Honeypot Boulevard: Understanding Malicious Activity via Decoy Accounts

Jeremiah Onaolapo
Department of Computer Science
University College London

Thesis submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy
of
University College London

January 2019

Declaration

I, Jeremiah Onaolapo, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Jeremiah Onaolapo

Abstract

This thesis describes the development and deployment of honeypot systems to measure real-world cybercriminal activity in online accounts. Compromised accounts expose users to serious threats including information theft and abuse. By analysing the modus operandi of criminals that compromise and abuse online accounts, we aim to provide insights that will be useful in the development of mitigation techniques.

We explore account compromise and abuse across multiple online platforms that host webmail, social, and cloud document accounts. First, we design and create realistic decoy accounts (honeypots) and build covert infrastructure to monitor activity in them. Next, we leak credentials of those accounts online to lure miscreants to the accounts. Finally, we record and analyse the resulting activity in the compromised accounts.

Our top three findings on what happens after online accounts are attacked can be summarised as follows. First, attackers that know the locations of webmail account owners tend to connect from places that are closer to those locations. Second, we show that demographic attributes of social accounts influence how cybercriminals interact with them. Third, in cloud documents, we show that document content influences the activity of cybercriminals. We have released a tool for setting up webmail honeypots to help other researchers that may be interested in setting up their own honeypots.

Impact Statement

It is hard for researchers to study what happens to online accounts during illicit accesses unless such researchers are in control of a large online service. The resulting research gap inspired us to develop new ways to make it possible for researchers to carry out such studies. To this end, we designed, developed, and deployed honeypot systems across various online services. Our approach enables us to obtain and analyse primary data from compromised accounts, and will help other researchers to achieve similar results.

It is important for defenders to understand the behaviour of attackers at all times to keep defence mechanisms up to date. In view of this, it is possible to commercialise our work by building custom honeypot services for organisations that wish to protect their online assets from cybercriminals. Such honeypot systems will be deployed along with "tripwire" mechanisms to raise alerts when criminals gain access to privileged information. These systems would provide useful real-time information about attacker behaviour and such information can be used to train and improve detection and mitigation systems. Similar to our approach, CounterCraft, a European company, has built a commercial "Cyber Deception Platform" that deploys decoy online assets (for instance, virtual machines, documents, and mobile apps) to deceive attackers and collect valuable threat intelligence and protect organisations.[1] This shows that our honeypots indeed possess potential commercial value.

We have disseminated some of our findings to the security community. In other words, we presented peer-reviewed papers at international conferences, workshops, symposia, and invited talks (including a 2016 guest lecture in UCL's Crime Science

[1] https://www.countercraft.eu/

MSc programme). In addition to expanding the security community's knowledge of malicious activity, we also released an open-source tool for deploying webmail honeypots,[2] which researchers in Utrecht University (the Netherlands) are now using to carry out further research. These demonstrate the growing impact of our work on the academic community, in terms of contributions to practical knowledge and tools to make online activity safer for everyone.

Outside academic circles, the general public has benefited from our work via considerable press coverage by BBC News[3] and other news outlets. This has helped to raise public awareness about what happens to compromised accounts and ways to stay safe online. In 2016, the author presented our work at the UK Home Office (the government department responsible for security). Similarly, the author presented our work to industry experts, government agencies, and academics at the 2017 Academic Centres of Excellence in Cyber Security Research conference in Nottingham (UK). Finally, our work on social honeypots has won a "Secure the Internet" grant from Facebook.[4] Our work on webmail honeypots was a finalist in the 2017 Europe Cyber Security Awareness Week in Valence (France). Once again, these demonstrate the impact of our work on various sectors outside academia, towards solving real-world problems and engaging with the general public to reduce cybercrime.

[2] https://bitbucket.org/gianluca_students/gmail-honeypot

[3] https://www.bbc.co.uk/news/technology-37510501

[4] https://research.fb.com/facebook-awards-more-than-800000-in-secure-the-internet-grants/

Acknowledgements

First and foremost, I would like to thank my supervisor, Gianluca Stringhini, for his mentorship in research and scholarly work. I deeply appreciate his friendly guidance, leadership by example, and timely provision of research resources throughout my PhD programme. I also thank my second supervisor, Emiliano De Cristofaro, for ensuring that my research journey progressed as planned.

I am thankful to my coauthors in UCL (London), Telefonica I+D (Barcelona), and other research institutions around the world, for demonstrating true scholarly collaboration. I am also grateful to my collaborators in Google and Facebook. Their help and advice enabled me to scale up experiments and minimise problems that would have emerged otherwise.

My research journey would have been boring without my colleagues in the UCL Information Security research group. I would like to thank them for bringing excitement to the journey. I will miss the table football matches and other social activities we enjoyed together. They also gave advice on my manuscripts and presentations. I would like to say thanks to staff and students of the UCL Computer Science department for their support throughout my PhD journey in UCL.

Many thanks to the Petroleum Technology Development Fund (Nigeria) for sponsoring my research work and taking care of my living expenses in the UK.

Finally, I would like to thank my family for their support and encouragement on this journey — they made sure I did not walk alone. To everyone else not mentioned here, I would like to express my heartfelt appreciation to them.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

> *"Hardware is easy to protect: Lock it in a room, chain it to a desk, or buy a spare. Information poses more of a problem. It can exist in more than one place; be transported halfway across the planet in seconds; and be stolen without your knowledge."*
>
> – Bruce Schneier

Total global spending on cybersecurity from 2017 to 2021 will likely exceed 1 trillion dollars, according to a 2018 report by Cybersecurity Ventures.[1] This shows the massive impact (and cost) of cybercrime on organisations and individuals alike. A 2014 study revealed that unauthorised parties had gained access to online accounts that belonged to 30% of participants [85]. Recent colossal data breaches further highlight the magnitude of cybercrime and its effects — these incidents include Yahoo (3 billion compromised accounts), Adult Friend Finder (412.2 million compromised accounts), and eBay (145 million compromised accounts).[2]

Malicious online activity perpetrated by cybercriminals include email spamming [48], malware dissemination [26, 1], phishing [47], information theft [25, 90], social spamming [45, 93, 108, 98], unauthorised crawling [57], account hijacking [25, 41] and Denial-of-Service attacks (DOS) [9], among others. To facilitate such activity, cybercriminals rely on an underground ecosystem of interconnected merchants trading

[1] https://cybersecurityventures.com/cybersecurity-market-report/

[2] https://www.csoonline.com/article/2130877/data-breach/the-biggest-data-breaches-of-the-21st-century.html

fake and compromised accounts, botmasters in control of massive botnet infrastructure, malware developers and distributors, and other actors [92, 98, 26]. To mitigate malicious online activity, it is important to disrupt the operations of the underground ecosystem. This requires a deep understanding of that ecosystem.

It is difficult to study the activity of criminals that specialise in stealing and selling online accounts, without being in charge of an online service, for instance, Google or Facebook. In other words, it is difficult for researchers to gain access to private compromised accounts to study the behaviour of criminals in them. Hence, there is limited research literature in this space (in Chapter 2, we discuss this in detail). The rare exceptions are studies that analysed publicly available account information, for instance, posts made on Twitter by compromised accounts [41, 42].

To close this research gap, we develop new ways and infrastructure (honeypot systems) to study compromised accounts without being in control of the online services that host them. We focus on hijacked webmail accounts, social accounts, and cloud documents. To this end, we construct realistic decoy accounts and documents, which we refer to as *honey assets* (honey accounts or honey documents). We deploy those honey assets, record accesses to them using our honeypot infrastructure, analyse the resulting data, and draw inferences on malicious activity in compromised accounts. By relying on honey assets instead of real accounts that belong to real users, we ensure that no harm happens to anyone during our experiments. We discuss ethical considerations in more detail in Chapter 4 (Section 4.4.5), Chapter 5 (Section 5.4.5), and Chapter 6 (Section 6.4.5) respectively.

In this thesis, we present multiple findings that provide the research community with a better understanding of what happens when online accounts are attacked. For instance, we discovered that attackers that know the locations of webmail account owners tend to connect from places that are closer to those locations. We infer that this is an attempt to evade current security mechanisms employed by online services to discover suspicious logins. Also, in webmail and social accounts, search terms revealed that behavioural modelling could work in identifying anomalous behaviour in online accounts. In webmail accounts, we observed that search terms mostly contained financial/sensitive information while search terms recorded in social accounts indicated less interest in financial information. In cloud documents, we discovered that the activity of cybercriminals varied, depending on sheet content. For instance, we recorded more modifications in sheets containing cryptocurrency information than sheets containing traditional banking information.

Other observations in Chapters 4 (webmail accounts), 5 (social accounts), and 6 (cloud documents) include activity timing, modifications made to online assets, and differences in account activity depending on demographic attributes of online accounts, among others. In Chapter 7, we discuss the implications of those findings.

Our work contributes to the security community by shining light on the activity of cybercriminals and providing new ways to study compromised online accounts. We discuss existing research literature, point out research gaps, and present our honeypot approach to studying online accounts. We also describe our experiments, present our findings and what they imply, and highlight what remains to be done. We are hopeful that this work will provide new insights, tools, and techniques for online service providers, fellow researchers, and other parties seeking to mitigate cybercrime and make online activity safer for everyone.

1.1 Thesis statement

There is limited research work on activity within online accounts after criminals gain access to them. This is because it is hard to study online accounts without being in control of a large online service (say Facebook or Gmail), with the exception of publicly available account data (for instance, Twitter). Through the lenses of honeypots, we can provide a deeper understanding of what happens to such accounts, provide tools and techniques for researchers to carry out further studies, and bridge the existing research gap. Our findings will also provide insights that can be used to improve detection and mitigation systems that protect online users from cybercriminals.

