Protection Against Semantic Social Engineering Attacks

Ryan Heartfield and George Loukas

Abstract Phishing, drive-by downloads, file and multimedia masquerading, domain typosquatting, malvertising and other semantic social engineering attacks aim to deceive the user rather than exploit a technical flaw to breach a system's security. We start with a chronological overview to illustrate the growing prevalence of such attacks from their early inception 30 years ago, and identify key milestones and indicative trends which have established them as primary weapons of choice for hackers, cyber-criminals and state actors today. To demonstrate the scale and widespread nature of the threat space, we identify over 35 individually recognised types of semantic attack, existing within and cross-contaminating between a vast range of different computer platforms and user interfaces. Their extreme diversity and the little to no technical traces they leave make them particularly difficult to protect against. Technical protection systems typically focus on a single attack type on a single platform type rather than the wider landscape of deception-based attacks. To address this issue, we discuss three high-level defense approaches for preemptive and proactive protection, including adopting the semantic attack killchain concept which simplifies targeted defense; principles for preemptive and proactive protection for passive threats; and platform based defense-in-depth lifecycle designed to harness technical and non-technical defense capabilities of platform providers and their user base. Here, the human-as-a-security-sensor paradigm can prove particularly useful by leveraging the collective natural ability of users themselves in detecting deception attempts against them.

1 Introduction

It is often posited that the user can be the "weakest link" [1] in information security, because even the strongest technical protection can be bypassed or undermined

R. Heartfield (✉) · G. Loukas
University of Greenwich, London, UK
e-mail: r.j.heartfield@gre.ac.uk; g.loukas@gre.ac.uk

M. Conti et al. (eds.), *Versatile Cybersecurity*, Advances in Information Security 72, https://doi.org/10.1007/978-3-319-97643-3_4

if an attacker successfully manipulates a user into divulging a password, opening a malicious file or visiting a compromised website. We begin by introducing the concept of "semantic social engineering attacks" formalised as *Semantic Attack* [31], which refers to a cyber threat targeting the user-computer interface as an attack vector, circumventing traditional technical security controls through user deception rather than by exploiting technical vulnerabilities. Common examples include phishing emails and websites, drive-by downloads, file and multimedia masquerading, domain typosquatting, malvertising and Trojan horse software to name a few.

Semantic attacks target human nature as a unique and distinct vulnerability in a computer system's security by triggering key emotional, behavioural and cognitive processes designed to elicit specific user response which allows an attacker to defeat a system's information security. Semantic attacks can be highly successful because without the requisite training and conditioning for threat detection (consider an operating system without the defense of antivirus scrutinising each and every system call it is being asked to make), human nature tends towards trust rather than mistrust. As a result, the threat is ubiquitous and the variation between attack vectors (such as their degree of complexity and target platform) is extreme, ranging from state-backed Advanced Persistent Threats employing multi-stage/platform attack vectors to that of script kiddies and pay-as-you-go bots generating automated phishing emails campaigns. Due to the vast problem space, attacks can be technically basic [34, 39, 40], highly complex [41, 60] or a combination of the two [36, 61].

Over the years, numerous defenses have been proposed at scientific research level to target exploitations such as website and phishing attacks [42–45, 62], as well as at commercial level [5, 6, 18–20]. However, they almost always fail to consider the wider problem space in which semantic attacks pose a threat, the result of which is the design of technical mitigations to address very specific attack vectors, lacking the flexibility to detect conceptually similar attacks across different platforms. Furthermore, over the years traditional deception-based attacks, such as phishing emails, spoofed websites and drive-by downloads, have shifted to new platforms in social media [35], cloud applications [36] and near field communications [37], and the advent of the Internet of Things (IoT) [38] will extend considerably the impact of semantic attacks through threats to physical space. The more effective semantic "cyber-physical" attacks prove [2], the larger the threat space becomes.

1.1 A Brief History of Semantic Attacks in Computer Systems

Semantic social engineering attacks first emerged in computer systems as early as 1989 when the *"AIDS Information Introductory Diskette"* Trojan [63] was sent to a mailing group in which Dr Joseph Popp, the Trojan's author, subscribed. To gain access to a computer system, a diskette pertaining to contain information about the AIDS virus deceived the recipients into inserting it into their system. The diskette contained a Cryptovirus [64] which ransomed users for money by

encrypting their systems files. Another noteworthy semantic attack appeared a year later, introducing what we call today "Scareware". The malware, aptly named *Nightmare* [7], was distributed via diskettes called "Fish Disks" designed to share applications between *Amiga* computer systems. On execution, every five minutes *Nightmare* would hijack the computer screen for 0.8 s to display a full-screen image of a skull with bullet wound and blood leaking out, whilst playing a loud shriek on the speakers. The malware posed no obvious risk to user data, but the concept of scaring/panicking a user would later be employed by many cyber criminals to force users into opening malware or paying for fraudulent services [46]. In 1995, new attacks 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, where cyber criminals would purchase domain names that were similar to those of legitimate websites. Users who mistyped the domain name URL of a legitimate website would be redirected to a malicious or fraudulent website. During the same year, service provider America Online (AOL) experienced growing success with a popular instant messaging tool, which hackers soon realised that it could be exploited, and developed an attack tool that lead to the first use of the term "phishing". *AOHell* [47] contained a "fisher" tool that enabled hackers to steal passwords and financial information by generating instant messages to random AOL users with content such as: "Hi, this is AOL Customer Service. We are running a security check and need to verify your account. Please enter your username and password to continue".

Over the next decade, phishing attacks became widespread. In 2000, the infamous *ILOVEYOU* "worm"[1] contained a malicious visual basic script titled "LOVE-LETTER-FOR-YOU.txt.vbs" [9], initially spreading through corporate Philippine mailing lists and eventually affecting over 45 million computers systems worldwide. This attack was copied a year later in 2001 by the *Anna Kournikova* worm, using the same worm generating script [48]. 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 [10]. Leading up to today, the rapid growth of the Internet, multimedia services and mobile platforms, have enabled semantic attacks to spread further into Android devices [49], peripheral hardware accelerated by direct memory access [11] (e.g., Thunderbolt and Firewire devices), file sharing networks [50], search engine optimisation engines [65] and drive-by malware on websites [51], 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 online social paradigm has enabled cyber criminals to take advantage of "friend" recommendations, user "posts" and sharing of media or apps that are replicated and automated with the network [12, 35, 52]. Also concerning is the potential for semantic attacks to result in physical impact, through cyber-physical and IoT systems.

[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 in [8].

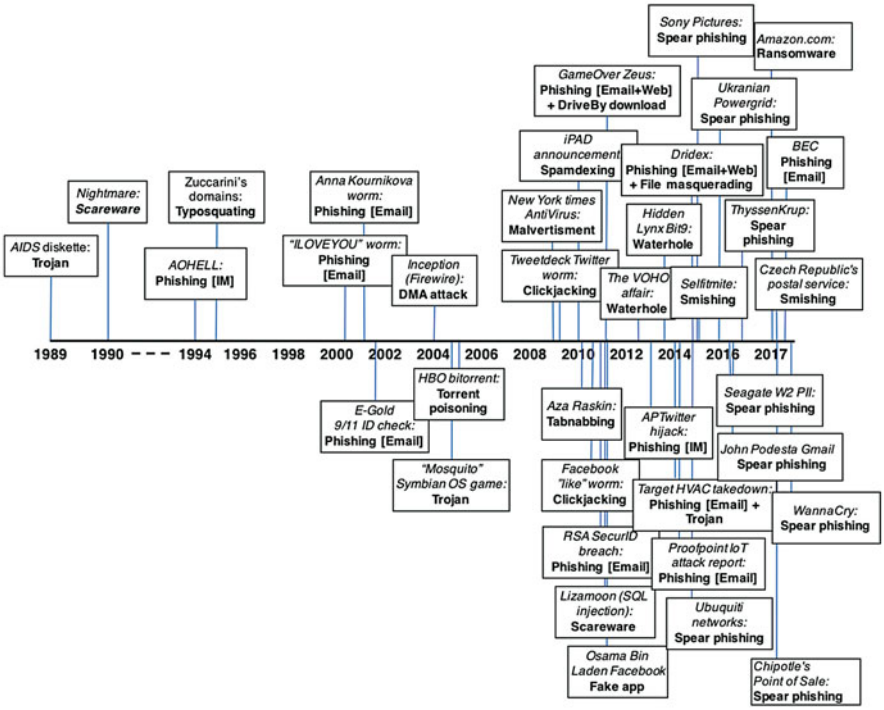

Fig. 1 Timeline of notable semantic attacks

In Fig. 1, we provide a timeline of high-profile semantic attacks identifying the chronological emergence and persistence of different types.

1.2 Characterising the Extreme Diversity in Semantic Attacks

There exist over 35 individually types or variations of semantic attack, existing within and cross-contaminating between different platforms and systems (Table 1).

2 The Scale of the Threat Today: Characterising the Impact of Semantic Attacks

Semantic attack statistics have been dominated by phishing incidents due to their widespread use by cyber criminals and consistent success in breaching computer systems. A 2012 report by Trend Micro identified that over 90% of targeted malware attacks discovered were initiated through spear-phishing [13]. In 2014, *Social*

Table 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 and or organisation
Pharming	Installing malicious code on a personal computer or server, misdirecting users to fake web sites without knowledge or consent
Whaling	Type of phishing attack that targets high-profile end users such as corporate executives, politicians and celebrities
QRishing	Phishing style attack using quick response (QR) codes to distribute malicious file/links
Blue snarfing	Phishing attack enticing users to install malware which grants access to target device via the Bluetooth protocol
Smishing	Phishing style attack sent via mobile short message service (SMS)
URL spoofing	Impersonating a websites URL address such as copying domain name by exploiting bugs in web browsers
DriveyBy download	Implanting a malicious file through programmatic manipulation of scripts on a vulnerable web platform
Waterhole	Targeted version of a DriveBy download attack, typically targeting platforms a victim accesses
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 app (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 masquerade as a secure website
Scareware	Malicious program tricking a user into buying/downloading unnecessary often malicious software (e.g., antivirus protection)
Rogueware	Standalone malware program pretending 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
Typosquatting	Registering similar domain names which rely on typographical errors when inputting a website address into a browser

(continued)

Table 1 (continued)

Attack Pseudonym	Description
Combosquating	Form of typosquatting registering domain names that combine popular trademarks with a string of words or phrases
RansomWare	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 persuading users to share on their profile, often used to spread fake apps and phishing URLs
Click jacking	Concealing hyperlinks beneath legitimate click-able content, 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 which applies to mobile devices where users touch the interface instead of using a mouse or keypad
Cursor jacking	Variation of clickjacking where users are deceived by means of a custom cursor image and 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 media plugin typically spread by through a fake video post on social media posting
Madware	Aggressive advertising placement in mobile devices photo albums, calendar entries and notification bar
Browser extension malware	Malicious browser-add similar to Trojan app that steals personal information and/or add browser to attacker botnet

Engineer reported that 90% of the 129 billion emails sent daily are malicious. Clicking on email links accounted for 80% of reported phishing attacks, and phishing itself represented 77% of all socially-based attacks [14]. In 2015, Statista reported that phishing and deception-based attacks accounted for 62% of all cyber attacks experienced by companies world-wide [15], with 59% reported by US companies alone [16]. Furthermore, the average time to resolve this type of attack for a US-based company was 20 days [17], with damages of 12% for medium and 16% for large enterprises' total operating costs. 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 [3] for years 2008 to 2016, illustrating in Fig. 2 that the number of phishing reports received by APWG is dramatically increasing.

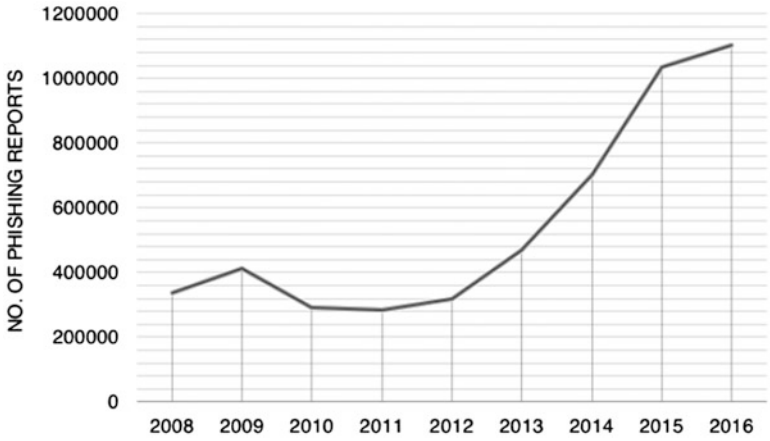

Fig. 2 APWG phishing report statistics for years 2008–2016 [3]

The Internet Security Threat Report, a yearly security study produced by Symantec, expands beyond traditional phishing statistics and organises semantic attacks amongst four categories: mobile and IoT, social media and spam, web threats and targeted attacks. Figures 2, 3 and 4 summarise a number of semantic attacks and threats utilising deception techniques from years 2012 to 2016 [4]. For mobile platforms, from 2012 to 2016 approximately 14.8 million apps were categorised as malware; with a further 22.4 million apps categoried as grayware. Malware and grayware require users to agree to install applications, granting aggressive permissions to the applications on the device, 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 period of 2013 to 2015, whilst the number of recipients per campaign have decreased by an average of 25% each year, which may indicate that attackers are developing methods for spear phishing which require fewer targets for successful exploitation and are more difficult to detect. Whilst spear phishing attacks continue to target the financial sector, attackers are now often targeting the energy and health-care sectors too (Fig. 4).

