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 440–450. ACM, 2010.

[179] Invincea. Sandboxie, 2014. URL `http://www.sandboxie.com/`.

[180] BufferZone Pro. Bufferzone-pro, 2014. URL `http://www.trustware.com/BufferZone-Pro/`.

[181] Qubes OS. Qubes os project, 2015. URL `https://www.qubes-os.org/`.

[182] A. Bianchi, J. Corbetta, L. Invernizzi, Y. Fratantonio, C. Kruegel, and G. Vigna. What the app is that? deception and countermeasures in the android user interface. In *36th IEEE Symposium on Security and Privacy*. IEEE, 2015.

[183] M. Egele, D. Brumley Y. Fratantonio, and C. Kruegel. An empirical study of cryptographic misuse in android applications. In *Proceedings of the 2013 ACM SIGSAC conference on Computer and communications security*, pages 73–84. ACM, 2013.

[184] S. Motiee, K. Hawkey, and K. Beznosov. Do windows users follow the principle of least privilege?: investigating user account control practices. In *Proceedings of the Sixth Symposium on Usable Privacy and Security*. ACM, 2010.

[185] B. C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *Communications Magazine*, 9:33–38, 1994.

[186] B. Desmond, J. Richards, R. Allen, and A. G. Lowe-Norris. *Active Directory: Designing, Deploying, and Running Active Directory.* O'Reilly Media, 2008.

[187] B. Turner, D. Lundell, J. Zamora, and C. Calderon. Microsoft forefront identity manager 2010 technical overview. Technical report, 2010.

[188] Y. Boshmaf, I. Muslukhov, and K. Beznosov M. Ripeanu. Key challenges in defending against malicious socialbots. In *Proceedings of the 5th USENIX Conference on Large-scale Exploits and Emergent Threats, LEET (Vol. 12)*, 2012.

[189] K. Lee, J. Caverlee, and S. Webb. The social honeypot project: protecting online communities from spammers. In *Proceedings of the 19th international conference on World wide web*. ACM, 2010.

[190] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: social honeypots+ machine learning. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 435–442. ACM, 2010.

[191] M. B. Salem and S. J. Stolfo. Modeling user search behavior for masquerade detection. In *Recent Advances in Intrusion Detection*, pages 181–200. Springer Berlin Heidelberg, Jan 2011.

[192] M. Ruskov, P Ekblom, and M. A. Sasse. Towards a simulation of information security behaviour in organisations. In *Cyberpatterns*, pages 177–184. Springer International Publishing, 2014.

[193] G. Stringhini, C. Kruegel, and G. Vigna. Shady paths: Leveraging surfing crowds to detect malicious web pages. In *Proceedings of the 2013 ACM SIGSAC conference on Computer and communications security*, pages 133–144. ACM, 2013.

[194] L. Lu, R. Perdisci, and W. Lee. Surf: detecting and measuring search poisoning. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 467–476. ACM, 2011.

[195] Z. Li, S. Alrwais, Y. Xie, F. Yu, and X. Wang. Finding the linchpins of the dark web: a study on topologically dedicated hosts on malicious web infrastructures. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 112–126. IEEE, 2013.

[196] S. Lee and J. Kim. Warningbird: Detecting suspicious urls in twitter stream. In *NDSS*. NDSS, 2012.

[197] Z.H. Abdullah, N. I. Udzir, R. Mahmod, and K. Samsudin. Towards a dynamic file integrity monitor through a security classification. *Internal Journal of New Computer Architectures and their Applications (IJNCAA)*, 3:766–779, 2011.

[198] R. Dhanalakshmi and C. Chellappan. Detection and recognition of file masquerading for e-mail and data security. In *Recent Trends in Network Security and Applications*, pages 253–262. Springer Berlin Heidelberg, 2010.

[199] M. Hara, A. Yamada, and Y. Miyake. Visual similarity-based phishing detection without victim site information. In *Computational Intelligence in Cyber Security, 2009. CICS'09. IEEE Symposium on*, pages 30–36. IEEE, 2009.