1.2 Contributions

Overall, this thesis makes the following contributions:

- To achieve the goal of understanding malicious activity in online accounts, we propose a systems-based life cycle approach to the development and deployment of honeypots, and identify a set of minimum requirements, with careful consideration for research ethics to avoid harming people.

- We design and develop a system to monitor activity in Gmail accounts towards understanding malicious activity in compromised webmail accounts. We publicly release the source code of our system[3] to allow other researchers to deploy their own Gmail accounts for related studies. To the best of our knowledge, it is the first publicly available Gmail honeypot infrastructure.

- We design and develop another system to instrument and monitor compromised social accounts on Facebook, and perform large-scale experiments to observe differences in account activity per demographic attributes of the accounts.

- We introduce some improvements to the cloud document monitor system originally proposed in a 2016 USENIX workshop paper [62]. To understand what happens to compromised cloud documents, we then create and deploy Google spreadsheets containing fake banking records and cryptocurrency information (fake financial details).

- We present detailed analysis of activity in compromised webmail accounts, social accounts, and cloud documents. We also discuss the implications of our findings, especially for online services seeking to improve their detection and mitigation techniques and systems.

Parts of the work in this thesis have been peer-reviewed and presented in top conferences and workshops. In addition, some parts have received considerable

[3] https://bitbucket.org/gianluca_students/gmail-honeypot

press coverage on BBC News,[4] Huffington Post,[5] and The State of Security,[6] among other news outlets. This shows that our work has contributed to the research community and increased the awareness of the general public about compromised online accounts. Overall, this will lead to safer online activity for everyone.

1.3 Peer-reviewed papers

As we previously mentioned, parts of the work in this thesis have been published in peer-reviewed conferences and workshops, in collaboration with other researchers. Some aspects of our honeypot infrastructure and findings appear in the following papers.

- Adrian Bermudez Villalva, Jeremiah Onaolapo, Gianluca Stringhini, Mirco Musolesi. Under and over the surface: a comparison of the use of leaked account credentials in the Dark and Surface Web. In *Crime Science (Journal)*, 2018.

- Emeric Bernard-Jones, Jeremiah Onaolapo, and Gianluca Stringhini. BABELTOWER: How Language Affects Criminal Activity in Stolen Webmail Accounts. In *Companion Proceedings of The Web Conference (WWW)*, 2018.

- Jeremiah Onaolapo, Enrico Mariconti, and Gianluca Stringhini. What Happens After You Are Pwnd: Understanding the Use of Leaked Webmail Credentials in the Wild. In *ACM Internet Measurement Conference(IMC)*, 2016.

- Martin Lazarov, Jeremiah Onaolapo, and Gianluca Stringhini. Honey Sheets: What Happens to Leaked Google Spreadsheets? In *USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2016.

Collaborators. In Chapters 4, 5, and 6, we mention our academic and non-academic collaborators, and acknowledge their contributions in detail.

[4] https://www.bbc.co.uk/news/technology-37510501

[5] https://www.huffingtonpost.co.uk/entry/what-hackers-actually-do-with-your-stolen-personal-information_uk_58049f32e4b0e982146cd18f

[6] https://www.tripwire.com/state-of-security/security-data-protection/heres-what-happens-after-your-webmail-account-is-compromised/

1.4 Scope of work

In this thesis, our scope of work encompasses studying malicious activity within compromised assets (with specific focus on online accounts and cloud documents) to shed light on the modus operandi of criminals that connect to stolen accounts. Our work does not directly study sales of stolen information (products and prices), or activity that is external to the accounts under study. Those topics are outside the scope of this work. Although we discuss possible ways to apply our findings to the development of better automatic detection and mitigation systems, specific implementations of such systems (machine learning approaches, for instance) are outside the scope of this work. Instead, we focus on exploring new ways and systems to study malicious activity in online accounts.

1.5 Thesis outline

The remainder of this thesis is organised as follows. In Chapter 2, we discuss ways through which criminals steal online accounts, how they misuse the stolen accounts, and victimise online users. We also discuss existing research literature, identify research gaps, and discuss the justification for our research approach (honeypots). Chapter 3, which strongly interconnects the remaining chapters, provides a high-level discussion of our honeypot approach, with focus on minimum requirements and our honeypot development life cycle. We also discuss merits and limitations of our approach, in addition to alternative methods. The next three chapters build on the approach proposed in Chapter 3. Chapter 4 presents our Gmail honeypot. It also describes our experiments on compromised Gmail accounts and resulting findings. Similarly, Chapter 5 presents our large-scale Facebook honeypot, experiments on compromised social accounts, and our resulting findings. Chapter 6 presents our cloud document honeypot, experiments on compromised cloud spreadsheets, and our findings. Finally, in Chapter 7, we discuss the implications of our findings and suggest ways to improve existing detection and mitigation systems of online services. We also discuss implementation-specific limitations of our honeypots and potential future work.

Chapter 2

Literature Review

In this chapter, we explain various ways by which cybercriminals gain illegitimate access to online accounts and misuse them. We also discuss how to detect such malicious activity. Finally, we explore the role of decoy accounts in understanding what happens to compromised accounts.

2.1 Stealing online accounts

Cybercriminals can gain access to online accounts using various methods and tools. These include botnets, data breaches, and account hijacking. We will focus on them since they are particularly relevant to the work in this thesis.

2.1.1 Via botnets

A botnet is a huge network of compromised computers (also known as bots) that receive instructions from one or more Command-and-Control (C&C) servers under the control of a botmaster [90]. The legitimate administrators and users of such compromised computers are usually oblivious to the fact that their machines have become members of the bot network. Botnets are often used to send spam [59] and steal sensitive information in bulk, for instance, online banking credentials [19]. Cybercriminals also use botnets to stage Distributed Denial-of-Service (DDoS) attacks on victims' network infrastructure, as seen in the 2016 *Mirai* DDoS attack on

Brian Krebs' cybersecurity blog.[1] Common means by which vulnerable machines are "enlisted" into botnets include drive-by downloads [67] and malware delivered through phishing or spam emails [1]. Communication links between bots and C&C servers are usually established via IRC, HTTP, or P2P channels, depending on the organisation of the botnet [90]. Bots are not always desktop or laptop computers — social accounts [21] and IoT devices [9] can be enlisted as bots as well.

A *socialbot* is software that masquerades as a real user in an Online Social Network (OSN) [21]. Socialbots post messages, upload content, and send connection requests to other accounts on OSNs. Similar to the traditional botnets explained earlier, socialbots are also controlled by botmasters. When socialbots infiltrate social graphs of unsuspecting victims, they have the ability to harvest personal data of their victims (by scraping their profile pages) [21]. Such valuable data can be used or sold by the botmaster for further nefarious operations, including spamming [44], phishing [93], and identity theft [18]. Socialbots have allegedly been involved in attempts to sway elections[2] and dissemination of fake news [113, 111, 112]. These show that socialbots, like traditional botnets, have the capacity to inflict substantial harm on victims.

Existing botnet mitigation techniques include *infiltration* and *hijacking* [90] which enable defenders to learn about and take over botnet communications, towards disrupting the cybercriminal operation(s) behind the botnet. Both mitigation techniques are costly and time-limited in the face of advanced botnets, since they usually involve reverse engineering malware binaries and communication protocols [90]. These are non-trivial tasks, and botnets continually evolve in ways that defeat existing countermeasures.

2.1.2 Via data breaches

Another way through which cybercriminals compromise online accounts is by mounting information-stealing attacks on vulnerable servers and terminals, often resulting

[1] https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/

[2] http://uk.businessinsider.com/twitter-russia-facebook-election-accounts-2017-10

in massive data breaches. The techniques they employ include *SQL* injection [46], password guessing [103], and social engineering attacks [37], for instance, by tricking employees of the target organisation to give up their authentication credentials.

Recent massive data breaches include Yahoo (3 billion compromised accounts), Adult Friend Finder (412.2 million compromised accounts), and eBay (145 million compromised accounts) incidents.[3] Given the scale, severity, and frequency of data breaches in recent times, it is important for the security community to find lasting solutions to this ongoing problem. This constitutes the primary motivation for the work in this thesis towards understanding what cybercriminals do with stolen online accounts. In other words, since data breaches cannot be completely mitigated yet, it is important for the security community to understand what cybercriminals do with stolen accounts, post-compromise, to help develop better detection and mitigation systems.

Data breaches are often compounded by the problem of password reuse across various online services [35]. Also, as the security community knows quite well, strong passwords place heavy burdens on users and this leads to usability issues [55, 66]. This has brought about a situation in which users often opt for memorable but weak passwords to secure their accounts. The combined problem of password reuse and weak passwords makes it easy for criminals to breach accounts across multiple services, even the ones that did not suffer direct data breaches. Existing countermeasures include cryptographic hashing [80], password managers [86], multi-factor authentication [32], public-key authentication [81], and proximity authentication [88].

2.1.3 Via account hijacking

As explained earlier, online accounts are valuable resources and are attractive targets in the eyes of cybercriminals. They hijack accounts to gain access to the wealth of information stored in them. Webmail accounts, for instance, often become "hubs"

[3] https://www.csoonline.com/article/2130877/data-breach/the-biggest-data-breaches-of-the-21st-century.html

that accumulate sensitive information like credit card details, password reset information, and government identification documents, as a result of regular everyday use (see Figure 2.1). The immediate implication is that a successful attack on a webmail account can lead to a chain of further attacks on other accounts linked to that webmail account.