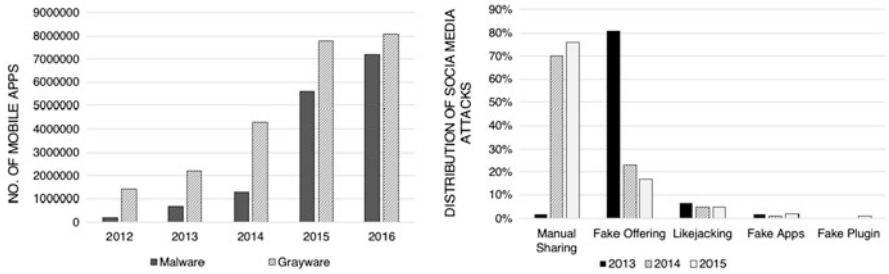

Fig. 3 Classification of mobile apps analysed by Symantec during 2012–2016 (left), Distribution method of social networks and social media scams/attacks by percent from 2013–2015 (right) [4]

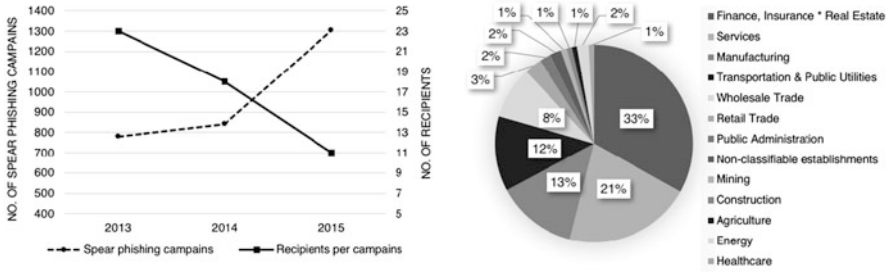

Fig. 4 Number of spear phishing campaigns and average number of attack recipients per campaign from 2013–2015 (left), top industries targeted by spear phishing attacks in 2015, ordered by majority percentage (right) [4]

3 Attacking the Weakest Link: Designing, Developing and Launching Semantic Attacks

Towards a core and collective understanding of semantic attack composition, beyond individual attacks on specific platforms, we start by exploring the specific characteristics which formulate the design, development and distribution of semantic attacks. To illustrate the functional components of a semantic attack, we employ taxonomy in [31], describing the generic schematic structures of semantic attacks, which apply irrespective of specific attack vectors that may be used (e.g., specific platform user interface). Next, we apply this approach on notable real world semantic attacks.

3.1 Generic Attack Structure

Semantic attacks, irrespective of attack vector, follow a generic functional structure in terms of design and delivery [31]. The high-level structure can be seen as

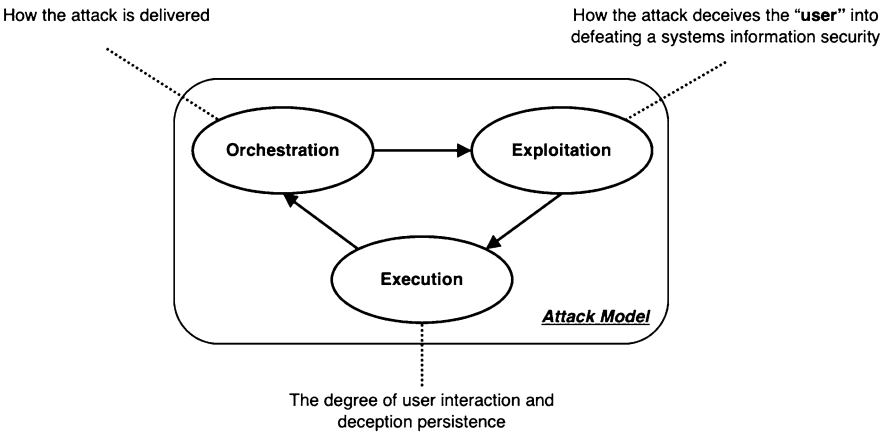

Fig. 5 High-level attack model showing the interactions between each behavioural function within semantic social engineering attacks

comparable to any kind of cyber threat in that it consists orchestration, exploitation and execution. Orchestration consists of user targeting (including information gathering), distribution and automation parameters, exploitation is the application of deception vectors (to elicit compromising user actions) and their technical construction through the user interface, and execution is the functional interaction required by the user during attack runtime and the persistence of the deception vector(s) after the attack is complete. Below, we summarise each of these individual elements as to their functional behaviour in semantic attack design and illustrate the process in Fig. 5.

- **Target description**. *The targeting parameters of the semantic attack*. Typically, this is a target user (an individual or organisation) or target platform. The former constitutes "explicit" targeting, which requires tailored attack delivery and may predetermine the method of distribution, automation and deception vectors to employ after a information gathering phase. By comparison, a specific user interface platform is a form of "promiscuous" targeting, as the attack vector does not control who is exposed, the functionality of the target platform and the behaviours of the users do (e.g., social media sharing).
- **Method of Distribution**. *The means by which a semantic attack reaches a target platform or user*. There are two means of distribution: software or hardware, the latter of which can also result in subsequent software executed distribution. Hardware is always a local distribution vector (e.g., within physical proximity of the user), while software is local (i.e., through a hardware interface) or remote (over a distributed application and network i.e., the Internet). For local hardware interaction, examples include direct memory access peripherals (e.g., Firewire), local hardware with software executed distribution is a system that is

locally interfaced with a target system (as initiated by the user), with physical communication and software execution (e.g., USB flash drives).

- **Method of Automation**. *The degree of attacker supervision of the semantic attack activation and administration*. An attack can be fully automated by predefining all its functional procedures in a format similar to worms, whereby the attack contains all of the procedural code necessary to operate without specific attacker execution, interaction or administration. The degree of automation depends on the functionality of the target platform and the behaviour of its user base. For highly targeted attacks that require tailored deception vectors which are meaningful to specific user(s) or organisational attributes, manual attacker operation may be required. This includes activities such as specifying when the attack is executed, or responding in real-time to user interaction (e.g., instant messaging). The degree of attack automation is dependent on the target description.
- **Deception Vector**. *The deception techniques designed to persuade the target user(s) into performing a compromising action*. The deception techniques developed for a semantic attack effectively represent the human exploitation parameters which persuade the user into performing a compromising action (e.g., clicking on a URL, or opening a executable file). At a high level, deception within a semantic attack has three modes. Firstly, the use of cosmetic, visually convincing deception by masquerading as a legitimate entity (through a specific user computer interface design), secondly behavioural deception by conforming to system convention in respect to expectations of user interface functionality and response to user interaction and thirdly a hybrid combination of cosmetic and behavioural deception.
- **Interface Manipulation**. *The technical implementation of an semantic attacks deception vector(s)*. Interface manipulation is the technical means used to establish a semantic attacks deception vector on a target platform's user interface. There are two ways in which this achieved, either through (ab)using legitimate platform functionality or programmatically modifying and or spoofing it in order to change appearance of behaviour.
- **Executions steps**. *The number of functional steps an attack requires the user to carry out in order to execute the exploitation payload*. The primary interaction with a semantic attack is the corresponding user action in response to exposure to its deception vector(s). Depending on the attacks required users actions, this can be a single step (e.g., a single user click) or multiple steps (e.g., multiple users clicks) in order for the attacks exploitation to complete; or as the means direct the user to another semantic attack in the attack chain.
- **Attack persistence**. *The persistent level of deception after user exploitation*. After successful exploitation, it is rare for a deception vector to continuing executing, as typically exploitation is a one-off procedure to forward the user to another semantic attack in a attack chain or as the user action has enabled execution of the intended attack payload. However, in some cases a semantic attack will continually execute deception vectors, as is common with Scareware.

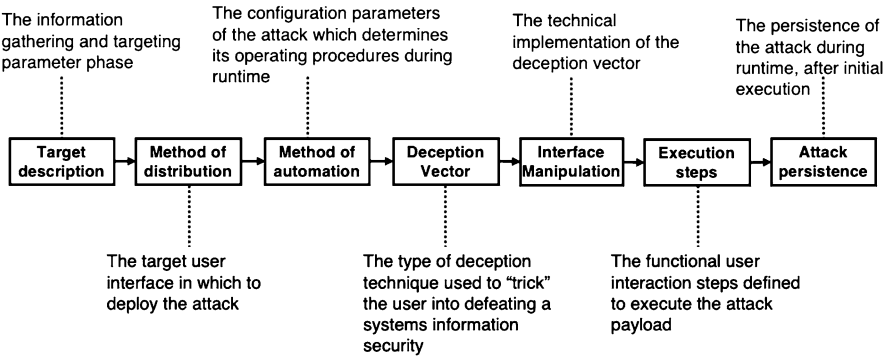

Fig. 6 High-level structure of a semantic social engineering attack illustrated as series of linear steps which together formulate the design, development and execution behaviour of a semantic attack [31]

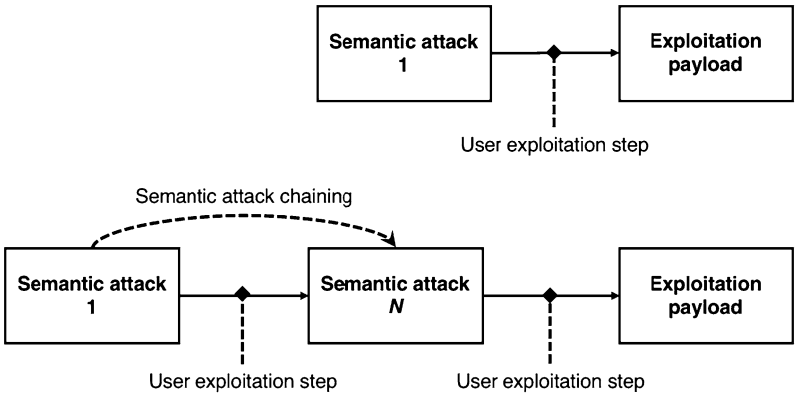

Fig. 7 Attack models for a single phase (top) and multi-phase (bottom) semantic attack. Here each individual attack (e.g. semantic 1 . . . n), is formulated by and therefore contains the linear structured criteria and corresponding parameters defined in Fig. 6

Semantic attacks follow the same functional structure, regardless of whether they are executed as individual semantic attacks or chained together within a multi-phase attack. Each individual semantic attack is distinguished by its functional elements (as per Fig. 6), even if in practice certain parameters, such as the target description and method of distribution, are shared as a consequence of attack chaining. To illustrate this, Fig. 7 provides an abstract example of a single semantic attack against a chain semantic attack model, and Fig. 8 provides examples of how multiple individual semantic attacks form a multi-phase semantic attack through chaining.

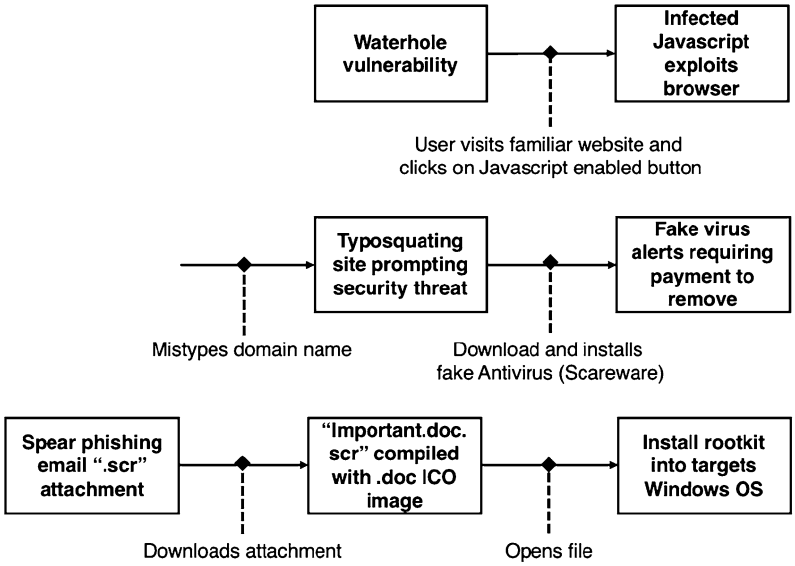

Fig. 8 Examples of single (top) and multi-phase semantic attacks (middle and bottom). Here the waterhole vulnerability is a one step attack requiring that a target simply visit the infected website for the attack to complete. For the typosquating attack, after a user visits the website, they are then prompted with a scareware download, which formulates a second phase semantic attack in the attack chain. For the spear phishing email, the user must download a masqueraded file, which consitutes the second phase semantic attack through user execution of the file to complete the attack

3.2 Semantic Attacks in the Press

In the following attacks, we employ these generic attack structure principles to provide a low-level breakdown of three notable semantic attacks: Spear phishing, QR code phishing and multimedia masquerading on social network platforms.