[200] T. Bhardwaj, K. T. Sharma, and M. R. Pandit. Social engineering prevention by detecting malicious urls using artificial bee colony algorithm. In *Proceedings of the Third International Conference on Soft Computing for Problem Solving*, pages 355–363. Springer India, 2014.

[201] C. Seifert, J. W. Stokes, C. Colcernian, J. C. Platt, and L. Lu. Robust scareware image detection. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 2920–2924. IEEE, 2013.

[202] C. Dietrich. *Identification and Recognition of Remote-Controlled Malware*. PhD thesis, Universittsbibliothek Mannheim, 2013.

[203] Darknet. Evilap defender detect evil twin attacks, 2015. URL `http://www.darknet.org.uk/2015/04/evilap-defender-detect-evil-twin-attacks/`.

[204] L. Alvisi, A. Clement, A. Epasto, S. Lattanzi, and A. Panconesi. Sok: The evolution of sybil defense via social networks. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 382–396. IEEE, 2013.

[205] M. Chandrasekaran and K. Narayanan S. Upadhyaya. Phishing email detection based on structural properties. In *NYS Cyber Security Conference*, pages 1–7, 2006.

[206] P. Singhal and N. Raul. Malware detection module using machine learning algorithms to assit in centralized security in enterprise networks. *International Journal 4*, 2012.

[207] D. Gavrilut, M. Cimpoesu, D. Anton, and L. Ciortuz. Malware detection using machine learning. In *Computer Science and Information Technology*, IMCSIT'09. International Multiconference on, pages 735–741. IEEE, 2009.

[208] S. Dong-Her, C. Hsiu-Sen, C. Chun-Yuan, and B. Lin. Internet security: malicious e-mails detection and protection. *Industrial Management and Data Systems*, 7:613–623, 2011.

[209] H. Sandouka, A. J. Cullen, and I. Mann. Social engineering detection using neural networks. In *CyberWorlds CW'09 International Conference on*, pages 273–278. IEEE, 2009.

[210] H. Drucker, S. Wu, and V. N. Vapnik. Support vector machines for spam categorization. *Neural Networks, IEEE Transactions on*, 5:1048–0154, 1999.

[211] I. Fette, N. Sadeh, and A. Tomasic. Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, pages 649–656. ACM, 2007.

[212] R. Basnet, S. Mukkamala, and A. H. Sung. Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications in Industry*, pages 373–383. Springer Berlin Heidelberg, 2008.

[213] S. Garera, N. Provos, M. Chew, and A. D. Rubin. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 1–8. ACM, 2007.

[214] A. Bergholz, J. H. Chang, G. Paa, F. Reichartz, and S. Strobel. Improved phishing detection using model-based features. In *CEAS*, 2008.

[215] A. Bergholz, J. De Beer, S. Glahn, M. F. Moens, G. Paa, and S. Strobel. New filtering approaches for phishing email. *Journal of computer security*, 1:7–35, 2010.

[216] V. Raskin, J. M. Taylor, and C. F. Hempelmann. Ontological semantic technology for detecting insider threat and social engineering. In *Proceedings of the 2010 workshop on New security paradigms*. ACM, 2010.

[217] G. Xiang, J. Hong, C. P. Rose, and L. Cranor. Cantina+: a feature-rich machine learning framework for detecting phishing web sites. *ACM Transactions on Information and System Security (TISSEC)*, 2:21, 2011.

[218] A. Aggarwal, A. Rajadesingan, and P. Kumaraguru. Phishari: Automatic real-time phishing detection on twitter. In *eCrime Researchers Summit (eCrime)*, pages 1–12. IEEE, 2012.

[219] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song. Design and evaluation of a real-time url spam filtering service. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 447–462. IEEE, 2011.

[220] G. Stringhini and O. Thonnard. That aint you: Blocking spearphishing through behavioral modelling. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 78–97. Springer, 2015.

[221] T. Halevi, N. Memon, and O. Nov. Spear-phishing in the wild: A real-world study of personality, phishing self-efficacy and vulnerability to spear-phishing attacks. January 2015. URL `http://dx.doi.org/10.2139/ssrn.2544742`.