Automated hijacking is usually carried out using botnets (as described in Section 2.1.1), while manual hijacking is facilitated mostly by low-volume spearphishing attacks on unsuspecting victims [25]. When manual hijacking attacks succeed, the cybercriminal usually performs a quick assessment of the stolen accounts to determine their value and decide what to do with them — usually to sell the account credentials in an underground market or discard them, depending on the perceived value of each account. Hijacked accounts can also be used to send spam and phishing messages to exploit the existing trust between the victim and their contacts. This is because spam filters are more likely to allow messages from known contacts to pass through them [41, 25].

According to Bursztein et al. [25], detecting manual hijacking activity is more difficult than detecting automated hijacking activity. This is because manual hijacking is a low-volume activity and manual hijackers mimic normal users. Thus, it is difficult to tune error rates of automatic detection systems to discover manual hijacking incidents. In addition, manual hijackers are usually skilful enough to know how to evade detection. In Chapter 4, we show this in detail. It further highlights the need to study the modus operandi of manual hijackers closely — this is the main motivation underpinning our work.

2.2 Misuse and abuse

We have discussed how cybercriminals perpetrate information theft via botnets, data breaches, and manual hijacking, and how they harm victims. In this section, we discuss ways by which cybercriminals misuse online accounts and abuse their victims.

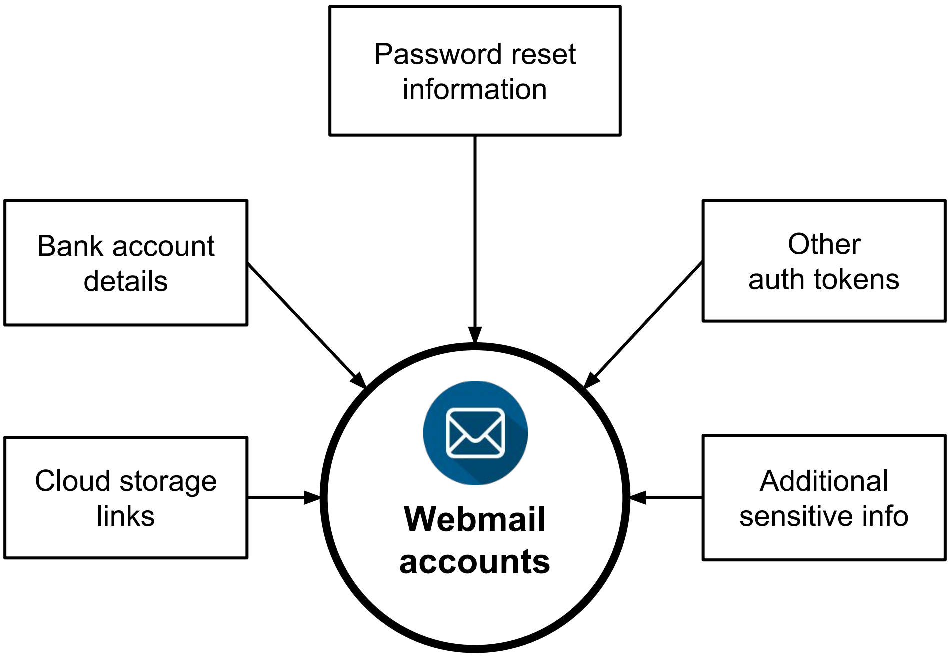


Figure 2.1: Webmail accounts, like most types of online accounts, accumulate sensitive information with regular use. This sensitive information attracts cybercriminals seeking to steal and monetise sensitive personal information.

2.2.1 Information theft and misuse

As we mentioned earlier, criminals leverage botnet infrastructure or mount phishing attacks [47] to steal sensitive information from victims. For instance, the Zeus malware family (also known as Zbot) [19] steals login credentials and sends them to C&C servers. Other information-stealing malware include Corebot and Dridex. As of 2015, Dridex likely brought about losses amounting to 100 million dollars worldwide.[4] This shows the magnitude of harm that results from information theft and highlights the importance of further studies in this space, to mitigate harm. After stealing sensitive information from victims, cybercriminals stockpile that information for later use, or sell it via dark markets [31], underground forums [69], and paste sites, among other outlets. Illicit uses of stolen information include spamming (for instance, using stolen authentication credentials), spearphishing attacks [94] (aided by knowledge of private information), and blackmail attacks,[5] among others.

2.2.2 Spam

Spam, which can be defined as unsolicited messaging, is a problem that has plagued online services for a long time, including but not limited to webmail services [39] and social networks [93]. These services attract many users and collect huge quantities of personal and sensitive data, as we explained in Section 2.1.3. This, in turn, attracts cybercriminals seeking treasure troves of sensitive data. To gain illicit access to this treasure, they leverage botnets to send bulk unsolicited messages (spam) usually containing malicious payloads to unsuspecting victims. Sometimes, they send targeted malicious messages, for instance, during spearphishing attacks. However, not all spam messages contain malicious payloads — some contain benign newsletters, marketing offers, and other information from non-malicious entities. Spam distribution results in substantial earnings for cybercriminals — on the other hand, spam prevention efforts by the security community are disproportion-

[4] https://www.theguardian.com/technology/2015/oct/13/nca-in-safety-warning-after-millions-stolen-from-uk-bank-accounts

[5] https://www.wired.com/2015/08/happened-hackers-posted-stolen-ashley-madison-data/

ately costly, according to a 2013 study by Anderson et al. [7].

2.2.3 Scams

The main motive of cybercriminals has shifted from mischief and fun to financial gain [53]. One of the ways by which cybercriminals obtain such illicit gain is by scamming unsuspecting victims. This brings the infamous *419 scams* to mind. A 419 scam is any scheme designed to fleece people of their money, as defined in Section *419* of the Nigerian Criminal Code [100]. A classic *419* scam starts with a message from a scammer to the potential victim, usually describing immense wealth (fictional) that would purportedly benefit the victim. Once the scammer gains the trust of the potential victim, the scammer asks the victim to pay a “small” fee to process a big “reward.” When the victim pays the fee, then the scam is complete. The scammer will not communicate with the victim anymore. Even if the scammer resumes communication, it will usually be an attempt to coax the victim into sending more money [40]. These operations are mostly carried out manually and often rely on the victim’s greed or pity to succeed [56].

Apart from 419 scams, there are other popular variants of online scams, including *dating scams* or *romance scams* that leverage dating websites. Cybercriminals target them, set up fake profiles, and seduce vulnerable users seeking companionship [107]. When they gain the trust of their victims, the scammers make demands for funds to handle spurious matters, including visa processing and flight tickets. Some scammers request expensive flower baskets [52].

Scam letters were initially sent via *snail mail* (post) long before the era of the Internet. With the proliferation of electronic communication via the Internet, 419 scam messages are now sent *en masse* to potential victims via email and fax [40]. Online scams cause heavy financial and psychological losses to victims, and are not fully understood yet [52, 68].

2.2.4 Cyberbullying

Social networks have evolved from online venues where users connect with friends, loved ones, and strangers alike, to colossal platforms and ecosystems comprising marketplaces, news outlets, and much more. Users spend a lot of time interacting on these platforms. Some users make a living from their presence on social networks, for instance, Instagram celebrities with millions of followers. However, social networks are also known to attract toxic behaviour and cyberbullying.

Previous studies have investigated the roles of cyberbullies, victims, bystanders, and how they interact [4, 11, 38]. Other studies explored the detection of cyberbullying in social networks [72, 71, 73], toxic online forums [17, 50] and gaming communities [20]. The effect of anonymity on toxic online behaviour has also been investigated [17, 51, 77]. It is important to note that toxic behaviour is not contained within toxic online communities. For instance, there is evidence that coordinated attacks originate from `4chan.org` (an online forum) towards users of other services. Hine et al. [50] studied this problem and proposed an algorithm to detect such coordinated attacks.

2.3 Detecting malicious activity

The problem of malicious activity in online accounts has generated a lot of interest in the research community. In this section, we review key studies that focus on the detection and mitigation of malicious activity in online accounts.

2.3.1 Understanding manual hijacking

Bursztein et al. [25] investigated manual hijacking of accounts rather than automatic hijacking by botnets. They show that manual hijacking is not common, and demonstrate that phishing is the primary method that manual hijackers use to acquire user credentials. The study illustrates the importance of decoy account credentials in understanding malicious activity. Other studies have also leveraged decoy credentials and fake documents to study malicious activity [36, 62], and we discuss them further in Section 2.4.

Cybercriminals prefer to operate from compromised accounts (rather than fake accounts) since malicious activity is harder to detect in compromised accounts that belong to real users. Egele et al. [41] presented COMPA, a tool that detects malicious activity in online social networks by building statistical models of normal behavioural patterns of users. Deviations from such behaviour can then be used to detect compromised accounts.

Similarly, Stringhini et al. [94] developed a tool for the detection of spearphishing attacks based on behavioural modelling of senders. The tool looks out for anomalous email sending behaviour and writing habits, rather than checking the email content for suspicious words like traditional spam filters do. To understand the phishing ecosystem, Han et al. [47] deployed sandboxed phishing kits, recorded live interactions of various parties with the kits, and shed light on the phishing life cycle.

2.3.2 Understanding spam

Thomas et al. [99] studied Twitter accounts under the control of spammers. They discussed the modes of operation of cybercriminals that disseminate spam. They also reported that the majority of spam accounts rely on unsolicited mentions and hashtags to reach audiences that are wider than their limited social connections. In addition, they identified an emerging ecosystem of social spamming services (including affiliate programmes). However, the study did not propose detection mechanisms to discover the operations of spammers, despite pointing out that the underground spamming ecosystem is largely undisturbed by Twitter's current defence mechanisms. Stringhini et al. [93] studied social spam using 900 honeypot profiles and presented a tool for spam detection on Facebook and Twitter. In addition, [106, 21, 63, 13, 98] studied the problem of social spam and [14] applied machine learning techniques to distinguish legitimate users from video spammers and content promoters on *YouTube*, a popular online video social network.