3.2.1 The Podesta Spear-Phishing E-mails

During the 2016 United States presidential election race, John Podesta, former chief of staff to Bill Clinton (and at the time chairman of the 2016 Hillary Clinton presidential campaign), received an email purportedly from Google with a warning that his Gmail account had received a sign-in attempt from an IP address in Ukraine. It advised Podesta "you should change your password immediately", including a blue "CHANGE PASSWORD" box to be clicked. This attack was part of a chained semantic attack process, whereas once this button was clicked, Podesta's Gmail account was redirected to a Google login phishing page, where his credentials were entered and ultimately stolen, giving the attackers access to over fifty thousand

Table 2 Podesta spear-phishing e-mail attack

Spear phishing e-mail (Fig. 9)	
Target	This specific e-mail had been crafted for John Podesta, using the salutation "Hi John" and using a provocative warning associated to a falsified connection attempt from a politically sensitive country
Distribution	The e-mail was distributed to the user's Gmail inbox through the SMTP protocol from the attacker's own mailservers
Automation	Unlike typical spear phishing attacks which carried out manually, the particular attack parameters seemed to have been programmed automatically as the same attack vector was exposed to multiple political candidates during the US presidential election race
Deception	The attack mimicked both visual and behavioural functionality, by (1) spoofing a legitimate looking Gmail address, (2) copying exactly the Gmail email template, (3) creating what appears to be a genuine Google URL from Bitly shortening service where the attack then leads to a second semantic attack which spoofed the Google webpage used to reset an account passwords. These deception vectors were implemented by (1) pragmatically modifying the SMTP (or registering the corresponding domain if available), (2) copying the source code from the Gmail email template, (3) abusing Bitly's inbuilt functionality to create a custom URL string
Execution	The attack required the target user to perform a single action by clicking on the "CHANGE PASSWORD NOW" button, at which point the exploitation of the email is complete and the user is redirected to a phishing page designed to harvest their account login credentials
Persistence	After the email attack as successful (by clicking on the link in the email body), this attack is completed and exhibits no further persistence as the user is redirected to a phishing website semantic attack as the next stage in the attack chain

emails with highly sensitive exchanges and data related to the Hilary Clinton presidential campaign. In Table 2 we take a closer look at the low-level configuration of this spear phishing email, illustrating the attacks visual deception in Fig. 9.

3.2.2 WhatsApp "Jack" (QRLjacking)

In 2016, ethical hacker Mohamed Abd Elbaset demonstrated how to hack the WhatsApp web connectivity service (which is associated to a WhatsApp account) by employing a variation of QR code phishing (Qrishing). Unlike previous QRishing attacks which opted to generate QR code with malicious URLs, this attack employs the concept of QR link jacking where the attacker creates a legitimate client side browser session to WhatsApp web service to generate a QR code and forwards this legitimate QR code through a phishing webpage to the victim. Here, QR link phishing would normally be logically ordered after a phishing email in a chained set of semantic attacks. If the attacker has access to the victim's network, a phishing email can be replaced by using ARP cache poisoning to forward the victim to the phishing website. On scanning the legitimate QR code, the victim's WhatsApp account on their mobile device registers with the WhatsApp

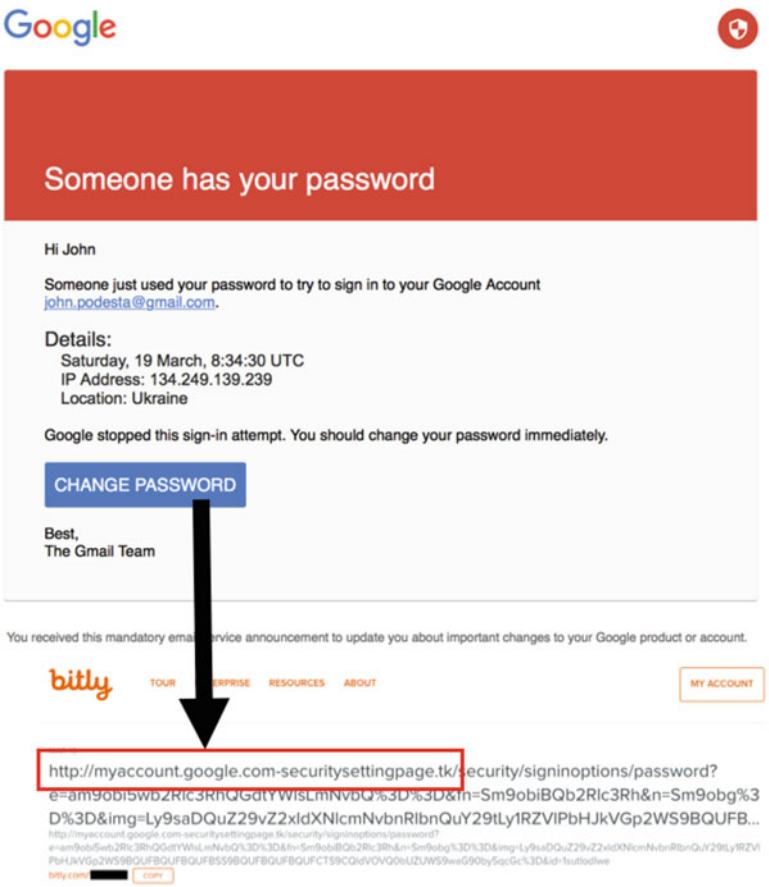

Fig. 9 Gmail spear phish attack against US former chairman of the 2016 Hillary Clinton presidential campaign. The top email body illustrates the spoofed gmail login alert with a change password button, with the bottom image showing the malicious link obfuscated by the button

service, which subsequently allows the attackers to register their connection to the WhatsApp web application with the victim's accounts. This results in the attacker having full access to any data transmitted from the victim's WhatsApp application on their device (Table 3).

3.2.3 The Case of the Facebook "Hungry Bear"

In 2009, there was an incident in a Berlin zoo where a lady jumped into a polar bear enclosure, which she subsequently survived. Soon, a Facebook multimedia masquerading scam emerged, using a doctored image, which appeared to be clickable video. The video was an image with a superimposed play video icon

Table 3 QRcode phishing link jacking attack

Spear phishing e-mail (Fig. 10)	
Target	Whilst designed for the WhatsApp platform, the recipient is explicitly defined as they must have a WhatsApp account to exploit and the attacker is required to distribute a phishing attack in order to expose the intended target to the QR code
Distribution	The QR WhatsApp authentication code is distributed through a mirrored website that maintains a persistent link to the attacker original client side connection
Automation	The attack is fully automated once established through a looping script mirroring the attackers client connection to WhatsApp web authentication page, requiring no further intervention from the attacker
Deception	The QR code and mirrored phishing website employ a combination of visual and behavioural deception. The QR code is a legitimate web authentication request, which when scanned responds correctly to the user authenticating their WhatsApp mobile account with the web service
Execution	The QR link jacking requires programmatic manipulation of the web authentication page generating the QR code for WhatsApp. The attacker must mirror the web page and create a script to continually update the QR code which is refreshed every 20 s on the WhatsApp web authentication
Persistence	Once the user has been duped into accessing the phishing website created by the attacker which hosts the mirrored QR code, they simply need to scan the QR code to generate an authorisation token which the attack requires to gain access to their account data. After scanning the code the attack execution is complete and the deception vector of the QR code ceases

Facebook video masquerading (Fig. 11)	
Target	The attack targets all Facebook users
Distribution	The fake video is distributed through social media profile timelines, provided as feeds to a profile's subscribers or friends through the Facebook *EdgeRank* algorithm; this increases the virality of the fake video post based on popularity such as post comments and links
Automation	The video masquerading post is automated once launched, whereby user sharing behaviour enables the video to be spread through inbuilt Facebook functionality. The process of redirection to a fake video website also requires no attacker intervention. It is a URL that activates once the image is clicked on
Deception	By superimposing the Facebook specific play video button on the image and augmenting the post with fake comments, the video masquerading attack utilises crude visual deception
Execution	The attack is constructed by simply creating a timeline post from a Facebook account and attaching the doctored image; using standard inbuilt Facebook functionality to embed the image as a hyperlink to an external website
Persistence	Once the fake video image is clicked by a user, the Facebook video deception vector is complete and the victim is forwarded to the secondary semantic attack

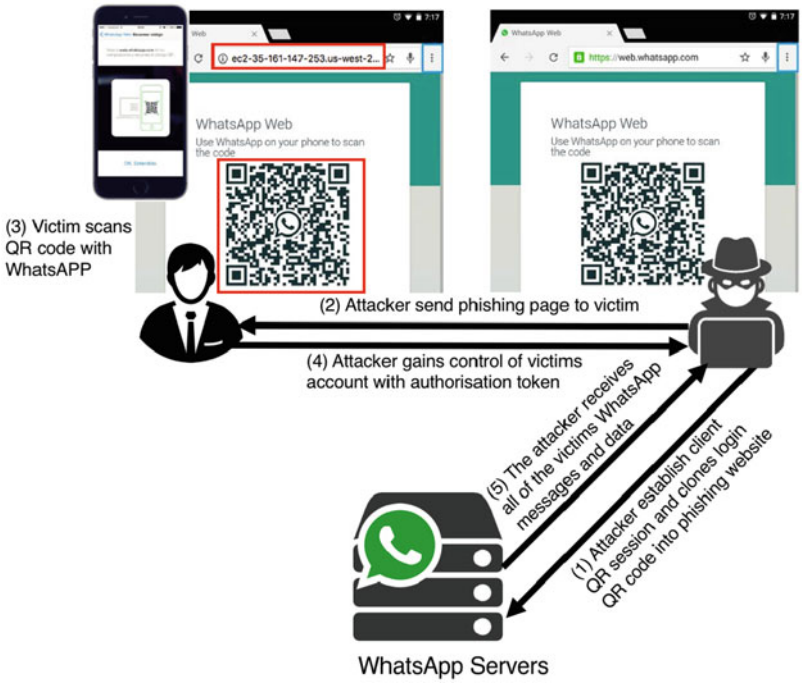

Fig. 10 WhatsApp QR link hijacking attack (initiated via sending a target victim a phishing website link). Here the spoofed QR code is automatically refreshed through the Javascript that has been copied from the legitimate WhatsApp authentication page

button, which redirected users to a secondary semantic attack in the form of a typical scam video webpage coercing the victim into completing a pop-up survey for access to the video and installing malware.

4 Methods for Defense Against Semantic Attacks

The extreme diversity of semantic attacks has led to many types of defenses proposed, often with multiple techniques developed for a single type (especially, phishing emails). Most defense mechanisms aiming to protect against the wider semantic attack space remain experimental products of research without integration or long-term empirical validation. So, the problem space is left with research and commercial tools which address only a small portion of the problem space. Here, we analyse the different defense approaches. We have already shown that individual types of semantic attack, irrespective of attack vector, are composed of the same functional elements. We have illustrated that individual semantic attacks can be used in attack chaining to direct users from one semantic attack to another to deliver

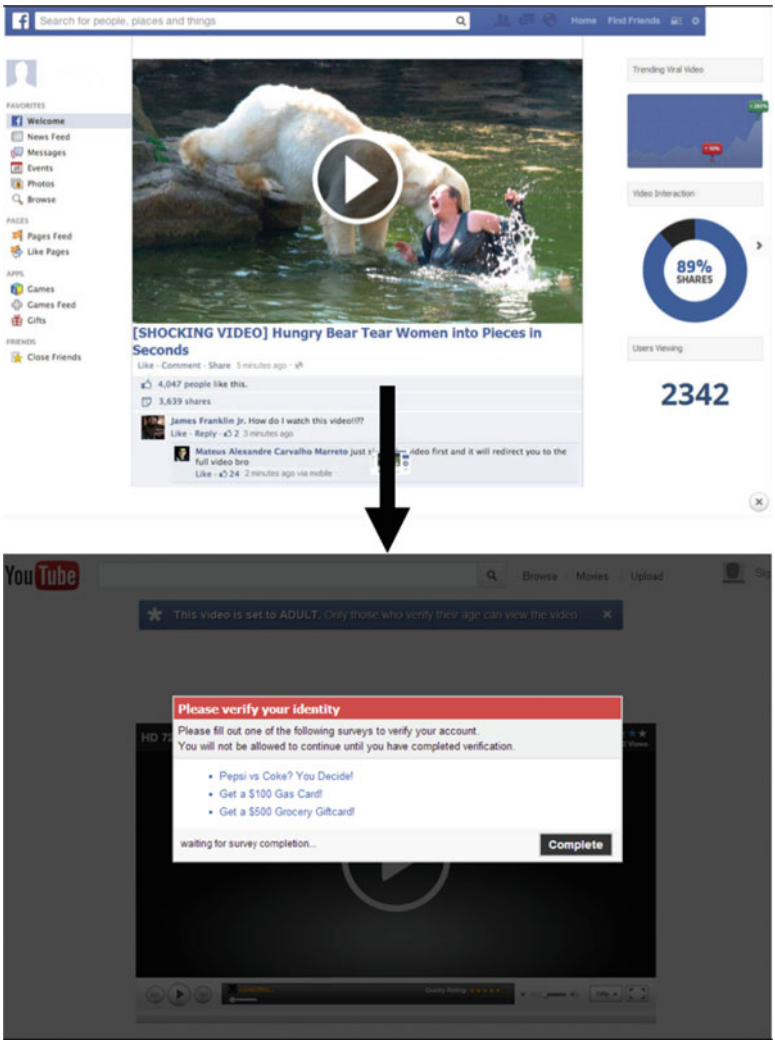

Fig. 11 The "Hungry Bear" Facebook video masquerading. The top image shows the fake Facebook video post of a bear allegedly attacking a woman, which once clicked directs the user to a new web page prompting the user to complete a survey that harvests sensitive user information to watch the spoofed video (bottom image)

the desired attack payload, and that in this composite attack architecture, some attack parameters such as targeting and automation can be shared across attacks. By focusing on both technical and non-technical mitigation concepts, rather than conducting an exhaustive search of the defense literature across all possible attack vectors (of which there are many, see Table 1), here we take a view of defense according to key concepts that would address the wider semantic attack problem

space. This is because relying solely on the low-level functionality of any single defense is insufficient as a means for defending against the wider threat space. Here, we consider defense across three complementary dimensions, which may offer a holistic defense architecture against semantic attacks: Semantic attack killchain for targeted defense simplification; principles for preemptive and proactive protection for passive threats; and platform based defense-in-depth lifecycle designed to harness technical and non-technical defense capabilities of platform providers and their user base.