[222] T. Halevi, J. Lewis, and N. Memon. A pilot study of cyber security and privacy related behavior and personality traits. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 737–744, May 2013.

[223] C. Whittaker, B. Ryner, and M. Nazif. Large-scale automatic classification of phishing pages. *NDSS*, 10, 2010.

[224] A. Calder and S. Watkins. *IT Governance: an international guide to data security and ISO27001/ISO27002*. Kogan Page Publishers, 2012.

[225] P. Kumaraguru, J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A.Blair, and T. Pham. School of phish: a real-world evaluation of anti-phishing training. In *In Proceedings of the 5th Symposium on Usable Privacy and Security*, July 2009.

[226] A. P. Felt, R. W. Reeder, H. Almuhimedi, and S. Consolvo. Experimenting at scale with google chrome's ssl warning. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2667–2670. ACM, 2014.

[227] M. Khonji, Y. Iraqi, and A. Jones. Phishing detection - a literature survey. *Communications Surveys and Tutorials IEEE*, 15(4):2091–2121, 2013.

[228] K. Rieck, T. Holz, C. Willems, P. Dssel, and P. Laskov. Learning and classification of malware behavior. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 108–125. Springer Berlin Heidelberg, 2008.

[229] A. Filippoupolitis, G. Loukas, and S. Kapetanakis. Towards real-time profiling of human attackers and bot detection. In *7th International Conference on Cybercrime Forensics Education and Training*, Canterbury, United Kingdom, July 2014.

[230] S. Kapetanakis, A. Filippoupolitis, G. Loukas, and T.S. Al Murayziq. Profiling cyber attackers using case-based reasoning. In *19th UK workshop on Case-Based Reasoning (UK-CBR 2014)*, Cambridge, United Kingdom, Dec 2014.

[231] M. Kandias, V. Stavrou, N. Bozovic, and D. Gritzalis. Proactive insider threat detection through social media: The youtube case. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, pages 261–266, Nov 2013.

[232] R. R. McCrae and J. P. Oliver. An introduction to the five-factor model and its applications. *Journal of Personality*, 60(2):175–215, 1992.

[233] R. B. Cialdini. *Influence: Science and practice.* Boston: Allyn and Bacon, 2001.

[234] J. G. Mohebzada, A. El Zarka, A. H. Bhojani, and A. Darwish. Phishing in a university community: Two large scale phishing experiments. In *Innovations in Information Technology (IIT), 2012 International Conference on*, pages 373–382, Cambridge, United Kingdom, April 2012.

[235] M. Workman. Wisecrackers: A theorygrounded investigation of phishing and pretext social engineering threats to information security. *Journal of the American Society for Information Science and Technology*, 59(4):662–674, 2008.

[236] I. M. A. Alseadon. *The impact of users' characteristics on their ability to detect phishing emails.* PhD thesis, Queensland University of Technology, Brisbane, May 2014. URL `http://eprints.qut.edu.au/72873/1/Ibrahim%20Mohammed%20A_Alseadoon_Thesis.pdf`.

[237] A. Karakasiliotis, S. M . Furnell, , and M. Papadaki. Assessing end-user awareness of social engineering and phishing. In *7th Australian Information Warfare and Security Conference*, 2006.

[238] J. S. Downs, M. B. Holbrook, and L. F. Cranor. Decision strategies and susceptibility to phishing. In *Proceedings of the second symposium on Usable privacy and security*, pages 79–90. ACM, 2006.

[239] J. S. Downs, M. Holbrook, and L. F. Cranor. Behavioral response to phishing risk. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pages 37–44, October 2007.

[240] M. Jakobsson, A. Tsow, A. Shah, E. Blevis, and Y. K. Lim. What instills trust? a qualitative study of phishing. In *Financial Cryptography and Data Security*, pages 356–361. Springer Berlin Heidelberg, 2007.

[241] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1065–1074. ACM, 2008.