Similarly, Stringhini et al. [92] studied the email spam ecosystem by advertising unique honeypot email addresses on the web. The study described the relationships among various actors in the spam landscape (deduced from statistical correlation), namely email harvesters, spammers, and botmasters. Stone-Gross et al. [91] studied a large-scale spam operation by analysing 16 C&C servers of *Pushdo/Cutwail* botnet. Other studies in email spam literature explored network-level spam detection approaches [48], statistical/machine-learning approaches [83, 39, 95], and the underground ecosystem that drives spam [91].

Detecting fake accounts. Fake accounts play a major role in the problem of spam. Wang et al. [104] proposed the use of patterns of click events to spot fake accounts, otherwise known as Sybils, in online services by building clickstream models of real users and fake accounts. They trained machine learning tools to spot fake accounts based on those clickstream models. Yang et al. [109] and Cao et al. [27] also proposed ways to uncover fake accounts on social networks.

2.3.3 Defeating information theft

Stone-Gross et al. [90] hijacked the Torpig botnet for ten days by taking advantage of weaknesses in communication protocols of the botnet. Their method of sinkholing all data sent from bots to the C&C server they hijacked is similar to our approach of sinkholing all emails sent from honey webmail accounts in Chapter 4. In 2012, Liu et al. [65] studied content privacy issues in Peer-to-Peer (P2P) networks by deploying honey files containing honey account credentials in P2P shared spaces. They monitored download events and concluded that attackers that downloaded the honey files had malicious intentions to make economic gain from the private data they obtained. They employed a similar approach to ours — in this thesis, we place decoy account credentials in strategic locations for cybercriminals to find and misuse them. However, they studied P2P networks while this thesis focuses on online accounts.

2.4 Honeypot boulevard

In this thesis, we employ honeypots, having observed the importance of well-designed honeypots in previous work. Hence, instead of exposing real users to malicious activity and potentially harming them in the process, we instead set up realistic honeypots and lured cybercriminals to them, to measure malicious activity in the wild. Next, we present an overview of honeypots and how they have evolved over time.

2.4.1 What is a honeypot?

A honeypot is a resource designed to receive unauthorised interactions. Unlike other security mechanisms that are designed to keep attackers away from protected assets, the value of a honeypot lies in its misuse by attackers [87]. Any attempt to access a honeypot should be considered suspicious [79]. Honeypots can be physical or virtual. A physical honeypot is a computer with its own IP address, while virtual honeypots are simulated atop real machines — the TCP/IP stack of the virtual machine (VM) is designed to appear similar to a real machine [79].

In addition to physical and virtual machines, files and online accounts can also be deployed as honeypots, as we did in Chapters 4, 5, and 6. A honey file is a bait file that triggers alarms when accessed, and it masquerades as a normal file with some inherent value [110]. Honey files are usually positioned in regular user file spaces, along with bait, for instance, attractive file names like `passwords.xls` or `account_details.txt` (to lure attackers). Attacker operations will then be logged during accesses to honey files [84, 65].

Honeypots have been around for ages. In the 1980s, Stoll [89] tracked down a German hacker who remotely gained unauthorised access to a computer network at the Lawrence Berkeley National Laboratory (LBNL). To achieve this, Stoll deployed a honeypot, among other tools. Similarly, in the 1990s, Cheswick led a hacker on a wild goose chase by tricking the hacker into believing they had accessed password files and vulnerable assets on an AT&T gateway machine [28].

Recent honeypots are more advanced. For instance, honeypots have been successfully deployed to study malware in the wild [10, 105], infiltrate botnets [90], and track social spam [106, 63]. According to Spitzner [87], honeypots can also be used to mitigate insider threats in organisations. This shows that honeypots are useful in the study of malicious activity targeting online accounts and users, and it justifies our use of honeypots.

2.4.2 Selected honeypot studies

Honeypots are usually built to deceive attackers and they can be used to detect unauthorised access to privileged information, record the behaviour of attackers after gaining such unauthorised access, or both. In this section, we present a selection of studies that illustrate deception-based techniques and honeypot usage in prior work.

Virvilis et al. [101] broadly explored deception as a defence approach, compared Advanced Persistent Threats (APTs) to insider attackers, and discussed the use of deception methods to detect sophisticated attackers. Achleitner et al. [2] and Chiang et al. [29] developed systems to disrupt reconnaissance activity (scanning) originating from attackers (APTs, for instance) within a network. Bowen et al. [23] proposed a way to automatically generate and inject decoy network traffic to ensnare and uncover eavesdroppers on a computer network. Bowen et al. [24] leveraged decoy credentials, also known as honeytokens, to “bait and delude” information-stealing malware to reveal itself. Bercovitch et al. [15] developed HoneyGen, a tool that automatically generates realistic honeytokens based on rules derived from real tokens. HoneyGen requires a high-quality input database of real tokens. Bowen et al. [22] developed a deception system based on decoy documents and decoy credentials, to discover insider attackers that attempt to exfiltrate sensitive information. They also formalised a set of properties (requirements) towards the design and implementation of honeypots.

Vrable et al. [102] built a prototype honeypot system that was able to run tens of thousands of virtual honeypots on a few physical servers. Alata et al. [5] recorded and studied the activity of attackers that gained access to a compromised machine via Secure Shell (`ssh`). They tried to distinguish between human attackers and automated programs by analysing the way attackers entered commands in the `ssh` terminal. Chin et al. developed HoneyLab, an infrastructure that allows various entities to deploy honeypots on shared computing infrastructure [30]. Mulliner et al. [70] proposed HoneyDroid, an Android-based honeypot that runs on real mobile phone hardware. Nazario [74] developed a virtual web client honeypot that can carry out dynamic analysis of JavaScript and Visual Basic Script, among others.

Kedrowitsch et al. [60] explored ways to improve Linux sandboxes for analysis of evasive malware. Barron and Nikiforakis [12] deployed honeypot machines and observed how system properties of those machines influenced the behaviour of attackers. Similarly, online accounts can be repurposed to study the operations of cybercriminals that interact with them. For instance, honeypots based on online accounts have been deployed to study social spam in OSNs [106, 63, 93] and email spam [92]. DeBlasio et al. [36] studied compromised websites by registering on those websites using honey webmail accounts. They monitored illegitimate accesses to the honey accounts that happened as a result of data breaches on those websites. They observed attackers that leveraged the problem of password reuse across online services. Other studies also investigated the behaviour of criminals in compromised webmail and cloud document accounts via honeypots [25, 76, 62].

The vast majority of existing research literature focuses on detecting malicious accesses, as these studies show, along with the ones mentioned previously in this chapter. On the other hand, the core of our work in this thesis focuses on analysing malicious activity, that is, post-compromise attacker behaviour.

2.4.3 Honeypots in politics

During the 2017 presidential campaign in France, the Macron campaign organisation devised an ingenious way to defeat hackers that sought to compromise their webmail accounts. The campaign organisation turned their own webmail accounts into a “tarpit” defence system by pre-stuffing those accounts with useless data (that is, useless for hackers). Thus, they wasted the resources of hackers that eventually breached those accounts.[6] This further highlights the validity of the honeypot approach to studying malicious activity in the wild. Those webmail accounts can be thought of as "time-wasting" honeypots, since they were deployed to anticipate hackers and waste their time. In contrast, our webmail honeypots (in Chapter 4), which predate the Macron honeypots, primarily track the actions of criminals in compromised webmail accounts — time wasting is an optional feature.

2.5 Research problem

There are many aspects of cybercrime that are not yet fully understood. We are particularly interested in this question — what do cybercriminals do with compromised accounts? They are hard to detect by existing automated scanning systems. As we stated earlier, this is because their interactions are manual and stealthier than automated activity, due to human intelligence and adaptation [52, 25]. As a result, cybercriminals appear to be winning the attackers-defenders game since they continue to make profits while forcing corporations and governments to make huge, disproportionate investments in security mechanisms [54]. A recent study reports that "indirect and defence costs" of cybercrime are at least times ten of cybercriminals' earnings [7]. To help mitigate the problem, our work focuses on understanding what cybercriminals do with compromised accounts. Our work will help to reduce the costs incurred by law enforcement agencies and corporations in the pursuit of better security, by providing a deeper understanding into the modus operandi of cybercriminals that attack online accounts.

2.6 Conclusion

In this chapter, we explored the literature on how cybercriminals gain illicit access to online accounts, and how they abuse and misuse such accounts. We also reviewed previous work to highlight techniques for detecting and mitigating malicious activity

[6] https://www.theregister.co.uk/2017/05/08/team_macron_pre_hack_opsec

in accounts, and noted that it is hard to detect manual hijacking attacks. It is also hard to study online accounts without being in control of a large online service. Finally, we described the role of honeypots in understanding malicious activity and provided a strong basis for our work.

Chapter 3

Honey Assets Method

In Chapter 2, we discussed previous work on understanding malicious activity in online accounts and pointed out a research gap. We also presented an overview of honeypots, how they have been employed in previous work, and why we chose to rely on honeypots to shed light on activity in compromised online accounts and cloud documents. We present details of our *honey assets method* in this chapter. Honey assets refer to the fake entities that were exposed (intentionally) to cybercriminals during experiments, for instance, webmail credentials and accounts in Chapter 4, social credentials and associated accounts in Chapter 5, and finally, cloud documents and links that point to them in Chapter 6. Our honey assets method forms a strong link that interconnects those chapters. Later, in Chapters 4, 5, and 6, we present specific honeypot instances based on this approach.