4.1 Semantic Attack Killchain: Defense Simplification

A primary aim for lasting and practical defense against semantic attacks is to address a wide range of attack vectors without introducing considerable complexity. Intuitively, it is possible to limit the number and types of user interfaces in which to target defense mechanisms by focusing defense on initial (i.e., entry) attack vectors, that if mitigated, would serve to kill a wider semantic attack chain. In line with Lockheed Martin's killchain model [70], we refer to this as the semantic attack killchain. For example, consider a phishing email containing a URL to an attack website, where once clicked the user is forwarded to a drive by download resulting in infection with a Trojan horse spyware application. Focusing on the individual semantic attacks in the attack chain (e.g., the drive-by download), rather than its possible permutations (phishing email → attack website → drive-by download → spyware), simplifies the objective of defense. In the above example, the spyware would be thwarted by blocking the phishing email or the attack website or the drive-by download.

Figure 12 illustrates how a semantic attack killchain architecture is constructed. Firstly, the aim is to identify the different entry vectors by which a semantic attack may target and reach an organisation/individual (which may change based on the environment context and platforms used). The purpose is to help establish an indicative threat landscape by highlighting the means by which both single and multi-phase semantic attacks pose a risk to technical security. Determining the potential entry vectors of semantic attacks then simplifies the strategic placement of defence mechanisms to both address semantic attacks that rely on attack chains in order execute certain deception vectors, as well as minimising the number of different defence systems required to be implemented to address such threats specifically. In Fig. 13, an example of how a semantic attack killchain would function is illustrated.

Table 4 provides an indicative list of common user interface platforms required to distribute different semantic attacks. For the instant messaging and website distribution categories, we include functionality observed in modern social media and networking sites, chat forums and message boards, whereas for the Appstore category we include the functionality provided by online webstores, appstore and app marketplaces for mobile devices. The table shows that prevalent user interfaces

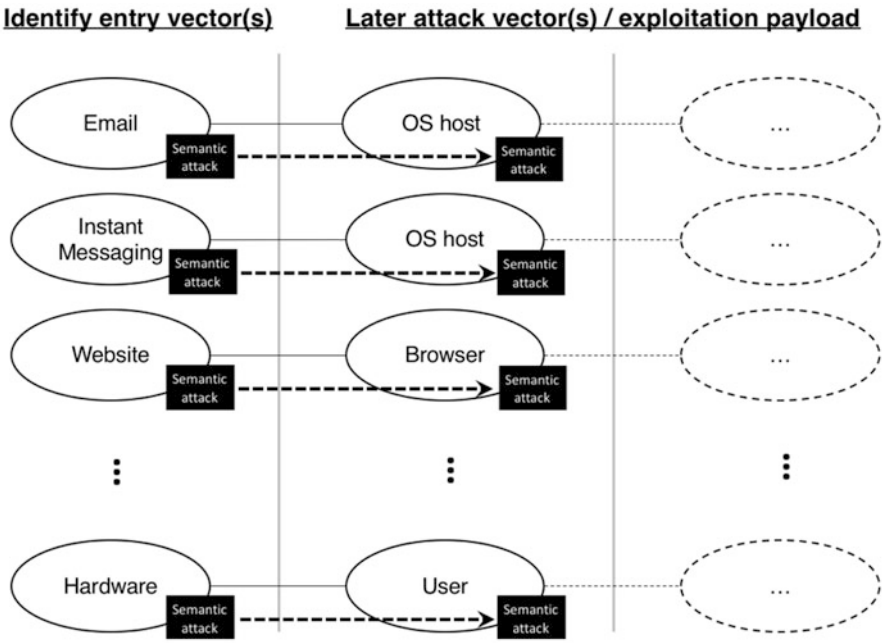

Fig. 12 The semantic attack killchain architecture aims to simplify the object of defence by providing a means to identify key platform entry vectors for semantic attacks. The aim is to design and implement defences which address a wide range of attacks whilst reducing the complexity of security mechanisms employed

are website platforms, followed by email and instant messaging platforms. However, for an attacker to initiate a targeted user attack (Target Description = "Explicit"), direct communication with a specific victim user is always required. Therefore, the sole use of a website or appstore interface as the primary distribution for user targeting becomes impractical for attackers as it limits the types of attack vectors available – especially for targeted attacks (thus introducing the need for an attack chain). For instance, if a website were the primary distribution means for a targeted exploit, the attacker may need to develop a complicated waterhole attack after finding a vulnerable platform that their target visits, ensuring that the deception vector for the target only activates for their specific browser's user agent string; this approach is of course complex and time consuming and therefore of less practicality to threat actors. In the same sense, for an appstore, or network (Net) or hardware (H/W) interface, simplicity is reduced by the need to direct the users through some means to these platform types. As a result, attackers often first rely on an initial unsolicited communication vector as distribution dependency in an attack chain, such as the use of email or instant messages containing a link to the target platform where a secondary semantic attack is positioned. As a basic high-level example of defense simplification against targeted semantic attacks specifically, for

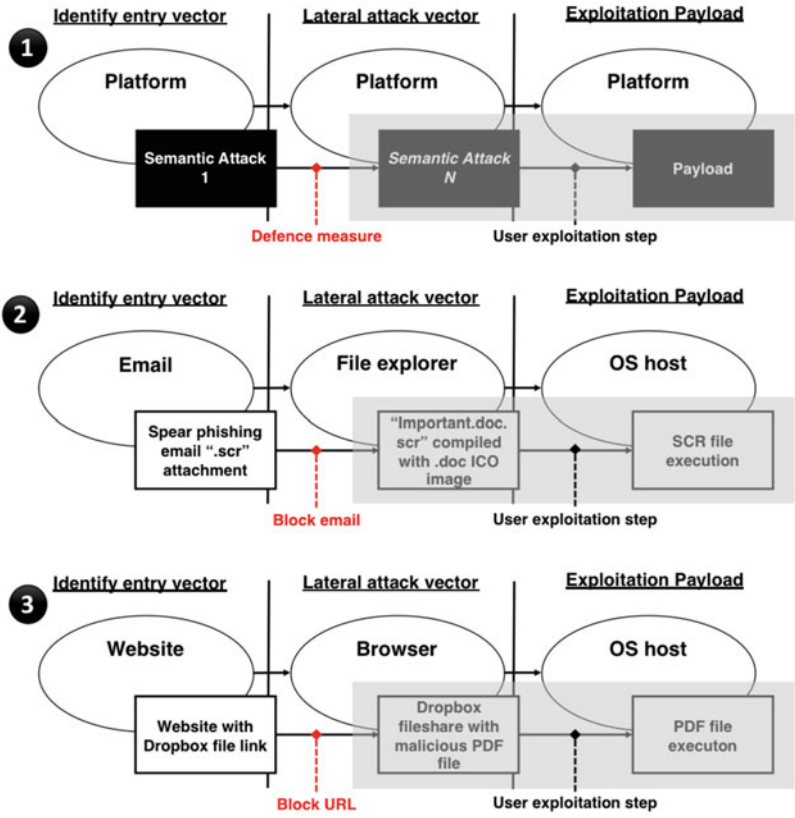

Fig. 13 Here an (1) abstract semantic attack chain from Table 7 is illustrated to show how a semantic attack consisting of multiple phases can be effectively nullified by addressing defence against the attack parameters of its first phase. By example, using the spear phishing threat example from Fig. 8 (2), we can see that by simply blocking the spear phishing attack as the first phase of the semantic attack chain, all subsequent phases and attack vectors can be averted. Equally, by blocking a malicious URL on a website, a malicious Dropbox share hosting PDF file malware can be also be prevented from deceiving the user into downloading and opening the file (3)

unsolicited targeted attacks (e.g., attacker directly contacts the target) the killchain can be reduced from 28 to 9 different attacks. The corresponding attacks and distribution platforms are shown in Table 5. The number can be further reduced if an attacker does not have control of a website for a waterhole and Bluetooth snarfing is mitigated by simply turning off Bluetooth. Further analysis of Table 5 shows that by developing defense mechanisms for email or instant messaging platforms would address 6 of the remaining different semantic attack vectors.

For completeness, in Table 6 we expand beyond the key killchain defences identified in Table 5 by further identifying applicable protection mechanisms that have been proposed for the wider landscape of different semantic attacks (Table 1). While we have no expectation to provide an exhaustive literature characterisation for such a long list of attacks, here, a sample of defense mechanisms and literature

Table 4 Methods of distribution required to **directly** exposure target users to corresponding semantic attacks

Attack vector	E-mail	Instant message	SMS	Appstore	Website	H/W interface	Net interface
Phishing	✓	✓	✓		✓	✓	
Spear phishing	✓	✓	✓			✓	
QRishing	✓	✓		✓	✓	✓	
Bluetooth snarfing							✓
Smishing			✓				
DriveyBy download					✓		
Waterhole					✓		
File masquerading	✓	✓			✓	✓	
Multimedia masquerading		✓		✓	✓		
GUI confusion					✓		
URL spoofing	✓	✓		✓	✓		
Visual SSL spoofing	✓				✓		
Scareware				✓	✓		
Malvertisement				✓	✓		
WiFi evil twin							✓
Trojan horse				✓	✓		
Self XSS	✓	✓			✓		
Typosquatting					✓		
Combosquatting					✓		
Tabnabbing					✓		
Sharebaiting					✓		
Click jacking					✓		
Cursor jacking					✓		
Spamdexing					✓		
Torrent poisoning					✓		
Fake app				✓	✓		
Fake plugin				✓	✓		
Malicious browser add-on				✓	✓		

papers on each of these attacks provides a useful tool for evaluating the current protection mechanisms against these threats. Here, we aim to identify existing approaches to defence that can be employed in unison with the semantic attack killchain to establish a practical and selective means of holistic defence against semantic attacks.

Table 5 Example of reduced semantic attack vectors by killchain platform defense

Attack vector	Primary distribution	Applicable defences
Phishing	Email IM SMS H/W	Machine learning [86, 89, 90, 99] User awareness [77, 78, 85, 88, 95, 98]
Spearphishing	Email IM SMS H/W	Machine learning [84, 86] User awareness [78, 92]
Smishing	SMS	Machine learning [71]
File masquerading	Email IM Website H/W	Integrity checking [83]
Multimedia masquerading	IM Website	Dynamic and static analysis [114]
Bluetooth snarfing	Net	Authentication/platform lock-down [100]
Waterhole	Website	Machine learning [72, 74] Dynamic analysis [73]
URL spoofing	Email IM Website	Machine learning [87, 97] User awareness [77, 78, 85, 88, 95, 98]
Visual SSL spoofing	Email Website	Heuristic scanning [94]

The examples in Table 5 largely agree with recent commercial defense products focusing on messaging platforms. Currently, most major email (Gmail [23], Microsoft Outlook [24], Yahoo [22]) and antivirus (Norton [25], Kaspersky [18], Sophos [20]) providers claim to have integrated robust detection capabilities for email threats.

In general, the utility of the semantic attack killchain can be expanded as a function of defense strategy across multiple and independent platforms, where platform providers aim to simplify their own semantic attack security by focusing protection mechanisms to address specific user interface functionality that would also serve to thwart other potential deception vectors that this may lead to through attack chaining. For instance, in the case of the social networking platform *Facebook*, focusing defense measures on *Facebook Messenger* as a distribution mechanism to plant malicious links to other semantic attacks within *Facebook* (e.g., *Facebook* pages with malicious content, fake Facebook videos, file masquerading etc.) or external phishing websites would serve as a killchain that simultaneously addressed multiple deception-based threats distributed on or via the *Facebook* platform.