[242] A. Vishwanath, T. Herath, R. Chen, J. Wang, and H. R. Rao. Why do people get phished? testing individual differences in phishing vulnerability within an integrated, information processing model. *Decision Support Systems*, 51(3): 576–586, 2011.

[243] M. Blythe, H. Petrie, and J. A. Clark. F for fake: four studies on how we fall for phish. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3469–3478, May 2011.

[244] M. Renaud. System and method for dynamically assessing security risks attributed to a computer user's behavior, December 2007.

[245] N. Stembert, A. Padmos, S. M. Bargh, S. Choenni, and F. Jansen. A study of preventing email (spear) phishing by enabling human intelligence. In *Intelligence and Security Informatics Conference (EISIC)*, pages 113–120. IEEE, 2015.

[246] Great Britain. *Equality act 2010.* Editions de l'Atelier, 2010.

[247] L. Weber and E. Dwoskin. Are workplace personality tests fair? *Wall Street Journal*, 2014.

[248] R. T. Wright and K. Marett. The influence of experiential and dispositional factors in phishing: an empirical investigation of the deceived. *Journal of Management Information Systems*, 27(1):273–303, 2010.

[249] N. Davinson and E. Sillence. It wont happen to me: Promoting secure behaviour among internet users. *Computers in Human Behavior*, 26(6):1739–1747, 2010.

[250] P. G. Nyeste and C. B. Mayhorn. Training users to counteract phishing. In *Proceedings of the Human Factors and Ergonomics Society Annual Meetings*, pages 1956–1960. Sage Publications, 2010.

[251] C. Wagner, S. Mitter, C. Krner, and M. Strohmaier. When social bots attack: Modeling susceptibility of users in online social networks. *Making Sense of Microposts*, 2, 2012.

[252] J. Wang, T. Herath, R. Chen, A. Vishwanath, and H. R. Rao. Research article phishing susceptibility: An investigation into the processing of a targeted spear phishing email. *IEEE Transactions on Professional Communication*, 55(4):345–362, 2012.

[253] P. A. Wang. Assessment of cyber security knowledge and behavior: An anti-phishing scenario. In *International Conference on Internet Monitoring and Protection (ICIMP)*, pages 1–7. IEEE, 2013.

[254] X. R. Luo, W. Zhang, S. Burd, and A. Seazzu. Investigating phishing victimization with the heuristic systematic model: A theoretical framework and an exploration. *Computers and Security*, 38:29–38, 2017.

[255] R. T. Wright, M. L. Jensen, J. B. Thatcher, M. Dinger, and K. Marett. Research note-influence techniques in phishing attacks: An examination of vulnerability and resistance. *Information Systems Research*, 25(2):385–400, 2014.

[256] G. Canova. Design, implementation and evaluation of an anti-phishing education app. Master's thesis, Technische Universitat Darmstadt, 2014.

[257] M. Alsharnouby, F. Alaca, and S. Chiasson. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies*, 82(C):69–82, 2015.

[258] A. K. Welk, K. W. Hong, O. A. Zielinska, R. Tembe, E. Murphy-Hill, and C. B. Mayhorn. Will the phisher-men reel you in?: Assessing individual differences in a phishing detection task. *International Journal of Cyber Behavior, Psychology and Learning (IJCBPL)*, 5(4):1–17, 2015.

[259] T. Kelley and B. I. Bertenthal. Attention and past behavior, not security knowledge, modulate users decisions to login to insecure websites. *Information and Computer Security*, 24(2), 2016.

[260] J. Wang, Y. Li, and H. R. Rao. Overconfidence in phishing email detection. *Journal of the Association for Information Systems*, 17(11), 2016.

[261] PhishMe. Enterprise phishing susceptibility report, 2016. URL `http://www.phishme.com/resources/whitepapers-and-brochures/`.

[262] M. Volkamer, K. Renaud, B. Reinheimer, and A. P. Kunz. User experiences of torpedo: Tooltip-powered phishing email detection. *Computers and Security*, 2017.

[263] H. S. Rhee, C. Kim, and Y. U. Ryu. Self-efficacy in information security: Its influence on end users information security practice behavior. *Computers and Security*, 28(8):816–826, 2009.