3.1 Criminals, visitors, or both?

Our work involves building and deploying bait resources (honey assets) and observing accesses and activity in them. Depending on the dissemination vectors employed to expose credentials of honey assets, it is reasonable to expect a wide variety of visitors to honey assets, ranging from curious "benign" visitors to the ones with criminal intentions, for instance, visitors that intend to derive illicit profit from stolen accounts. This brings the following question to mind: what is the correct terminology for visitors to honey assets? Even though our work is not a discourse in legal terminologies, we defer to Section 1 of the UK Computer Misuse Act 1990[1] which states that a party is guilty if they knowingly attempt to gain unauthorised access to computer resources (paraphrased). In view of this, all "visitors" that intentionally connect to our honey assets are potential criminals. However, as we stated previously, there is the possibility that not all parties that connect to our honey assets have criminal intentions. Hence, we refer to them mostly as "visitors" and sometimes as "criminals" or "cybercriminals." Despite this relaxed nomenclature, it is important to note once again that unauthorised accesses to computer resources are unlawful and we do not condone such accesses in any way.

A possible alternative point of view to distinguish benign visitors from malicious visitors is to specify a threshold based on activity level. Visitors that carry out further actions on honey assets after initially accessing them can be considered to be malicious. For example, visitors that edit payment information on compromised payroll sheets can be considered to be more malicious than the ones that perform no action after accessing such compromised information. However, this approach is not robust since it is possible that the visitors that perform no action after access have made copies of the information they gained access to, for later use, while the ones that carried out actions may have done so out of curiosity. In fact, there might be visitors with good intentions that will delete sensitive content from the compromised documents to protect victims (by limiting the exposure of compromised content).

It is therefore obvious that specifying an activity threshold to distinguish benign visitors from criminals will not work well. Hence, in this thesis, we rely on the UK Computer Misuse Act 1990, as earlier described, and refer to all parties that accessed our honey assets as "criminals" or "visitors" (interchangeably).

[1] https://www.legislation.gov.uk/ukpga/1990/18/contents

3.2 System requirements

We identified the following minimum requirements for our honey assets — they have to be *Accessible*, *Realistic*, *Measurable*, *Ethical*, and *Robust* (*ARMER*).[2] We explain these requirements next.

Accessible. When creating honey assets, it is essential to ensure that they will be accessible to the intended audience and researcher(s) that will build the required honey assets. In other words, honey assets are best built and deployed on platforms that the intended audience (cybercriminals) already has access to, or can gain access to, without much hassle. The same applies to the builder of honey assets (the chosen platform must be accessible to the researcher that intends to carry out studies using honeypots). Examples of accessible platforms include free webmail, social, and dating services.

Realistic. For honey assets to be convincing to the target audience, they have to be designed and built to look similar to real-world examples. For instance, a honey webmail account, despite being fake, must look like a webmail account that belongs to a real user (we achieved this in Chapter 4, for instance). It is therefore important to pay particular attention to the content and presentation of honey assets. It is necessary to source content for honey assets from data sources that resemble content from real users. Thus, at a glance, honey assets derived from such content will appear believable to cybercriminals and other visitors that gain access to the honey assets. This will help to reduce potential bias that may arise if honey assets appear to be "weird," since such weirdness may affect the behaviour of visitors to them. It is also possible to generate realistic decoy data by leveraging existing tools built for that purpose, for instance, HoneyGen [15].

Measurable. It is important to create honey assets in a way that the researcher managing them can easily collect data from them and perform measurements. This is achievable by relying on a combination of intrinsic tools, for instance, Google

[2] ARMER is a memorable acronym, nothing more. It has no relation to weaponry.

Apps Script[3] in Gmail accounts (see Chapter 4), and extrinsic tools, for instance, scripts developed by the researcher to connect to honey assets and record activity information. Choices and decisions surrounding honey assets depend heavily on this requirement — it is pointless to build and deploy honey assets if it will be hard or impossible to collect activity information from them. However, the instrumentation of decoy assets must be carried out in a "hidden" way so that attackers cannot easily observe the presence of such instrumentation tools, since honeypots are designed to deceive attackers into believing they are interacting with real assets.

Ethical. It is necessary to ensure that honey assets are designed, built, and deployed in an ethical manner. The main ethical goal to consider is to ensure minimal harm to the intended audience for honey assets, for instance, by isolating potentially harmful honeypot environments. For instance, the researcher must keep all recorded activity data safe and not de-anonymise visitors to honey assets. Also, if experiments involve running live malware samples (for instance, in Chapter 4), adequate care must be taken to ensure that those malware samples do not harm any internal or external parties, by following standard practices in malware research [82]. Researchers that build social honeypots should pay particular attention to [34, 97]. It is also important to protect the researcher responsible for honey assets. Honey assets and honeypot systems must be designed in a way that minimises the possible harm that researchers may face, for instance, if their identities become known during experiments. Hence, it is essential to take advantage of Virtual Private Networks (VPNs) and proxies, and incorporate them in honeypot infrastructure when necessary. In Chapters 4, 5, and 6, we further discuss the steps we took to ensure that our experiments were conducted in an ethical manner.

Robust. Honey assets and honeypot systems, as explored in this thesis, depend on external platforms to function. For instance, studies on compromised webmail accounts rely on accounts that are hosted by a webmail service, usually not under the direct control of the researcher studying them. It is therefore necessary to

[3] https://developers.google.com/apps-script/overview

build fault-tolerant honeypot systems and honey assets, so that changes in external entities (webmail services, for instance) will not adversely affect experiments. However, sometimes it is impossible to build a honeypot that automatically adapts to all changes in related external entities. The researcher responsible for the honeypot system and honey assets must be ever ready to make minor changes to the honeypot to adapt to changes in such external entities. An example includes adding minor updates to scripts that track a specific web page in a honey asset, for instance. If the online service changes some elements of the user interface of that page, the honeypot researcher will then have to update their script to match those changes. In summary, it is necessary to build a robust honeypot and pay particular attention to it during operation, to make minor changes when necessary.

3.3 Target population

Before designing and developing honeypot infrastructure, one of the key considerations to keep in mind is the target population, in other words, the attackers/criminals under study. It may be beneficial to customise the proposed honeypot infrastructure to the target population. For instance, basic attackers may require less effort on the part of the researcher, especially towards ensuring realism in the honey assets, unlike sophisticated attackers with higher skill levels. This should influence design choices, including how and where to source data for honey assets. For instance, should we source high-quality data from related real-world activity[4] or "garbage" data[5] from automated tools? Similarly, the target population should be factored into design decisions on the scale of honey assets that will be deployed to study them – a handful of hand-curated honey assets or a plethora of mass-generated honey assets? It is up to the researcher to decide. It is important to note that these design choices will also influence experimental design (for instance, where, when, and how to leak honey credentials), and should be carefully analysed in advance.

[4] Example – we derived some data from real-world messages posted on Twitter in Chapter 5.

[5] Fake paper generator – https://pdos.csail.mit.edu/archive/scigen/

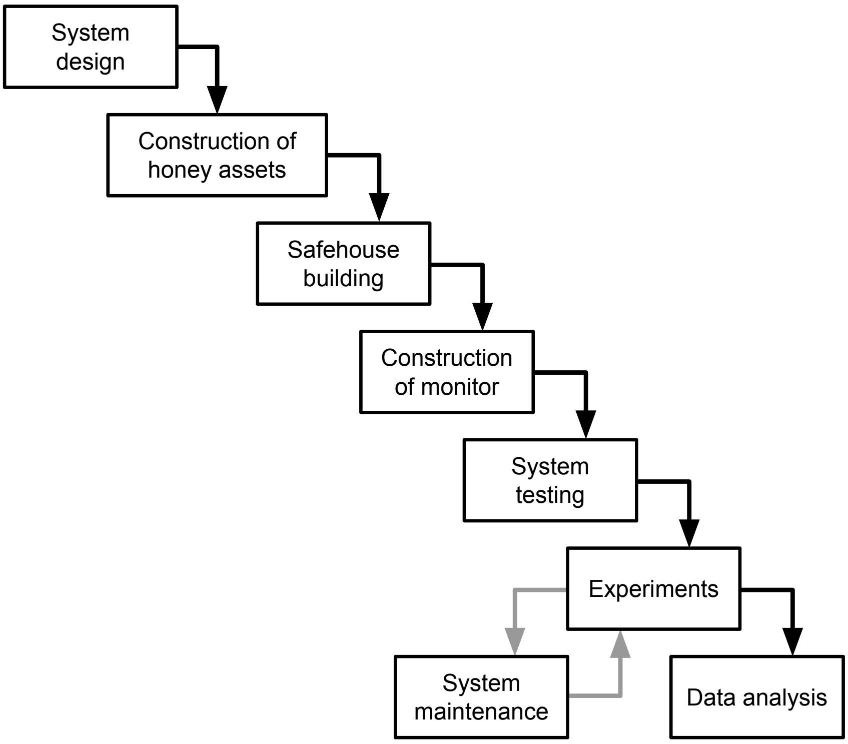


Figure 3.1: Our honeypot development life cycle inspired by the classic systems/software development life cycle (SDLC) [43]. It is important to note that the steps are not necessarily sequential and iteration may be necessary across steps.

3.4 Honeypot development life cycle

In this section, we present our honeypot development life cycle as shown in Figure 3.1, inspired by the classic systems/software development life cycle (SDLC) [43]. Next, we explain all steps of the honeypot development life cycle. It is important to note that those steps are not necessarily sequential and iterations may be necessary across steps. This honeypot development life cycle has been successfully applied to specific honeypot implementations in peer-reviewed work [76, 62, 16].