Table 6 The application of defence

Attack pseudonym	Primary distribution	Defence category
Spam	Email IM SMS, social media	Machine learning [86, 89–91] User awareness [77, 78, 85, 88, 98] Sandbox [96]
Pharming	Email IM SMS	Machine learning [86, 89, 90] User awareness [77, 85, 88, 95, 98]
Whaling	Email IM SMS	Machine learning [86, 90] User awareness [77, 85, 88, 95, 98]
QRishing	Email Website SMS	Machine learning [106]
DriveyBy download	Website	Machine learning [72, 74] Dynamic analysis [73]
GUI confusion	Mobile app	Machine learning/user awareness [93] Static analysis [82]
Adware	Website, app marketplace	Sandbox [105]
SSL Spoofing	Website	Machine learning
Scareware	Software app	Machine learning [103, 104]
Rogueware	Software app	Sandbox [105]
Malvertisement	Social media, website	Machine learning [115, 116]
WiFi evil twin	Net	Integrity checking [80] User awareness [79]
Trojan horse	Software/App	Sandbox [75, 76]
Rogue AP	Net	Integrity checking (RTT analysis) [107]
Self XSS	Browser	Integrity checking [117]
Typosquatting	Browser	Machine Learning [118] Integrity checking (rule-based) [119]
Combosquating	Browser	Integrity checking (rule-based) [119]
RansomWare	Software app	Sandbox [109] Machine learning [110] Formal methods [111]
Tabnabbing	Website	Machine learning [108]
Sharebaiting	Social media Website	User awareness [81]
Click jacking	Social media Website	Integrity checking [101]
Like jacking	Social media Website	Integrity checking [123, 124]
Touch jacking	Social media (mobile) Website (mobile)	Integrity checking [101, 123, 124]

(continued)

Table 6 (continued)

Attack pseudonym	Primary distribution	Defence category
Cursor jacking	Social media Website	Integrity checking [123]
Spamdexing	Search engine	Machine learning [120, 121] Heuristic scanning [122]
Torrent poisoning	Torrent software	Integrity checking (reputation scoring) [125]
DNS cache poisoning	DNS server	Authentication/integrity checking [102]
Fake app	Mobile app marketplace Side-loaded install	Machine learning/user awareness [93]
Fake plugin	Browser	Dynamic and static analysis [114]
Madware	Mobile app marketplace Side-loaded install	Machine learning/user awareness [93] Static analysis [82]
Browser extension malware	Browser	Static analysis [112] Integrity checking [113]

4.2 Principles for Preemptive and Proactive Protection Against Semantic Attacks

The use of a semantic attack killchain helps to simplify the placement and scope of defense against semantic attacks by reducing to what kind of platforms and where in those platforms to place defenses; with the aim to reduce the diversity of where an attacker can initiate the exposure of a semantic attack on a given platform. However, the killchain method alone cannot cater for the unpredictability of user access to different computer platforms through passive activity. That is, where a user inadvertently exposes themselves to a semantic attack through their computing habits and behaviour. For example, the effectiveness of a semantic killchain blocking certain semantic attacks by placing defenses within an email platform is effectively bypassed if a user chooses them self to access a malicious website or application directly, without being coerced to by an attacker. It is necessary therefore to design defenses to protect against user activity and behaviour which may expose users to passive semantic attack threats. Namely, preemptive (prevention of semantic attack execution) and proactive (detection and treatment of semantic attack exposure) system security. However, it remains a continued challenge to develop best practice preemptive and proactive defense techniques when their exists such extreme diversity between semantic attacks, even when they employ conceptually similar deception vectors across multiple disparate platforms. To address this complexity, it is valuable to revisit the generic semantic attack structure in Sect. 3.1 and analyse each modular component of a semantic attack to develop insights for establishing generic principles of preemptive and proactive defense that are

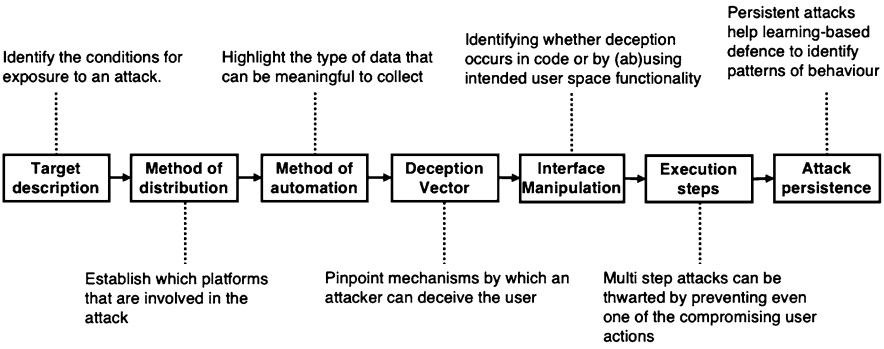

Fig. 14 Defense considerations with the generic semantic attack model. Here, the semantic attack model is employed as a template framework to develop defence measures against key aspects of an attack's construction and behaviour

independent of individual attack vectors. Below, we provide instructional examples for eliciting key parameters that aid the construction of preemptive and proactive defense mechanisms based on each generic semantic attack component, as well providing corresponding examples of defense systems that have been developed in research and commercial platforms. Figure 14 summarises the key defense considerations for each individual component of a semantic attacks structure.

- **Target description**. Distinguish between the different of targeting parameters across a range of semantic attack to identify both distinct and common conditions for exposure. For example, whether a user is at risk due to their identity (so that these attributes can be monitored and evaluated), or whether a users passive computer (whether personal or work related) usage inadvertently exposes them to certain semantic attacks.
  Recent advances in the detection of spear phishing email have demonstrated that by monitoring explicit user attributes and interactions in email content, corresponding meta-data can be learned proactively to generate anomalous behaviour facilitating the detection of spear phishing. For example, in 2015 researchers Stringhini and Thonnard developed a prototype spear phishing classification engine which collected and profiled behavioural features associated to email writing style, composition, communication context (e.g., time/date, email chain, contact interaction) within a support vector machine learning system to detect compromised email accounts. Similarly, commercial security vendor *Barracuda* [21] have introduced a spear phishing detection system called *Sentinel* that monitors an organisations communication history based on specific user email interactions as a context feature-set to train an artificial intelligence system to predict and prevent future attacks.
- **Method of Distribution**. Identify platforms that are involved in an attack to provide defense developers with the insight to choose which remote (e.g., involving a network) or local system to monitor to determine where best place

the defense mechanism. Establish patterns in distribution between systems, such as shared functionality (e.g., cross-site advertising plugins) to highlight where functionality supply chains open up a semantic attack distribution vector. Again, guiding developers as to points of vulnerability within a platforms user interface. Stringhini et al. have shown that by monitoring the redirection path taken to reach a web destination effectively identifies attack platforms involved in a malicious website, which are intentionally obfuscated from blacklists [56]. Instead of positioning defense locally on a web browser platform which scrutinises physical website features, the researchers demonstrated that patterns in HTTP redirection can be used as a distributed means of detection through network-based analysis; hence informing developers that such detection may be implemented within network security systems monitoring DNS and HTTP requests.

- **Method of Automation**. The type of automation exhibited dictates to a large degree the response mechanism or the type of data that can be collected for its detection. For instance, an attack that is fully automated is likely to leave a fingerprint of behaviour that can be used to develop attack signatures, whereas an attack that is conducted manually will benefit from focusing on specific attacker behaviour.

  In 2014, an example of measuring automated attack procedures by Ruskov et al. demonstrated how by dynamically monitoring the sequence of actions within a semantic attack can help to model user and attacker behaviour through simulation. This process can then be used to facilitate the development of knowledge-based defense systems that can more efficiently detect deception-based threats through enumeration of automatic attack procedures.
- **Deception Vector**. Establish the different deception vectors possible on a platforms user interface so that developers and researchers can pinpoint the mechanisms by which an attacker can manipulate the visual and/or system behaviour to "trick" the user into committing a compromising action.

  As susceptibility to deception vectors triggers user exploitation, it is generally agreed that semantic attack education is a core element of defense-in-depth against semantic attacks where technical mechanisms fail to prevent or proactively detect threats. As a result, research has explored interactive training through bitesize quizzes, test and games and attack simulations to maximise the effectiveness of learning [57, 58, 62], some of which have empirically proven to reduce susceptibility to deception vectors and have been converted into popular commercial offerings, with examples including PhishGuru [26], Anti-Phishing Phil and Phyllis [27] and PhishMe's Simulator [28] applications. However, most commercial solutions for security awareness training focus almost exclusively on phishing emails and websites, which constitute only a small portion of different semantic attacks possible deception vectors. Where research has explored further deception vectors in other attacks, these remain largely as prototype products. Moreover, the type of awareness training can vary just as much as the diversity of different semantic attacks if training is based on specific attack vectors rather than general concepts of good cyber hygiene; the prior of which can become outdated quickly. Therefore, where possible it is important that embedded awareness training and user interface security indicators are integrated both individually

and across interdependent platforms based on generic semantic attack principles rather than specific attack vectors.

- **Interface Manipulation**. Identify whether the deception vector occurs in code or by abusing intended user space functionality, to shape the design of a defense system and to narrow down its scope.
  To prevent or detect deception vectors without relying on the effectiveness of user awareness training requires platform developers to pinpoint vulnerabilities in the user interface where it may be (ab)used to execute deception vectors, whether through programmatic manipulation or intended user space functionality. In both cases, preemptive functions have been explored in research [53] for the android operating system to block malicious apps executing visual and behavioural deception through spoofing a legitimate applications appearance. The preemptive defense was implemented by capturing and analysing application program interface calls to the android graphic user interface to classify malicious behaviour. In the commercial space, the use of sandbox environments has gained popularity to interrogate the legitimacy of user interface functionality, for example as to whether certain functions result in potentially malicious behaviour. For instance, most modern web browsers employ sandbox technologies to isolate prevent website JavaScript coding from manipulating browsers' visual and behavioural properties, examples include the presentation of URLS in the address bar and the format of visual user security indicators such as the level of websites transport layer security. In the majority of cases commercial security technologies focus on preemptive programmatic manipulation rather than the misuse of normal user functionality.
- **Executions steps**. Execution steps: An attack that relies on more than one step can potentially be 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.
  Recent advances and greater uptake in the FIDO authentication protocol [29] has demonstrated robust proactive defense against phishing attacks, by enforcing two-factor authentication integrated between multiple architectures. Successful deception will not always result in user account compromise as the FIDO protocol employs temporal session keys generated by a second factor of authentication always available to the user (typically biometric).
- **Attack persistence**. 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 defense system (or platform user) to gradually identify its pattern of behaviour and report or block it.
  Whilst persistent deception for a singular semantic attack is uncommon, for Scareware attack vectors in particular persistent deception forms part of the exploitation payload. In 2011, Shahzad and Lavesson [54] proposed a machine learning approach based on mining variable length instruction sequences as a means for detection of persistent attack behaviors. In 2013, Microsoft demonstrated high detection accuracy in a prototype system that identified

Table 7 Summary of preemptive and proactive defenses based on generic semantic attack components

Attack component	Mechanism	Preemptive	Proactive	Practicality	Maturity
Target description	Integrity checking	✓	✓	P + O	Medium
Method of distribution	Platform monitoring	✓	✓	P + O	Low
Method of automation	Threat modelling		✓	O	Medium
Deception vector	User awareness training	✓	✓	O	Low
Interface manipulator	Platform sandbox		✓	P+O	High
Execution steps	Cross platform AAA		✓	P+O	High
Attack persistence	Machine learning classification	✓	✓	P + O	Low

persistent patterns in visual scareware deception through image detection with Logistic Regression using stochastic gradient descent [55].

Table 7 summarises the types of preemptive and proactive defenses, according to the generic functional elements of a semantic attack, their practicality for both the personal (P) user and organisational (O) operating environment, including their general maturity as defense solutions at the time of writing.

4.3 Platform-Based Defense-in-Depth Lifecycle

A defense-in-depth lifecycle for user platforms is intended to provide a multi-faceted framework approach for effectively implementing semantic attack defense. Its primary aim is to establish key roles and responsibilities for different components of a system (e.g., platform provider, security, developer and users) that contribute holistically to defense against semantic attacks. The platform defense lifecycle does not just represent a specific software or hardware platform providers' (e.g., social media or app vendor, email or website host), but also includes any organisational context providing access to user-interface platform(s) for their incumbent user base. In both scenarios, the lifecycle applies as a framework to provide through life defense for preemptive and proactive defense measures against semantic attack threats by establishing the ecosystem of responsibility which can be utilised to harness different defense capabilities through each functional element of the lifecycle.