[264] N. A. G. Arachchilage and S. Love. Security awareness of computer users: A phishing threat avoidance perspective. *Computers in Human Behavior*, pages 304–312, 2014.

[265] A. Luszczynska, U. Scholz, and R. Schwarzer. The general self-efficacy scale: multicultural validation studies. *The Journal of psychology*, 139(5):439–457, 2005.

[266] G. Chen, S. M. Gully, and D. Eden. Validation of a new general self-efficacy scale. *Organizational research methods*, 4(1):62–83, 2001.

[267] D. Colardyn and J. Bjornavold. Validation of formal, nonformal and informal learning: Policy and practices in eu member states. *European journal of education*, 39(1):69–89, 2004.

[268] B. D. Cullity. *Elementary signal detection theory.* Oxford University Press, Los Angeles, 2001.

[269] H. Abdi. Signal detection theory (sdt). *Encyclopedia of measurement and statistics*, pages 886–889, 2007.

[270] L. Li, D. Z. Yang, and F. C. Shen. A novel rule-based intrusion detection system using data mining. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, pages 169–172. IEEE, July 2010.

[271] J. Nahar, T. Imam, K. S. Tickle, and Y. P. P. Chen. Association rule mining to detect factors which contribute to heart disease in males and females. *Expert Systems with Applications*, 40(4):1086–1093, 2013.

[272] W. Lin, S. A. Alvarez, and C. Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data mining and knowledge discovery*, 6(1): 83–105, 2002.

[273] W. S. Yang, J. B. Dia, H. C. Cheng, and H. T. Lin. Mining social networks for targeted advertising. In *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*. IEEE, January 2006.

[274] P. N. Tan, M. Steinbach, and V. Kumar. Association analysis: basic concepts and algorithms. In *Introduction to data mining*, chapter 6, pages 327–414. Pearson Education, 2002.

[275] R. Ihaka and R. Gentleman. The R project for statistical computing, 2016. URL `https://www.r-project.org/`.

[276] M. Kuhn. Caret package. *Journal of Statistical Software*, 28(5), 2008.

[277] D. W. Hosmer and S. Lemeshow, editors. *Applied logistic regression*. John Wiley & Sons, 2004.

[278] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[279] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[280] M. Fernandez-Delgado, E. Cernadas, S. Barro, and D. Amorim. Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1):3133–3181, 2014.

[281] M. Kuhn. Variable selection using the caret package, 2012. URL `http://cran.cermin.lipi.go.id/web/packages/caret/vignettes/caretSelection.pdf`.

[282] C. Ambroise and G. J. McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. *Proceedings of the national academy of sciences*, 99(10):6562–6566, 2002.

[283] L. Myers and M. J. Sirois, editors. *Spearman correlation coefficients, differences between.* Wiley StatsRef: Statistics Reference Onlines, 2006.

[284] U. Olsson, F. Drasgow, and N. J. Dorans. The polyserial correlation coefficient. *Psychometrika*, 47(3):337–347, 1982.

[285] Symantec. Istr20: Internet security threat report, 2015. URL `https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf`.

[286] P. Finn and M. Jakobsson. Designing and conducting phishing experiments. *IEEE Tech. Soc.*, 6(2):66–68, 2008.

[287] F. Mouton, M. M. Malan, and H. S. Venter. Social engineering from a normative ethics perspective. In *Information Security for South Africa*, pages 1–8. IEEE, 2013.

[288] M. Jakobsson and J. Ratkiewicz. Designing ethical phishing experiments: a study of (rot13) ronl query features. In *Proceedings of the 15th international conference on World Wide Web*, pages 513–522. ACM, 2006.

[289] C. Soghoian. Legal risks for phishing researchers. In *eCrime Researchers Summit*, pages 1–11. IEEE, 2008.

[290] Phishtank. Out of the web and into the tank, 2015. URL `https://www.phishtank.com/`.

[291] Scamdex. The internet scam resource, 2015. URL `http://www.scamdex.com/`.