System design. This is the initial phase of the honeypot development process. Following the ARMER requirements in Section 3.2, the honeypot researcher selects honey asset types and hosting platforms. Other factors to consider include the required scale of experiments, plans towards automation (especially if it is a large-scale honeypot), and deployment platforms and outlets (for instance, where/how to leak honey credentials). This phase, like most phases of the honeypot development life cycle, requires strict adherence to all ARMER requirements. This is also the phase during which the researcher draws up detailed plans for experiments (how to execute them).

Construction of honey assets. This is the phase during which the honeypot researcher builds honey assets that will eventually be leaked to the intended audience, for instance, webmail or social accounts. The researcher will also populate honey assets with realistic data, or in the case of honey credentials, choose realistic credentials.

Safehouse building. Depending on the specific system design, it may be necessary to build an intermediate data store to serve as an anonymous buffer for activity data that will be recorded in honey assets. We call such an intermediate data store a *safehouse*. For example, in Chapters 4 and 6, we created safehouse webmail accounts to serve as buffers during data collection because our honey assets had the capability to send out emails containing details of activity data, and it was necessary to collect those emails via entities that were not obviously connected to us (specifically, safehouse webmail accounts with pseudonymous usernames). Afterwards, we collected activity data from safehouse webmail accounts and processed them offline.

Construction of monitor. This phase involves the construction of “sensors” in honey assets to record activity data, and “virtual telescopes,” which are monitor systems external to honey assets — they connect to honey assets and collect activity data. Alternatively, sensors may send activity data to telescopes (or a safehouse, as explained earlier). In this thesis, our sensors and telescopes often comprise suites of scripts, servers, parsers, and offline data storage. We explain them in detail in the following chapters.

System testing. When honey assets and monitor infrastructure are ready, the honeypot researcher has to test them in a controlled environment to ensure that all components work as planned, and that the entire honeypot pipeline runs as expected.

It might be necessary to make some adjustments during this phase. Note that the honeypot researcher should not leak any live asset yet — this sole purpose of this phase is to test components and the entire system prior to live deployment.

Experiments. After successfully testing the system, the honeypot researcher proceeds to leak honey assets to the intended audience while monitoring all honey assets via the monitor infrastructure. Leaks must be carried out in a convincing manner. For instance, while leaking honey assets, the researcher, within ethical bounds, may carefully mimic known modus operandi of cybercriminals that distribute stolen goods. In this thesis, we explain how we leaked honey assets at the beginning of each experiment. The first asset leak signifies the beginning of experiments.

System maintenance. In the course of experiments, it may be necessary to apply changes to honey assets or monitor infrastructure, or both. For instance, as explained earlier, changes in external platforms that host honey assets may necessitate minor updates in the monitor infrastructure. Similarly, cybercriminals that visit honey assets may change the credentials of those honey assets and it may be necessary to revert those credentials. These steps constitute system maintenance. Depending on the specific honeypot design and implementation, this phase may not always be necessary.

Data analysis. During and after experiments, the researcher analyses the data collected from honey assets and draws inferences. This concludes the honeypot development life cycle.

3.5 Potential alternatives

In this section, we discuss potential alternative approaches to understanding malicious activity in online accounts and justify our use of honeypots instead of these alternatives.

Alternative 1. An alternative to our honeypot approach will be to collaborate with online services (for instance, by working on site with them and deploying honeypots from the inside). This has the advantage of better visibility and ease of access than our approach can offer. It will likely translate into better nuance in research findings. However, the downside is the possible loss of research independence. Also, such experiments will be hard to replicate since they will be based on proprietary data and systems. Finally, non-disclosure agreements (NDAs) may place dire constraints on the dissemination of research findings.

Alternative 2. Another alternative will be to approach law enforcement agencies with a view to interviewing suspects (or convicts) that have been involved in cybercrime. This will elicit information on their methods and activity, for instance, how long they stay in stolen accounts, the content they pay particular attention to, and how they launch and coordinate multi-step attacks. First, gaining such sensitive access to participants will be hard for the average researcher. Second, an obvious problem is that such findings will be based on self-reporting which is prone to overestimation and underestimation [78], and may not be as reliable as collecting and analysing data in the wild. Our honey assets method, despite its limitations, addresses this problem by relying on primary data collected from accounts in the wild.

Alternative 3. It is possible to simulate compromised online accounts in a closed or controlled environment in which crowdsourced participants, say Mechanical Turk workers,[6] will pretend to be criminals and will exhibit "criminal behaviour" during their interactions with the accounts. This option was considered while laying out the author's initial research plan, but was quickly discarded, since the quality of findings will be questionable. The behaviour of participants will differ from real criminal behaviour. Our honey assets method addresses this problem by making criminals believe they are interacting with real accounts. They also do not know that their interactions will be recorded. This leads to realistic interactions.

Hence, we chose the honey assets method over other approaches.[7] Besides, the honey assets method provided an avenue for us to apply systems design and engineering expertise towards bridging an open research gap.

[6] `https://www.mturk.com/`

[7] In other words, we chose the honeypot boulevard over other roads.

3.6 Limitations of the honey assets method

First, as a researcher external to the online service that hosts honey assets, it is difficult to scale up experiments using our honey assets approach because creating many realistic assets takes a lot of time and effort. However, we succeeded in building a large-scale honeypot that comprised social accounts in Chapter 5 because we received some help from collaborators in the online service that hosted our honey assets. In other words, scaling up is not an issue if the service provider is closely involved in creating and operating honeypots.

Second, sourcing realistic data for honey assets is hard. Depending on the specific honeypot implementation, it may be sufficient to use randomly-generated data (for instance, fake financial data in Chapter 6). In other cases, it may be necessary to use datasets generated during real human activity, which may be hard to obtain. After obtaining such datasets, adequate care must be taken to remove all personally identifiable information (PII) from them, prior to use, which is a non-trivial task.

Third, specific honeypot implementations are not platform-agnostic (but our honey assets approach is platform-agnostic). This makes it impossible to reuse already implemented honeypot tools on different platforms. For instance, we had to build three different honeypot implementations for this thesis, instead of building one and reusing it across three platforms. This implies that our honey assets method consumes a lot of time and effort in building the required infrastructure and honey assets.

Despite these limitations, the honey assets method presented in this chapter has successfully produced honeypots on various platforms. This shows that our approach is a viable one for researchers seeking to understand malicious activity in online accounts.

Table 3.1: While exploring related studies, we encountered other system requirements to build honeypot systems.

This thesis	*Bowen et al.* [22]	*Chin et al.* [30]	*Mulliner et al.* [70]	*Vrable et al.* [102]
Accessible	Believable	Scalability	Monitoring	Scalability
Realistic	Enticing	Flexibility	Audit logging	Containment
Measurable	Conspicuous	Attack containment	Containment	
Ethical	Detectable	Stealth	Visibility	
Robust	Variability	Resource management		
	Non-interference	Ease of deployment		
	Differentiable			

3.7 Additional requirements

While exploring related studies, we encountered other system requirements that have been proposed to build honeypot systems. Table 3.1 shows those requirements listed alongside the ones we identified in this thesis (ARMER requirements). Mulliner et al. [70] used the term “challenges” instead of “requirements,” but close observation revealed that those challenges were equivalent to system requirements, hence we included them. Common requirements across most of the studies include *containment*,[8] *measurable*, and *accessible*. Table 3.1 reveals synonyms and exact matches for these requirements across most of the listed studies. *Containment* dominates the table — this shows the importance of ensuring that honey assets are designed and implemented in a way that minimises harm to all parties involved.

[8] Subsumed in the *ethical* requirement mentioned earlier in this chapter.

Chapter 4

Hijacked Webmail Accounts

4.1 Contributions

First, we developed a system to monitor activity in Gmail accounts towards understanding malicious activity in compromised webmail accounts. We publicly released the source code of our system[1] to allow other researchers to deploy their own Gmail accounts for related studies, and add to the understanding that the security community has of malicious activity in online services. To the best of our knowledge, it is the first publicly available Gmail honeypot infrastructure. Second, we deployed 100 honey accounts on Gmail and leaked credentials through various outlets — underground forums, public paste sites, and virtual machines infected with information-stealing malware. Third, we provide detailed measurements of the activity logged by our honey accounts over a period of 7 months. The work in this chapter was originally presented in the 2016 ACM Internet Measurement Conference (IMC'16) by the author of this thesis, and it appeared in IMC'16 conference proceedings. This work has appeared on BBC News,[2] Huffington Post,[3] and The State of Security,[4] among other news outlets. It also emerged as a finalist in the Cyber Security Awareness

[1] https://bitbucket.org/gianluca_students/gmail-honeypot

[2] https://www.bbc.co.uk/news/technology-37510501

[3] https://www.huffingtonpost.co.uk/entry/what-hackers-actually-do-with-your-stolen-personal-information_uk_58049f32e4b0e982146cd18f

[4] https://www.tripwire.com/state-of-security/security-data-protection/heres-what-happens-after-your-webmail-account-is-compromised/

Week Europe 2017 research competition. Via these, we have been able to increase the awareness of the general public about malicious activity in online accounts.

Collaborators. We express our profound appreciation to Enrico Mariconti (UCL PhD student) for his contributions to the design and implementation of the malware honeypot infrastructure in Section 4.4.3 (Enrico and the author built it collaboratively). Enrico also helped to carry out statistical tests in Section 4.5.5. We thank Mark Risher (Google) and Tejaswi Nadahalli (Google) for their support throughout the project.

4.2 Overview

The wealth of information that users store in webmail accounts on services such as Gmail, Yahoo! Mail, or Outlook, as well as the possibility of misusing them for illicit activities, attracts cybercriminals who actively engage in compromising such accounts. They obtain credentials to victims' accounts via phishing [37], infecting users with information-stealing malware [90], or compromising large password databases, leveraging the fact that people often use the same password across multiple services [35]. Stolen credentials and data can be used privately by the cybercriminal or sold in underground markets to other cybercriminals for profit [91].