In Fig. 15, we illustrate three key roles: platform developer, platform security and platform user, which form each element of a platforms defense lifecycle against semantic attacks. Whilst each role is distinct in its own right (e.g., contribution to defense and dependencies for its utilisation), this is not intended to indicate an

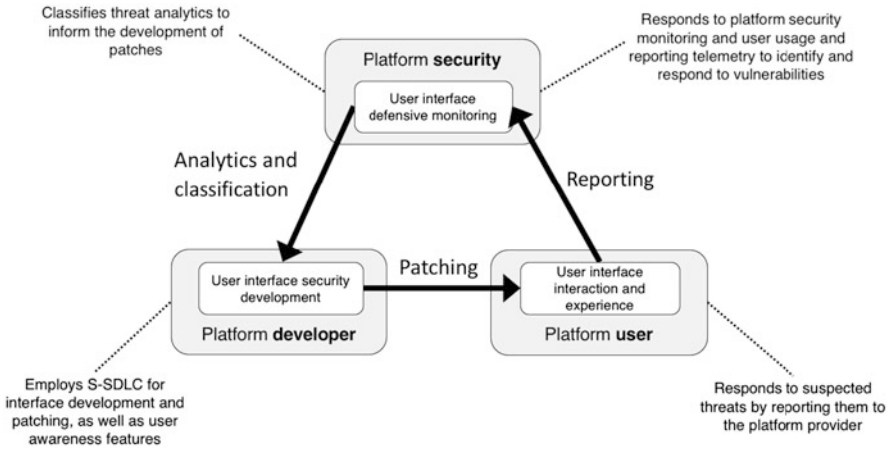

Fig. 15 The semantic attack defence life cycle consists of three continuously interacting defence functions: secure platform development, user threat reporting and platform security systems; which in combination aim to provide holistic and complementary preemptive and proactive protection against semantic attacks

implicit separation of the particular role amongst different entities, as each role in the lifecycle can exist in the same organisation. As an example, *Facebook* is a independent social network platform which implements internal functions for each role described, but for the *Facebook* platform and users only. However, within an organisational environment, each role may serve as a holistic function across all platforms that the organisation hosts or makes available to their user base. So, the lifecycle is intended as a high-level functional model that is applicable across multiple contexts to describe how to combine the capabilities of each type of platform role for defense against semantic attacks. Below, we elaborate on each role as to their defense function:

- **Platform developer**. Developers are responsible for programming both the internal and user interface functionality of a platform that is both secure and resilient to technical threats or abuse of intended user space functionality that would result in deception vectors for semantic attacks. Platform developers can employ the Secure Software Development Life cycle (S-SDLC) framework to design and integrate security considerations systematically into the core requirements and design of the platforms architecture, as well as utilise threat classification from system and user security telemetry. The following S-SDLC life cycle stages provide indications of activities to be carried out when introducing new user interface functionality or applying security patches against deception based threats.

  – *Requirements*: Define intended user space functionality and its expected limitations to establish any possible attack surface through misuse. This requires documenting system-to-system and system-to-user interactions which form the

platforms system of interest, then identifying how and if these interactions affect other related platforms within the deployment environment.

– *Design*: Develop user interface threat models which consider different elements of the platforms design. Highlight potential weak spots in the user interface that may be targets for misuse or vulnerabilities in data representation and transfer (within and externally from the platform) which can be used to inject toward or extract data from target users.

– *Coding*: Build programmatic determinism through static code analysis to establish confidence that the platforms programmatic features do not force the platform to exhibit visual or behavioural deception vectors if targeted by spoofed or injected data on the user interface (e.g., where a graphic or physical button may be (ab)used through normal user usage).

– *Testing*: Utilise "fuzzing" based test within user interface scenarios where different types of behaviour are arbitrarily executed as means to elicit anomalous system responses or situations which can form the basis for a deception vector.

– *Release/Maintain*: Implements processes and procedures from the platforms security monitoring capability as part of continual integration to support the delivery of telemetry based patching (both system and user awareness based) against internal as well as external platform vulnerabilities.

- **Platform security**. The security role within the platform defense lifecycle is responsible for both implementing preemptive security measures (e.g., to prevent technical vulnerability exploitation or platform misuse), as well supporting proactive defenses through monitoring and collecting system and user telemetry to aid in the detection of unknown or emergent threats. The latter of which feeds into preemptive defense through activation of security rules (e.g., blocking an activity), as well as providing crucial platform analytics and classification data that is forwarded to platform developers to produce platform patches and or to develop future secure user interface functionality. For semantic attacks, unlike traditional platform security controls such as intrusion detection systems or firewalls, the platform security requires to monitor and respond to key measurable elements of the user interface via telemetry produced by the platform itself and by the user base who access it.
- **Platform user**. Sole reliance on platform security alongside external technical defense mechanisms provided by platform users is often insufficient as a means of defense for detecting the vast range of semantic attacks, especially when deception vector utilises legitimate user space functionality [33]. It is imperative that outside of technical controls employed, that user telemetry is also harnessed. Here, the contribution of platform users for semantic attack defense is twofold. Firstly, platform users generate activity that can be analysed to determine if their behaviour is consistent with malintent or victimisation. Platform-based user activity creates meta-data that has been shown to be useful in a number contexts where human activity can be used for establishing situational awareness in natural disasters [59]. From a security standpoint, the same method can be used to identify where user activity is consistent with exploitation by a semantic attack or indeed the construction of one. The second utility of platform users

is their ability to report security threats, where a growing trend in platform security to address semantic attacks is to provide a means for users to report suspected threats. Most major email and browser platforms now provide inbuilt mechanisms for users to report phishing attacks or malicious content, with various external security companies developing enterprise platforms, such as *PhishMe Reporter* and *Wombat Security PhishAlarm*. In general, with access to both types of user telemetry (passive platform usage and active threat reporting), platform security and developers are provided with rich analytics that can be used to classify and intercept suspected threats in online platform defense systems or through patching of vulnerabilities in the user interface.

4.4 Defense in Hindsight

Taking in to consideration the three dimensions of defense we have discussed, here we evaluate how these defense approaches would have provided defense against the three real world semantic attacks illustrated in Sect. 3.2.

- **Podesta Emails**.
  - *Semantic Attack killchain*: By detecting and blocking the Google phishing email, the Google login phishing website which was used to capture user credentials would have been thwarted. Therefore, to prevent this exploitation it would have required only that the email was blocked or detected as malicious to prevent compromise.
  - *Principles of preemptive and proactive defense*: If the email account in question had enabled mails two form factor authentication mechanisms then on redirection to the attackers phishing login page and input of login credentials the attackers would still have been unable to successfully access the email account.
  - *Defense-in-depth lifecycle*: This particular email template was spoofed directly from Google and was not the first time it had been used or reported as phishing. Therefore, if first phishing report had been forward by Google's platform security to their Gmail platform security developers the combination of the emails content, images and domain name (i.e., not being an official Google email address) would have served as key features to create a detection signature. Which could have subsequently been built into Gmails phishing detection engine. Here, the combination of user reporting, Gmail's security analyst (or system) forwarding the report details and classification to developers, and security system updates by developers would have prevented the email ever having reached John Podesta's email account.
- **WhatsApp QRishing**.
  - *Semantic Attack killchain*: As this specific attack it is dependent on either a phishing email, instant message or SMS in order to coerce the user in to

accessing the supposed WhatsApp URL detecting the attack through these initial methods of distributions would prevent the user from being persuaded into visiting the phishing webpage.
  - *Principles of premptive and proactive defense*: A combination of standard phishing defenses analysing the integrity of webpage content (e.g., use of logos, scripts and redirection from site mirroring) alongside common phishing website awareness training (e.g., domain highlighting) would provide both a preemptive and proactive defense measure against this threat.
  - *Defense-in-depth lifecycle*: Through a user reporting mechanism this particular threat to WhatsApp web application would have alerted the platform security team of the QR code authentication vulnerability. As a result, prompting developers to patch the system to enforce a secure means for authentication through a secondary authentication mechanism only accessible to the legitimate user, as well as instead exploring ways to protect to their web platform by prevent third party mirroring of their website scripts generating the QR code.

- **The case of the "Hungry Bear"**.
  - *Semantic Attack killchain*: Multimedia masquerading serves as the first attack in a chain, spreading through Facebook via user activity ("liking" and manually sharing the video). However, the subsequent phishing website and fake survey attack chain could have been thwarted by focusing defense on the redirection behaviour of the post to external platforms outside Facebook.
  - *Principles of preemptive and proactive defense*: Much like the *Facebook EdgeRank* algorithm analyses features associated to social interaction to suggested friends, posts, advertisements and material it believes certain users are interested in, by monitoring post behaviours alongside textual information and responses from affected users, this attack could be proactively identified by developing machine learning models that match patterns of anomalous redirection activity. For example, collecting data associated to the post video nature (such as title text), user comments asking how to view the video on *Facebook*, alongside the URL redirection provides features that can be fed into a learning algorithm to classify the post's malicious and deceptive behaviour.
  - *Defense-in-depth lifecycle*: The combination of platform security measures monitoring both platform functionality activity and that of its user base would have served as crucial telemetry to *Facebook* security teams for classifying this as a suspected malicious post automatically, in turn, highlighting to platform developers the need to embed security measures in certain post configuration (e.g., an image post with an embedded URL) before users are redirected to external platforms automatically. Alongside technical detection, providing user notification requesting for confirmation of external website would highlight the nature of the post to users and indicate anomalous activity.

5 Open Research Challenges

5.1 Emerging Threats in the Internet of Everything

Historically, semantic attack exploits in computer systems were limited to traditional Internet communications such as messaging and web application platforms. However, in IoT, the threat landscape includes vehicles, industrial control systems, and even smart home appliances. The result of this is that the impact is not limited to cyberspace (such as stealing information, compromising a system, crashing a web service etc.), but is branching into physical space too. Early examples can be observed through physical damage dealt by malware in manufacturing plants, rail signalling, water treatment plants and even nuclear facilities. We anticipate that cyber-physical systems, such as industrial control systems and vehicles, will soon become realistic targets for user deception. The potential impact in physical space makes them attractive targets, and the limited diversity of human-system interaction, makes it difficult for the users to detect misbehaviour. For instance, consider the Tire Pressure Monitoring System (TPMS) on a modern automobile. It consists of a sensor inside the tire, which monitors tire pressure and periodically transmits that data wirelessly to an electronic control unit (ECU) on the in-vehicle network of the automobile. If the tire pressure is below a threshold, this is displayed on the driver's dashboard as a tire pressure warning. Rouf et al. [68] have shown from as early as 2010 that spoofing these messages can be relatively straightforward. So, an attacker can wirelessly transmit fake TPMS data to the ECU and trigger a fraudulent warning. The driver-system interface does not provide anything more than a visual display of the warning. So, there is no way for the driver to tell that this is a deception attempt rather than a legitimate safety issue, and as such will probably decide to pull over as soon as possible.

While the potential of attacks on vehicles captures the public's interest, it is cyber-physical systems in industrial control that have been targeted several times in high-profile incidents in the past. The exploitation was almost always highly technical, but usually the initial point of entry was standard spear-phishing and in some cases watering hole attacks. In 2014, a German steel mill was attacked via spear-phishing, with the aim to capture user credentials, gain access to the back office and from there to the control network, ultimately damaging a blast furnace. In 2015, 80,000 homes in Ukraine lost power when phishing emails deceived employees of the electricity provider into clicking on an attachment in an email, purportedly from the Prime Minister of Ukraine [69].

Smart home IoT systems also constitute attractive targets as deception devices. Most commonly, they involve access to cloud, voice-activated artificial intelligence (such as Alexa or Siri) and workflow automation services, such as IFTTT and Stringify. Each one of these can be compromised by deceiving a user or with the purpose to deceive a user. A simple example would be to inject audio commands (e.g. "Alexa, purchase item X") in an audio or video file sent to a user via email. In fact, a similar incident (albeit not an attack by design) occurred in 2016, when

a 6-year old girl asked Alexa Can you play dollhouse with me and get me a dollhouse? and Alexa actually did order a $160 dollhouse. Then, when a news presenter repeated this on TV while covering the story, several Amazon Echos in people's homes attempted to order further dollhouses. At the same time, any breach of confidentiality relating to the smart home can lead to breach of physical privacy and thus be useful information in the hands of attackers building a picture about a household's pattern of life. Data sniffed from sensors or smart meters can tell when someone is in or out during the week, and even how they look or sound like if their Internet-connected camera is compromised. The impact can be further accentuated if the breach of privacy extends to users' smart wearables, especially when they relate to health, as attackers can use information on likely medical conditions to target them convincingly. Consider the impact that ransomware would have if it were designed for wearable or implantable medical devices. Protecting against semantic attacks in cyber-physical context has not been explored yet.

5.2 Human-as-a-Security-Sensor

By their very nature, semantic attacks are challenging for autonomous technical defenses to detect and prevent. In recent years, the focus on user awareness training has shifted to actively involving users as human sensors spotting and reporting suspected attacks. As alluded to in Sect. 4.3, the user plays a crucial role in the platform defense-in-depth lifecycle against semantic attacks, supplementing technical defense mechanisms with human detection efficacy. In [32], we established the concept of Human-as-a-Security-Sensor by demonstrating how one's ability to detect different semantic attacks can be predicted. We then showed experimentally how predictive modelling can be integrated into a real-world technical system to actively engage and empower users to report detection of different semantic attacks across a range of platforms in real-time; outperforming all technical defense systems compared against [33]. Moreover, there are now examples of security vendors and public organisations advocating and actively employing human security sensor functions to augment defense against semantic attacks, such as the "human sensor" publicised phishing detection platform *PhishMe Reporter* and the University of Oxford CERT team's phishing reporting portal [30]. However, to realise this concept's long term benefits, there is a need to extend to a wider range of threats (other than phishing) and to find effective means of encouraging users to take part. Furthermore, in IoT space, the means by which to report suspected threats may be less intuitive or "safe" as that of a laptop or mobile device user interface. If a deception vector is executed within the user interface of an industrial control system or vehicle, it may be dangerous to stop any activity to report a suspected threat and similarly dangerous to ignore a suspected deception as false. Taking the example of spoofed tire pressure warnings again, if onboard vehicle security fails to detect the deception, the driver has to make the decision whether to report the suspected deception and ignore the tire pressure warnings or pull over as per the attacker's aim. The Human-as-a-Security-Sensor paradigm harbours great potential

for augmenting existing technical defenses in the fight against semantic attacks, but finding a consolidated and secure means of reporting threats against multi-platform user interfaces is a complex challenge.