[292] Millersmiles. The web's dedicated anti-phishing service, 2015. URL `http://www.millersmiles.co.uk/`.

[293] Y. Zheng, T. Liu, Y. Wang, Y. Zhu, Y. Liu, and E. Chang. Diagnosing new york city's noises with ubiquitous data. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 715–725, Sep. 2014.

[294] E.H. Jürrens, A. Bröring, and S. Jirkai. A human sensor web for water availability monitoring. In *OneSpace*, 2009.

[295] M. Avvenuti, M. G. Cimino, S. Cresci, A. Marchetti, and M. Tesconi. A framework for detecting unfolding emergencies using humans as sensors. *SpringerPlus*, 5(1):1–23, 2016.

[296] R. M. Lee, M. J. Assante, and T. Conway. Analysis of the cyber attack on the ukrainian power grid., 2016. URL `http://www.nerc.com/pa/CI/ESISAC/Documents/E-ISAC_SANS_Ukraine_DUC_18Mar2016.pdf`.

[297] University of Oxford. Information security - report an incident, 2016. URL `https://www.infosec.ox.ac.uk/report-incident`.

[298] BBC News. Fake news: Facebook rolls out new tools to tackle false stories, 2016. URL `http://www.bbc.co.uk/news/world-us-canada-38336212`.

[299] PhishMe. Phishme reporter, 2016. URL `https://phishme.com/product-services/reporter`.

[300] Wombat Security. Wombat security announces new feature to reinforce secure employee behavior against phishing, 2016. URL `https://www.wombatsecurity.com/press-releases/phishalarm-email-add-in`.

[301] Sophos. Sophos phish threat, 2017. URL `https://www.sophos.com/products/phish-threat.aspx`.

[302] M. A. Sasse, C. C. Palmer, M. Jakobsson, S. Consolvo, R. Wash, and L. J. Camp. Helping you protect you. *IEEE Security and Privacy*, 12(1):39–42, 2014.

[303] L. Malisa, K. Kostiainen, and S. Capkun. Detecting mobile application spoofing attacks by leveraging user visual similarity perception. *IACR Cryptology ePrint Archive*, 2015.

[304] R. Heartfield, G. Loukas, and D. Gan. You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks. *IEEE Access*, 4:6910–6928, 2016.

[305] Ryan Heartfield and George Loukas. Evaluating the reliability of users as human sensors of social media security threats. In *International Conference On Cyber Situational Awareness, Data Analytics And Assessment (CyberSA)*, pages 1–7. IEEE, 2016.

[306] M. A. Sasse and M. Smith C. Herley H. Lipford K. Vaniea. Debunking security-usability tradeoff myths. *IEEE Security and Privacy*, 14(5):33–39, 2016.

[307] Birch Grove Software Inc. Activtrak, 2017. URL `https://activtrak.com/`.

[308] Yahoo. Secure your inbox, 2017. URL `https://uk.antispam.yahoo.com/`.

[309] Engadget. Google beefs up gmail security to fight phishing attempts, 2017. URL `https://www.engadget.com/2017/05/31/google-gmail-security-fight-phishing/`.

[310] Microsoft. Office 365 email anti-spam protection, 2017. URL `https://support.office.com/en-gb/article/Office-365-email-anti-spam-protection-6a601501-a6a8-4559-b2e7-56b59c96a586`.

[311] ProtonMail. Effective spam filtering with encrypted email, 2017. URL `https://protonmail.com/blog/encrypted-email-spam-filtering/`.

[312] ESET. Eset anti-phishing, 2017. URL `https://www.eset.com/us/anti-phishing/`.

[313] GMX. Spam filter: The cleanest inbox, 2017. URL https://www.gmx.com/mail/spam-filter/.

[314] Mail. Stay safe from phishing: Your worry free email, 2017. URL https://www.mail.com/mail/spam-filter/499562-stay-safe-phishing-email.html.

[315] Firefox. How does phishing and malware protection work, 2017. URL https://support.mozilla.org/en-US/kb/how-does-phishing-and-malware-protection-work.