Cybercriminals use compromised accounts in multiple ways. First, they can use them to send spam [41]. This practice is particularly effective because the established contacts of the account are likely to trust its owner, and are therefore more likely to open the messages that they receive from them [58]. Similarly, the stolen account is likely to have a history of good behaviour with the hosting service, and malicious messages sent from it are less likely to be detected as spam, especially if the recipients are within the same service (e.g., a Gmail account that sends spam to other Gmail accounts) [96]. Alternatively, cybercriminals can use stolen accounts to collect sensitive information about victims. Such information includes financial credentials, login information to other online services, and personal messages of the victim [25].

In general, it is difficult to study the activity of criminals in compromised online accounts without being in control of a large online service, hence there is limited research literature in this space (see Chapter 2 for a detailed coverage of related literature). The rare exceptions are studies that look at information that is publicly observable, such as messages posted on Twitter by compromised accounts [41, 42]. To close this gap, we present a system that is able to monitor the activity of attackers in Gmail accounts.

We set up 100 Gmail accounts and populated them with data to look like webmail accounts that belong to employees of a fictional company. We refer to these accounts as *honey accounts*. To understand how criminals use these accounts after they are compromised, we leaked credentials to the accounts on multiple outlets, modelling different ways by which cybercriminals share and obtain access to stolen credentials, namely public paste sites, underground forums, and information-stealing malware. We then recorded activity in the honey accounts for 7 months.

Our analysis allows us to draw a taxonomy of different actions performed by visitors on stolen Gmail accounts, and provides interesting insights into keywords that visitors typically search for when looking for valuable information in these accounts. We also show that visitors exhibit various skill levels depending on the outlet they source stolen credentials from. Our findings complement what was reported in previous work on manual account hijacking [25], and show that the modus operandi of miscreants varies considerably depending on how they obtain credentials to stolen accounts. In summary, this chapter shines light on what happens within compromised webmail accounts, and will be useful for other researchers and webmail services in the quest for better detection and mitigation systems.

4.3 Background

4.3.1 Gmail accounts

In this chapter, we focus on Gmail accounts with particular attention to the actions performed by cybercriminals when they gain access to a victim's account. We made this choice over other webmail platforms because Gmail allows users to set up scripts to augment the functionality of their accounts. It is therefore an ideal platform for developing webmail-based honeypots. To ease the understanding of the rest of this chapter, we briefly summarise the capabilities offered by webmail accounts in general, and by Gmail in particular.

After authenticating to a Gmail account, a user is presented with a view of their *inbox*. This contains all emails that the user has received and highlights the ones that have not been read yet by displaying them in boldface font. Users have the option to mark emails that are important and need particular attention by *starring* them. Users are also given *search* functionality which allows them to find emails of interest by entering related keywords. They can also organise emails by placing related messages in folders or assigning descriptive labels to them. Such operations can be automated by creating rules to automatically process received emails. When writing emails, content is saved in a *drafts* folder until the user decides to send it. Sent emails can be found in a dedicated folder and they can be also be searched by the user.

4.3.2 Google Apps Script

Google Apps Script is a cloud-based scripting engine that can be used to augment Google Apps and extend their functionality.[5] It is JavaScript-based but runs on Google Cloud, not client endpoints. It is possible to write scripts, otherwise known as lightweight *apps*, to perform specified tasks when a condition is met or an event happens. For instance, a time-driven *trigger* can be fired at a particular time of day or an event-driven trigger fired by a file open event. When a trigger is fired, the JavaScript function associated to that trigger will be executed, for instance, to send an email to a specified address or carry out some computations. A detailed treatment of triggers can be found in Google Apps Script documentation.[6] Our webmail honeypot infrastructure relies on time-driven and event-driven triggers within a

[5] https://developers.google.com/apps-script/overview

[6] https://developers.google.com/apps-script/guides/triggers/

hidden custom app to track and report accesses and changes in webmail accounts to us. It is important to note the resource quotas and execution limits imposed by Google on scripts.[7] These quotas and limits must be considered and factored into design decisions on projects that incorporate Apps Script, because exceeding them will cause scripts to fail.

4.4 Method and experimental setup

In this section, we describe the process of creating and deploying honey accounts. We also present a detailed explanation of how our webmail honeypot system works.

4.4.1 Honey accounts

Our honey accounts are webmail accounts instrumented with Google Apps Script to monitor activity in them in a stealthy manner. The script, hidden in otherwise empty Google spreadsheets, sends notifications to a webmail account under our control (we refer to it as a *safehouse webmail account*), whenever an email is opened, sent, or "starred." In addition, each script sends copies of all draft emails in honey accounts to the safehouse webmail account. We added a *heartbeat message* function to each honey account to send a status notification once a day to the safehouse webmail account, to attest that the account was still functional and had not been blocked by Google. As we mentioned earlier, within each honey account, each script instance was hidden in an otherwise empty and inconspicuous Google spreadsheet, and authorised by the author prior to deployment.[8] This made it unlikely for attackers to find and delete them.

4.4.2 Data collection

In this section, we describe the main components of the webmail honeypot infrastructure that monitors honey accounts, as shown in Figure 4.1.

[7] https://developers.google.com/apps-script/guides/services/quotas

[8] We started all script instances prior to deployment and they continued running during experiments.

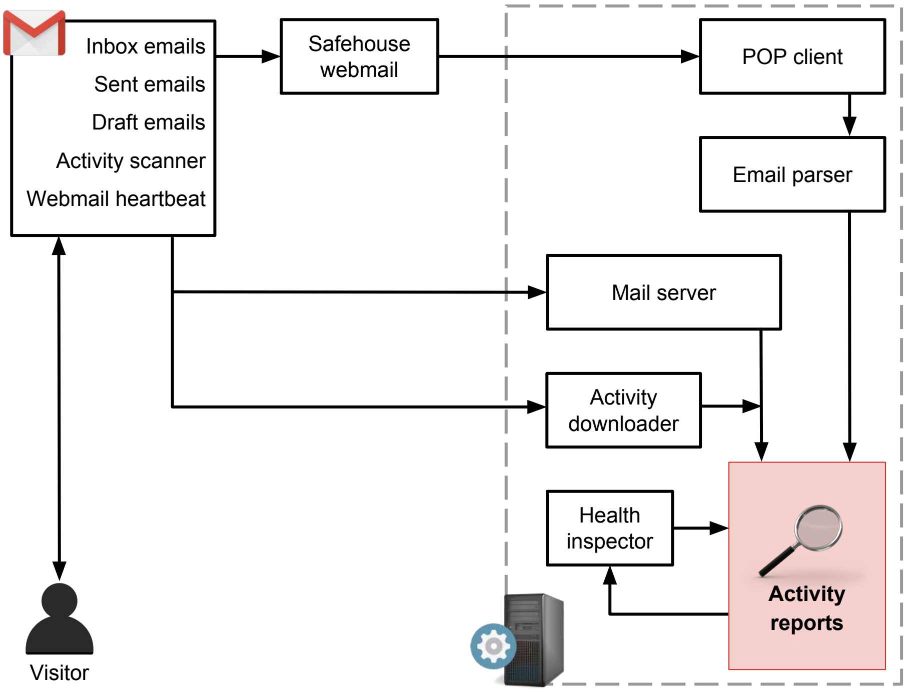


Figure 4.1: Overview of the webmail honeypot infrastructure. Honey accounts send activity records to us via the safehouse webmail account, mail server, and activity downloader, for offline parsing.

Safehouse webmail account. This is a regular webmail account that acts as a safe haven for communications originating from the honey accounts. The script that we hid in honey accounts sends notifications and heartbeat messages to the safehouse webmail account, as described earlier. We periodically retrieve those email notifications via an email client that runs the Post Office Protocol (POP) and parse them offline.

Mail server. One of the main components of our infrastructure comprises a modified mail server. Unlike a regular mail server that forwards email messages, we configured the mail server to receive emails and write them to disk only. It does not forward them to the intended destination. In other words, it works as a *sinkhole mail server*. To minimise abuse, we configured each honey account's default *send-from* address to an email address under our control (it points to our sinkhole mail server). Hence, all emails sent from the honey accounts were delivered to the mail server only, not to the intended destination, and we avoided the problem of spam from honey accounts.

Activity downloader. Google Apps Script is powerful but does not provide all the information required in this chapter. For example, it does not provide location information and IP addresses of visitors to honey accounts. To track those accesses, we set up an activity downloader, an external script that drives a web browser, periodically connects to each honey account, and records information about visitors (cookie identifier, geolocation information, and times of accesses, among others). It navigates to the visitor activity page of each honey account and downloads that information to disk, for offline parsing. By collecting information from visitor activity pages, we obtain location and system configuration information regarding accesses, as provided by Google's geolocation and system configuration fingerprinting system.

Health inspector. To check that honey accounts were up and running, we periodically ran the health inspector on offline activity reports to check for recency (with emphasis on heartbeat messages[9] that were sent by honey accounts daily). Out-

[9] Trivia — Heartbeat messages that arrive at the safehouse webmail account from our honey accounts contain the string "ALIVE" to indicate good health.

of-date heartbeat messages indicate dead accounts (that is, blocked by Google) or hijacked accounts (someone changed their passwords).

We believe that our honey account and monitoring framework unleashes multiple possibilities for researchers who want to carry out further studies on the behaviour of attackers in webmail accounts. For this reason, we released the source code of our system publicly.[10] In addition to the work in this chapter, our webmail honeypot infrastructure was also employed in [16] to study the effects of language differentiation on the activity of cybercriminals in webmail accounts (not included in this thesis). This demonstrates the versatility of our system.