5.3 Cyber Hygiene 2.0

Perhaps the most common advice for stepping up the security of an individual or organisation is to protect themselves against basic cyber threats by keeping their software and operating system up to date, avoiding unsafe websites or email attachments from people they do not know, and other basic measures collectively referred to as "cyber hygiene". The aim is prevention, and it is without doubt that such measures do improve overall security posture. However, as threats gradually become more advanced, and this is certainly the case for semantic attacks, individual users need to be equipped with recommendations on not only how to prevent, but also how to identify and respond to less basic attacks that have not been thwarted through prevention. Here, researchers can make use of the early steps that have been taken in the field of neuroscience, for instance using fMRI to show that the areas that exhibit the most brain activity when a user successfully distinguishes between a phishing and a legitimate website are those associated with decision-making, attention, and problem-solving [67], or using mindfulness to improve attention when facing a phishing attempt [66]. We anticipate that the next phase for cyber hygiene efforts will be in the form of simple techniques and habits that can help detect threats rather than relying on successful prevention.

6 Conclusion

Semantic attacks have been posing a significant and sustained threat to computer information security for almost 30 years, with a cryptovirus appearing as early as 1989 and scareware existing since 1990. The basic principles in deceiving users have remained largely the same, yet the threat has not been thwarted. On the contrary, all statistics point to a continuous increase in the number and diversity of semantic attacks and worsening impact. We argue that the ineffectiveness of the very large number of technical security approaches developed is that they look at each type of attack in isolation. For example, the deception logic and the nature of the tell-tale signs of phishing in email and in social media are the same. Yet, the two cannot be addressed by the same technical security mechanism. In response to this challenge, we have discussed three high-level defense approaches, which can be attractive areas for further exploration in addressing the wider semantic attack space. Wholly unsurprisingly for a type of threat that is based on user deception, the key in defense is again the human, whether as developer of user interfaces or as a user acting as human sensor, but not making the same mistake of attempting to detect threats in

isolation. The human users are best placed to thwart deception attempts against them if this capability can be leveraged as part of technical defense systems.

References

1. Schneier, B., 2011. Secrets and lies: digital security in a networked world. John Wiley and Sons.
2. Loukas, G., 2015. Cyber-physical attacks: A growing invisible threat. Butterworth-Heinemann.
3. APWG, 2018. APWG Phishing Attack Trends Reports. https://apwg.org/resources/apwg-reports/.
4. Symantec, 2018. Security Center Archived Publications - Internet Security Threat Reports https://www.symantec.com/security-center/archived-publications.
5. FirstCyberSecurity, 2009. Protecting your brand online and creating customer confidence. http://www.firstcybersecurity.com/main/IPRiskMReview.pdf.
6. Webroot, 2013. Webroot real-time anti-phishing service. http://www.webroot.com/shared/pdf/WAP-Anti-Phishing-102013.pdf.
7. Amiga Fish-Disk Database, 1990. Fish-disk 448 content: Nightmare. http://amiga-fish.erkan.se/amiga-fish-disk-448-contentNightMare/.
8. Cisco, 2017. Viruses, worms, trojans, and bots. https://www.cisco.com/c/en/us/about/security-center/virus-differences.html.
9. M. Bishop, 2000. Analysis of the iloveyou worm. Internet:http://nob.cs.ucdavis.edu/classes/ecs155-2005-04/handouts/iloveyou.pdf.
10. Financial Cryptography, 2005. GP4.3 - growth and fraud - case 3 - phishing, 2005. http://financialcryptography.com/mt/archives/000609.html.
11. M. Dornseif, 2004. 0wned by an ipod, 2004. Presentation. https://www.slideshare.net/KarlFrank99/owned-by-an-ipod
12. G. Cluley, 2011. Osama bin laden death video scam spreads virally on facebook. https://nakedsecurity.sophos.com/2011/05/02/osama-binladen-death-video-scam-spreads-virally-on-facebook/.
13. TrendLabs, 2012. Spear-phishing email: Most favored apt attack bait. Technical report, TrendLabs - APT Research Team. http://www.trendmicro.com/cloud-content/us/pdfs/securityintelligence/white-papers/wp-spear-phishing-email-most-favoredapt-attack-bait.pdf.
14. Social Engineer (2014). The social engineering infographic. http://www.social-engineer.org/social-engineering/socialengineering-infographic/
15. Statista, 2015. Types of cyber attacks experienced by companies worldwide as of August 2015. http://www.statista.com/statistics/474937/cyber-crime-attacks-experienced-by-global-companies/.
16. Statista, 2015. Average number of days to resolve a cyber attack on companies in the united states as of august 2015. http://www.statista.com/statistics/193463/average-days-toresolve-a-cyber-attack-in-us-companies-by-attack/.
17. Statista, 2015. Share of cyber crime damages caused to u.s. companies through phishing and social engineering in 2015. http://www.statista.com/statistics/193465/financial-damagecaused-by-phishing-for-us-companies/.
18. Kaspersky, 2017. Kaspersky internet security 2017. https:www.kaspersky.co.uk/internet-security
19. Avast, 2017. Safezone browser. https://www.avast.com/f-safezone.
20. Sophos, 2017. Intercept X tech specs. https://www.sophos.com/en-us/products/intercept-x/tech-specs.aspx.
21. Barracuda, 2017. Evolution of Spear Phishing. https://assets.barracuda.com/assets/docs/dms/Barracuda_Sentinel_WP_Evolution_Spear_Phishing_US.pdf
22. Yahoo, 2017. Secure your inbox. https://uk.antispam.yahoo.com/.

23. Engadget, 2017. Google beefs up gmail security to fight phishing attempts. https://www.engadget.com/2017/05/31/google-gmail-security-fight-phishing/.
24. Microsoft, 2017. Office 365 email anti-spam protection. https://support.office.com/en-us/article/https://support.office.com/en-us/article/Office-365-email-anti-spam-protection-6a601501-a6a8-4559-b2e7-56b59c96a586
25. Symantec, 2017. Norton security review 2017: Top antivirus provider with fully furnished internet security suites. https://fatsecurity.com/review/norton.
26. Wombat Security, 2017. PhishGuru Simulated Phishing Attacks. https://www.wombatsecurity.com/security-education/phishguru-simulated-phishing-attacks
27. Wombat Security, 2017. Security Awareness Training Modules https://www.wombatsecurity.com/security-education/security-awareness-training-modules
28. PhishMe, 2017. PhishMe Simulator. https://phishme.com/product-services/simulator-2/
29. FIDO alliance, 2017. How FIDO Works. https://fidoalliance.org/how-fido-works/
30. University of Oxford, 2016. Information security - report an incident. https://www.infosec.ox.ac.uk/report-incident.
31. Heartfield, R. and Loukas, G., 2016. A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks. ACM Computing Surveys (CSUR), 48(3), pp. 37.
32. Heartfield, R., Loukas, G. and Gan, D., 2016. You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks. IEEE Access, 4, pp. 6910–6928.
33. Heartfield, R., Loukas, G. and Gan, D., 2017, June. An eye for deception: A case study in utilizing the human-as-a-security-sensor paradigm to detect zero-day semantic social engineering attacks. In Software Engineering Research, Management and Applications (SERA), 2017 IEEE 15th International Conference on (pp. 371–378). IEEE.
34. Jordan, M. and Gouday, H., 2005. The signs, and semiotics of the successful semantic attack. In 14th Annual EICAR Conference (pp. 344–364).
35. Huber, M., Mulazzani, M., Weippl, E., Kitzler, G. and Goluch, S., 2011. Friend-in-the-middle attacks: Exploiting social networking sites for spam. IEEE Internet Computing, 15(3), pp. 28–34.
36. Heartfield, R. and Loukas, G., 2013. On the feasibility of automated semantic attacks in the cloud. In Computer and Information Sciences III (pp. 343–351). Springer, London.
37. Madlmayr, G., Langer, J., Kantner, C. and Scharinger, J., 2008, March. NFC devices: Security and privacy. In Availability, Reliability and Security, 2008. ARES 08. Third International Conference on (pp. 642–647). IEEE.
38. Weber, R.H., 2010. Internet of Things–New security and privacy challenges. Computer law and security review, 26(1), pp. 23–30.
39. Dhamija, R., Tygar, J.D. and Hearst, M., 2006, April. Why phishing works. In Proceedings of the SIGCHI conference on Human Factors in computing systems (pp. 581–590). ACM.
40. Drake, C.E., Oliver, J.J. and Koontz, E.J., 2004, August. Anatomy of a Phishing Email. In CEAS.
41. Huber, M., Mulazzani, M. and Weippl, E., 2010, September. Who on earth is Mr. Cypher: automated friend injection attacks on social networking sites. In IFIP International Information Security Conference (pp. 80–89). Springer, Berlin, Heidelberg.
42. Aburrous, M., Hossain, M.A., Thabatah, F. and Dahal, K., 2008, April. Intelligent phishing website detection system using fuzzy techniques. In Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on (pp. 1–6). IEEE.
43. Chou, N., Ledesma, R., Teraguchi, Y. and Mitchell, J.C., 2004, February. Client-Side Defense Against Web-Based Identity Theft. In NDSS.
44. Huang, H., Zhong, S. and Tan, J., 2009, August. Browser-side countermeasures for deceptive phishing attack. In Information Assurance and Security, 2009. IAS'09. Fifth International Conference on (pp. 352–355). IEEE.

45. Kumaraguru, P., Rhee, Y., Acquisti, A., Cranor, L.F., Hong, J. and Nunge, E., 2007, April. 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 (pp. 905–914). ACM.
46. Giles, J., 2010. Scareware: the inside story. New Scientist, 205(2753), pp. 38–41.
47. Rekouche, K., 2011. Early phishing. arXiv preprint arXiv:1106.4692.
48. Kabay, M.E., 2001. Viruses and worms: more than a technical problem. Ubiquity 2001. ACM
49. Leavitt, N., 2005. Mobile phones: the next frontier for hackers?. Computer, 38(4), pp. 20–23.
50. Kong, J., Cai, W. and Wang, L., 2010, February. The evaluation of index poisoning in bittorrent. In Communication Software and Networks, 2010. ICCSN'10. Second International Conference on (pp. 382–386). IEEE.
51. S. Doherty, J. Gegeny, B. Spasojevic, and J. Baltazar, 2013. Hidden lynx - Professional hackers for hire. Symantec Security Response. https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/hidden-lynx-hackers-13-en.pdf
52. Irani, D., Balduzzi, M., Balzarotti, D., Kirda, E. and Pu, C., 2011, July. Reverse social engineering attacks in online social networks. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 55–74). Springer, Berlin, Heidelberg.
53. Bianchi, A., Corbetta, J., Invernizzi, L., Fratantonio, Y., Kruegel, C. and Vigna, G., 2015, May. What the app is that? deception and countermeasures in the android user interface. In Security and Privacy (SP), 2015 IEEE Symposium on (pp. 931–948). IEEE.
54. Shahzad, R.K. and Lavesson, N., 2011, August. Detecting scareware by mining variable length instruction sequences. In Information Security South Africa (ISSA), 2011 (pp. 1–8). IEEE.
55. Seifert, C., Stokes, J.W., Colcernian, C., Platt, J.C. and Lu, L., 2013, May. Robust scareware image detection. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on (pp. 2920–2924). IEEE.
56. Stringhini, G., Kruegel, C. and Vigna, G., 2013, November. Shady paths: Leveraging surfing crowds to detect malicious web pages. In Proceedings of the 2013 ACM SIGSAC conference on Computer and communications security (pp. 133–144). ACM.
57. Asanka, N., Love, S. and Scott, M., 2012. Designing a mobile game to teach conceptual knowledge of avoiding'phishing attacks'. International Journal for e-Learning Security, 2(1), pp. 127–132.
58. Sheng, S., Magnien, B., Kumaraguru, P., Acquisti, A., Cranor, L.F., Hong, J. and Nunge, E., 2007, July. 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 (pp. 88–99). ACM.
59. Aulov, O. and Halem, M., 2012. Human sensor networks for improved modeling of natural disasters. Proceedings of the IEEE, 100(10), pp. 2812–2823.
60. Marforio, C., Francillon, A. and Capkun, S., 2011. Application collusion attack on the permission-based security model and its implications for modern smartphone systems. Technical Report. ETH Zurich.
61. Selvaraj, K. and Gutierrez, N.F., 2010. The rise of PDF malware. Symantec Security Response. https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the_rise_of_pdf_malware.pdf.
62. Kumaraguru, P., 2009. Phishguru: a system for educating users about semantic attacks. Carnegie Mellon University.
63. Bates, J., 1990. Trojan horse: AIDS information introductory diskette version 2.0. Virus Bulletin, pp. 3–6.
64. Young, A. and Yung, M., 1996, May. Cryptovirology: Extortion-based security threats and countermeasures. In Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on (pp. 129–140). IEEE.
65. Howard, F. and Komili, O., 2010. Poisoned search results: How hackers have automated search engine poisoning attacks to distribute malware. Sophos Technical Papers, pp. 1–15.