[316] Chrome. Google safe browsing, 2017. URL https://safebrowsing.google.com/.

[317] Opera. Opera fraud and malware protection, 2017. URL http://www.opera.com/help/tutorials/security/fraud/.

[318] Comodo. Dragon internet browser, 2017. URL https://www.comodo.com/home/browsers-toolbars/browser.php#tab-features.

[319] Avast. Safezone browser, 2017. URL https://www.avast.com/f-safezone.

[320] Microsoft. Secucrity enhancements for microsoft edge, 2017. URL https://docs.microsoft.com/en-us/microsoft-edge/deploy/security-enhancements-microsoft-edge.

[321] Apple. Defending your online privacy and security., 2017. URL https://www.apple.com/uk/safari/.

[322] Comodo. Comodo cloud antivirus, 2017. URL https://antivirus.comodo.com/cloud-antivirus.php.

[323] AVG. Avg anti-virus free, 2017. URL http://www.avg.com/en-gb/free-antivirus-download.

[324] Avast Internet Security. Avast internet security, 2017. URL https://www.avast.com/en-gb/internet-security.

[325] Microsoft. Windows defender smartscreen, 2017. URL https://docs.microsoft.com/en-us/windows/threat-protection/windows-defender-smartscreen/windows-defender-smartscreen-overview.

[326] Symantec. Norton security review 2017: Top antivirus provider with fully furnished internet security suites, 2017. URL https://fatsecurity.com/review/norton.

[327] Kaspersky. Kaspersky internet security 2017, 2017. URL https://www.kaspersky.co.uk/internet-security.

[328] Sophos. Intercept x tech specs, 2017. URL https://www.sophos.com/en-us/products/intercept-x/tech-specs.aspx.

[329] Facebook. What can i do about phishing?, 2017. URL https://www.facebook.com/help/166863010078512?helpref=faq_content.

[330] TipTopSecurity. Is google drive safe to use? how google secures your files online, 2016. URL https://tiptopsecurity.com/is-google-drive-safe-to-use/.

[331] Microsoft. Mitigate threats by using windows 10 security features, 2017. URL https://docs.microsoft.com/en-us/windows/threat-protection/overview-of-threat-mitigations-in-windows-10.

[332] L. Ross, L. Irani, M. Silberman, A. Zaldivar, and B. Tomlinson. Who are the crowdworkers? shifting demographics in mechanical turk. In *CHI'10 extended abstracts on Human factors in computing systems*, pages 2863–2872. ACM, 2010.

[333] M. Marge, S. Banerjee, and A. I. Rudnicky. Using the amazon mechanical turk for transcription of spoken language. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 5270–5273. IEEE, 2010.

[334] D. W. Barowy, C. Curtsinger, E. D. Berger, and A. McGregor. Automan: A platform for integrating human-based and digital computation. *Communications of the ACM*, 59(6):102–109, 2016.

[335] W. S. Lasecki, C. D. Miller, I. Naim, R. Kushalnagar, A. Sadilek, D. Gildea, and J. P. Bigham. Scribe: Deep integration of human and machine intelligence to caption speech in real time. *Communications of the ACM*, 60(11), 2017.

[336] K. Krol, J. M. Spring, S. Parkin, and M. A. Sasse. Towards robust experimental design for user studies in security and privacy. In *Learning from Authoritative Security Experiment Results (LASER) Workshop*, 2016.

[337] S. S. Rahman, R. Heartfield, W. Oliff, G. Loukas, and A. Filippoupolitis. Assessing the cyber-trustworthiness of human-as-a-sensor reports from mobile devices. In *Software Engineering Research, Management and Applications (SERA 2017), 15th ACIS International Conference on*. IEEE, June 2017.

[338] B. Pan, Y. Zheng, D. Wilkie, and C. Shahabi. Crowd sensing of traffic anomalies based on human mobility and social media. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 344–353. ACM, 2013.

[339] R. Dave, S. K. Boddhu, M. McCartney, and J. West. Augmenting situational awareness for first responders using social media as a sensor. *IFAC Proceedings Volumes*, 46(15):133–140, 2013.