66. Jensen, M.L., Dinger, M., Wright, R.T. and Thatcher, J.B., 2017. Training to mitigate phishing attacks using mindfulness techniques. Journal of Management Information Systems, 34(2), pp. 597–626.
67. Neupane, A., Saxena, N., Maximo, J.O. and Kana, R., 2016. Neural Markers of Cybersecurity: An fMRI Study of Phishing and Malware Warnings. IEEE Transactions on Information Forensics and Security, 11(9), pp. 1970–1983.
68. Ishtiaq Roufa, R.M., Mustafaa, H., Travis Taylora, S.O., Xua, W., Gruteserb, M., Trappeb, W. and Seskarb, I., 2010, February. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In 19th USENIX Security Symposium, Washington DC (pp. 11–13).
69. Koppel, T., 2015. Lights out: a cyberattack, a nation unprepared, surviving the aftermath. Broadway Books.
70. Hutchins, E.M., Cloppert, M.J. and Amin, R.M., 2011. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare and Security Research, 1(1), pp. 80.
71. Joo, J.W., Moon, S.Y., Singh, S. and Park, J.H., 2017. S-Detector: an enhanced security model for detecting Smishing attack for mobile computing. Telecommunication Systems, 66(1), pp. 29–38.
72. Cova, M., Kruegel, C. and Vigna, G., 2010, April. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In Proceedings of the 19th international conference on World wide web (pp. 281–290). ACM.
73. Jayasinghe, G.K., Culpepper, J.S. and Bertok, P., 2014. Efficient and effective realtime prediction of drive-by download attacks. Journal of Network and Computer Applications, 38, pp. 135–149.
74. Lu, L., Yegneswaran, V., Porras, P. and Lee, W., 2010, October. Blade: an attack-agnostic approach for preventing drive-by malware infections. In Proceedings of the 17th ACM conference on Computer and communications security (pp. 440–450). ACM.
75. Blsing, T., Batyuk, L., Schmidt, A.D., Camtepe, S.A. and Albayrak, S., 2010, October. An android application sandbox system for suspicious software detection. In Malicious and unwanted software (MALWARE), 2010 5th international conference on (pp. 55–62). IEEE.
76. Brickell, E.F., Hall, C.D., Cihula, J.F. and Uhlig, R., Intel Corp, 2011. Method of improving computer security through sandboxing. U.S. Patent 7,908,653.
77. Cone, B.D., Irvine, C.E., Thompson, M.F. and Nguyen, T.D., 2007. A video game for cyber security training and awareness. Computers and Security, 26(1), pp. 63–72.
78. Heartfield, R. and Loukas, G., 2018. Detecting semantic social engineering attacks with the weakest link: Implementation and empirical evaluation of a human-as-a-security-sensor framework. Computers and Security, 76, pp. 101–127.
79. Heartfield, R., Loukas, G. and Gan, D., 2016. You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks. IEEE Access, 4, pp. 6910–6928.
80. Darknet, 2015. EvilAP Defender Detect Evil Twin Attacks. (2015). http://www.darknet.org.uk/2015/04/evilap-defender-detect-evil-twin-attacks/.
81. Heartfield, R. and Loukas, G., 2016, June. Evaluating the reliability of users as human sensors of social media security threats. In Cyber Situational Awareness, Data Analytics And Assessment (CyberSA), 2016 International Conference On (pp. 1–7). IEEE.
82. Bianchi, A., Corbetta, J., Invernizzi, L., Fratantonio, Y., Kruegel, C. and Vigna, G., 2015, May. What the app is that? deception and countermeasures in the android user interface. In Security and Privacy (SP), 2015 IEEE Symposium on (pp. 931–948). IEEE.
83. Dhanalakshmi, R. and Chellappan, C., 2010, July. Detection and recognition of file masquerading for e-mail and data security. In International Conference on Network Security and Applications (pp. 253–262). Springer, Berlin, Heidelberg.
84. Stringhini, G. and Thonnard, O., 2015, July. That ain't you: Blocking spearphishing through behavioral modelling. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 78–97). Springer, Cham.

85. Aggarwal, A., Rajadesingan, A. and Kumaraguru, P., 2012, October. PhishAri: Automatic realtime phishing detection on twitter. In eCrime Researchers Summit (eCrime), 2012 (pp. 1–12). IEEE.
86. Basnet, R., Mukkamala, S. and Sung, A.H., 2008. Detection of phishing attacks: A machine learning approach. In Soft Computing Applications in Industry (pp. 373–383). Springer, Berlin, Heidelberg.
87. Bhardwaj, T., Sharma, T.K. and Pandit, M.R., 2014. 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 (pp. 355–363). Springer, New Delhi.
88. Asanka, N., Love, S. and Scott, M., 2012. Designing a mobile game to teach conceptual knowledge of avoiding'phishing attacks'. International Journal for e-Learning Security, 2(1), pp. 127–132.
89. Bergholz, A., Chang, J.H., Paass, G., Reichartz, F. and Strobel, S., 2008, August. Improved Phishing Detection using Model-Based Features. In CEAS.
90. Dong-Her, S., Hsiu-Sen, C., Chun-Yuan, C. and Lin, B., 2004. Internet security: malicious e-mails detection and protection. Industrial Management and Data Systems, 104(7), pp. 613–623.
91. Drucker, H., Wu, D. and Vapnik, V.N., 1999. Support vector machines for spam categorization. IEEE Transactions on Neural networks, 10(5), pp. 1048–1054.
92. Stembert, N., Padmos, A., Bargh, M.S., Choenni, S. and Jansen, F., 2015, September. A study of preventing email (spear) phishing by enabling human intelligence. In Intelligence and Security Informatics Conference (EISIC), 2015 European (pp. 113–120). IEEE.
93. Malisa, L., Kostiainen, K. and Capkun, S., 2017, March. Detecting mobile application spoofing attacks by leveraging user visual similarity perception. In Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy (pp. 289–300). ACM.
94. Corbetta, J., Invernizzi, L., Kruegel, C. and Vigna, G., 2014, September. Eyes of a human, eyes of a program: Leveraging different views of the web for analysis and detection. In International Workshop on Recent Advances in Intrusion Detection (pp. 130–149). Springer, Cham.
95. Kumaraguru, P., 2009. Phishguru: a system for educating users about semantic attacks. Carnegie Mellon University.
96. Lee, K., Caverlee, J. and Webb, S., 2010, April. The social honeypot project: protecting online communities from spammers. In Proceedings of the 19th international conference on World wide web (pp. 1139–1140). ACM.
97. Lee, S. and Kim, J., 2012, February. WarningBird: Detecting Suspicious URLs in Twitter Stream. In NDSS (Vol. 12, pp. 1–13).
98. Sheng, S., Magnien, B., Kumaraguru, P., Acquisti, A., Cranor, L.F., Hong, J. and Nunge, E., 2007, July. 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 (pp. 88–99). ACM.
99. Xiang, G., Hong, J., Rose, C.P. and Cranor, L., 2011. Cantina+: A feature-rich machine learning framework for detecting phishing web sites. ACM Transactions on Information and System Security (TISSEC), 14(2), p.21.
100. Pandeym T. and Khare, P, 2017. Bluetooth Hacking and its Prevention. http://www.lnttechservices.com/sites/default/files/resources/pdf/whitepapers/2017-12/Bluetooth-Hacking-and-its-Prevention.pdf
101. Shamsi, J.A., Hameed, S., Rahman, W., Zuberi, F., Altaf, K. and Amjad, A., 2014, January. Clicksafe: Providing security against clickjacking attacks. In High-Assurance Systems Engineering (HASE), 2014 IEEE 15th International Symposium on (pp. 206–210). IEEE.
102. Larson, M., Massey, D., Rose, S., Arends, R. and Austein, R., 2005. DNS security introduction and requirements. IETF. https://tools.ietf.org/html/rfc4033
103. Shahzad, R.K. and Lavesson, N., 2011, August. Detecting scareware by mining variable length instruction sequences. In Information Security South Africa (ISSA), 2011 (pp. 1–8). IEEE.

104. Seifert, C., Stokes, J.W., Colcernian, C., Platt, J.C. and Lu, L., 2013, May. Robust scareware image detection. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on (pp. 2920–2924). IEEE.
105. BufferZone Pro, 2014. BufferZone-Pro sandbox. http://www.trustware.com/BufferZone-Pro/
106. Alnajjar, A.Y., Manickam, S., Anbar, M., Al-saleem, S. and Elejla, O., 2016. TrustQR: A New Technique for the Detection of Phishing Attacks on QR Code. Advanced Science Letters, 22(10), pp.2905–2909.
107. Beyah, R., Kangude, S., Yu, G., Strickland, B. and Copeland, J., 2004, December. Rogue access point detection using temporal traffic characteristics. In Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE (Vol. 4, pp. 2271–2275). IEEE.
108. Al-Khamis, A.K. and Khalafallah, A.A., 2015, November. Secure Internet on Google Chrome: Client side anti-tabnabbing extension. In Anti-Cybercrime (ICACC), 2015 First International Conference on (pp. 1–4). IEEE.
109. Kharraz, A., Arshad, S., Mulliner, C., Robertson, W.K. and Kirda, E., 2016, August. UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. In USENIX Security Symposium (pp. 757–772).
110. Vinayakumar, R., Soman, K.P., Velan, K.S. and Ganorkar, S., 2017, September. Evaluating shallow and deep networks for ransomware detection and classification. In Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on (pp. 259–265). IEEE.
111. Mercaldo, F., Nardone, V., Santone, A. and Visaggio, C.A., 2016, June. Ransomware steals your phone. formal methods rescue it. In International Conference on Formal Techniques for Distributed Objects, Components, and Systems (pp. 212–221). Springer, Cham.
112. Bandhakavi, S., King, S.T., Madhusudan, P. and Winslett, M., 2010, August. VEX: Vetting Browser Extensions for Security Vulnerabilities. In USENIX Security Symposium (Vol. 10, pp. 339–354).
113. Ter Louw, M., Lim, J.S. and Venkatakrishnan, V.N., 2008. Enhancing web browser security against malware extensions. Journal in Computer Virology, 4(3), pp. 179–195.
114. Ford, S., Cova, M., Kruegel, C. and Vigna, G., 2009, December. Analyzing and detecting malicious flash advertisements. In Computer Security Applications Conference, 2009. ACSAC'09. Annual (pp. 363–372). IEEE.
115. Li, Z., Zhang, K., Xie, Y., Yu, F. and Wang, X., 2012, October. Knowing your enemy: understanding and detecting malicious web advertising. In Proceedings of the 2012 ACM conference on Computer and communications security (pp. 674–686). ACM.
116. Poornachandran, P., Balagopal, N., Pal, S., Ashok, A., Sankar, P. and Krishnan, M.R., 2017. Demalvertising: A Kernel Approach for Detecting Malwares in Advertising Networks. In Proceedings of the First International Conference on Intelligent Computing and Communication (pp. 215–224). Springer, Singapore.
117. Patil, K., 2016. Request dependency integrity: validating web requests using dependencies in the browser environment. International Journal of Information Privacy, Security and Integrity, 2(4), pp. 281–306.
118. Banerjee, A., Rahman, M.S. and Faloutsos, M., 2011. SUT: Quantifying and mitigating url typosquatting. Computer Networks, 55(13), pp. 3001–3014.
119. Szurdi, J., Kocso, B., Cseh, G., Spring, J., Felegyhazi, M. and Kanich, C., 2014, August. The Long "Taile" of Typosquatting Domain Names. In USENIX Security Symposium (pp. 191–206).
120. Almeida, Tiago, Renato Moraes Silva, and Akebo Yamakami. "Machine learning methods for spamdexing detection." International Journal of Information Security Science 2, no. 3 (2013): 86–107.
121. Geng, G.G., Wang, C.H. and Li, Q.D., 2008, January. Improving Spamdexing Detection Via a Two-Stage Classification Strategy. In Asia Information Retrieval Symposium (pp. 356–364). Springer, Berlin, Heidelberg.

122. Abou-Assaleh, T. and Das, T., 2006, November. Combating spamdexing: Incorporating heuristics in link-based ranking. In International Workshop on Algorithms and Models for the Web-Graph (pp. 97–106). Springer, Berlin, Heidelberg.
123. Shahriar, H., Haddad, H. and Devendran, V.K., 2015. Request and Response Analysis Framework for Mitigating Clickjacking Attacks. International Journal of Secure Software Engineering (IJSSE), 6(3), pp. 1–25.
124. Johns, M. and Lekies, S., 2013, October. Tamper-resistant likejacking protection. In International Workshop on Recent Advances in Intrusion Detection (pp. 265–285). Springer, Berlin, Heidelberg.
125. Sarjaz, B.S. and Abbaspour, M., 2013. Securing BitTorrent using a new reputation-based trust management system. Peer-to-Peer Networking and Applications, 6(1), pp. 86–100.