4.4.3 Experiment setup

In line with the honey assets method proposed in Chapter 3, we first set up honey accounts on Gmail and then leaked them through multiple outlets often used by cybercriminals.

Creating honey accounts. We created 100 Gmail accounts and assigned random combinations of popular first and last names to them, similar to the approach in [93]. It is important to note that creating these accounts is a manual process. Google rate-limits the creation of new accounts from the same IP address by presenting a phone verification page after a few accounts have been created. These factors imposed limits on the number of honey accounts we succeeded in creating. We then populated the freshly-created accounts with emails from the public Enron email dataset [61]. This dataset contains emails sent by executives of Enron, an energy corporation, and was publicly released as evidence during Enron's bankruptcy trial. This dataset is suitable for our purposes since the emails in it are typical emails exchanged by corporate users. To make the honey accounts believable and avoid raising suspicion from cybercriminals that connect to them, we mapped distinct recipients in the Enron dataset to our fictional honey profiles (that is, the fictional "owners" of the honey accounts), and replaced first names and last names in the dataset with honey first names and last names. In addition, we changed all instances of

[10] https://bitbucket.org/gianluca_students/gmail-honeypot

“Enron” to a fictional company name that we came up with. In order to have realistic email timestamps, we translated the old Enron email timestamps to recent timestamps slightly earlier than our experiment start date. For instance, given two email timestamps t_1 and t_2 in the Enron dataset such that t_1 is earlier than t_2, we translated them to more recent timestamps T_1 and T_2 such that T_1 corresponds to an earlier time than T_2. We then scheduled those emails to be sent to the recipient honey accounts at times T_1 and T_2 respectively. We sent 200 – 300 emails from the transformed Enron dataset to each honey account while populating them.

Leaking account credentials. To achieve our objectives, we had to entice cybercriminals to interact with the honey accounts. To this end, we selected paste sites and underground forums as appropriate venues for leaking account credentials, since they tend to be misused by cybercriminals for dissemination of stolen credentials. In addition, we leaked some credentials through malware since this is a popular way by which professional cybercriminals steal credentials and compromise accounts [19]. We divided the honey accounts into groups and leaked their credentials in different locations as shown in Table 4.1. We leaked 50 accounts, in total, via the paste sites listed in Table 4.2. For 20 of them, we leaked basic credentials (username and password pairs) on the popular paste sites `pastebin.com` and `pastie.org`. We leaked 10 account credentials on Russian paste websites (`p.for-us.nl` and `paste.org.ru`). For the remaining 20 accounts, we leaked username and password pairs along with UK and US location information of the fictional owners that we associated with the honey accounts. We also included date of birth information for each fake person. We leaked 30 account credentials on underground forums (listed as outlets 5 – 8 in Table 4.2). For 10 of them, we specified only username and password pairs, without additional information. In a manner similar to the paste site leaks described earlier, we appended UK and US location information to underground forum leaks, and claimed that the fictional account owners lived in those locations. We also included date of birth information for each fake person.

For forum leaks, we used the forums listed in Table 4.2. We selected them because they were open for anybody to register and were highly ranked in Google

Table 4.1: Honey account groupings showing the number of account credentials we leaked via each outlet type.

Group	*Accounts*	*Leak outlet*
1	30	paste websites (no location)
2	20	paste websites (with location)
3	10	forums (no location)
4	20	forums (with location)
5	20	malware (no location)

search results. We acknowledge that some underground forums are not open and have a strict vetting policy to let users in [91]. Unfortunately, however, we did not have access to any private forum. The same approach of studying open underground forums has been used in previous work [3]. While leaking credentials on underground forums, we mimicked the modus operandi of cybercriminals that was outlined by Stone-Gross et al. [91]. They showed that cybercriminals often post a sample of their stolen datasets on forums to show that the accounts are real, and promise to provide additional data in exchange for a fee. We recorded the messages that we received on underground forums, mostly enquiries about obtaining the full dataset, but we did not respond to them.

Finally, to study the activity of criminals that obtain credentials through information-stealing malware, we leaked credentials of 20 accounts to information-stealing malware. To this end, we selected malware samples from Zeus family, one of the most popular information-stealing malware families [19], as well as samples from Corebot family. We provide detailed information about our malware honeypot infrastructure (sandbox) in the next section.

The reason for leaking different accounts on different outlets is to study differences in the behaviour of cybercriminals that gain access to stolen credentials through different sources. Similarly, we provide decoy location information in some leaks and not in others, to observe differences in malicious activity depending on the amount and type of information available to cybercriminals. As we show in Section 4.5, accesses to honey accounts were heavily influenced by the presence of additional location information in leaked credentials.

Table 4.2: To lure visitors to the honey accounts, we leaked account credentials through paste sites, underground forums, and information-stealing malware. We chose these paste sites because they allow public pastes, and the forums because joining them does not require introductions or vetting.

Outlet	*Type*	*URL or name*
1	Paste site	`pastebin.com`
2	Paste site	`pastie.org`
3	Paste site	`p.for-us.nl`
4	Paste site	`paste.org.ru`
5	Forum	`offensivecommunity.net`
6	Forum	`bestblackhatforums.eu`
7	Forum	`hackforums.net`
8	Forum	`blackhatworld.com`
9	Malware	Zeus infostealer
10	Malware	Corebot infostealer

Malware honeypot infrastructure. Our malware sandbox system works as follows. A local web server entity manages honey credentials (usernames and passwords) and infomation-stealing malware samples. The host machine creates a Virtual Machine (VM) which contacts the web server to request an executable malware file and a honey credential file. The structure is similar to the one explained in [59]. The malware file is then executed in the VM (that is, the VM infects itself with malware), after which a script drives a browser in the VM to login to Gmail using the previously downloaded credentials. This exposes the honey credentials to malware that is already running in the VM, and leads to credential theft. After some time, the infected VM is deleted and a fresh one is created. This new VM downloads another malware sample and a different honey credential file, and repeats the infection and login operation. To maximise the efficiency of our configuration prior to the experiment, we carried out a test without the Gmail login process, to select only samples whose C&C servers were still up and running.

4.4.4 Threats to validity

We acknowledge that seeding honey accounts with emails from the Enron dataset may introduce bias into our results, and may make the honey accounts less believable to visitors. However, it is necessary to note that the Enron dataset is the only large publicly available email corpus, to the best of our knowledge. To make the emails believable, we changed names, dates, and company name in the emails, using automatic *search-and-replace* string processing techniques. In the future, we will work towards obtaining or generating a better email dataset. Also, some visitors may notice that the honey accounts did not receive any new emails during the period of observation, and this may affect the way visitors interact with the accounts. Another threat is that we only leaked honey credentials through the outlets listed previously (namely paste sites, underground forums, and malware), therefore our results reflect the activity of participants present on those outlets only. Finally, since we selected underground forums that are publicly accessible, our observations may not reflect the modus operandi of actors who are active on closed forums that require vetting to join. Despite these factors, our approach provides valuable insights into what happens in compromised webmail accounts and provides a robust way for other researchers seeking to carry out related experiments.

4.4.5 Ethics

The experiments in this chapter require some ethical considerations. First of all, by granting cybercriminals access to our honey accounts, we incur the risk that those accounts will be used to harm third parties. To minimise this risk, we configured the accounts in a way that all emails would be forwarded to a sinkhole mail server under our control, and never delivered to the outside world. We also established a close collaboration with Google and made sure to report any malicious activity that needed attention to them. Although the suspicious login filters that Google typically uses to protect their accounts from unauthorised accesses were disabled for our honey accounts, all other malicious activity detection algorithms were still in place, and in fact, Google suspended a number of accounts that engaged in suspicious activity. It is important to note, however, that our approach does not rely on help from Google to work. Our main reason for seeking Google's help to disable suspicious login filters was to ensure that all accesses got through to the honey accounts (most accesses would be blocked if Google did not disable the login filters). This does not directly affect our methodology, and as a result does not reduce the wider applicability of our approach. It is also important to note that Google did not share with us any details on the techniques used internally for the detection of malicious activity on Gmail. Another point of risk was ensuring that the malware in our VMs did not harm third parties. To mitigate this risk, we followed common practices [82] such as restricting the bandwidth available to our virtual machines and sinkholing all email traffic sent by them. Finally, our experiments involved deceiving cybercriminals by providing them fake accounts that contained fake personal information. To ensure that our experiments were run in an ethical fashion, we obtained ethics approval from UCL beforehand.

4.5 Data Analysis

We monitored activity in the honey accounts for 7 months, from 25th June, 2015 to 16th February, 2016. In this section, we first provide an overview of our findings and then discuss a taxonomy of the types of activity that we observed. We focus on the differences in modus operandi shown by cybercriminals who obtain credentials to accounts from various outlets. We then investigate if cybercriminals attempt to evade location-based detection systems by connecting from locations that are closer to the places that account owners typically connect from. We also develop a metric to infer keywords that attackers search for when looking for interesting information in an email account. Finally, we analyse how certain types of cybercriminals appear to be stealthier and more advanced than others.

Cookies and accesses. Google records each unique access to a Gmail account and labels that access with a unique cookie identifier. These unique cookie identifiers, along with more information including times of accesses, are included in visitor activity pages of Gmail accounts. Our scripts (previously described in Section 4.4) extract this data. For the sake of convenience, we will use the terms “cookie” and “unique access” interchangeably in the remainder of this chapter.

4.5.1 Activity overview

We created, instrumented, and leaked 100 Gmail accounts for our experiments. To avoid biasing the results, we removed all accesses made to honey accounts by IP addresses from our honeypot infrastructure. We also removed all accesses that originated from London (UK) where our monitoring infrastructure was located. After this filtering operation, we observed 326 unique accesses to the accounts, during which 147 emails were opened, 845 emails were sent, and 12 unique draft emails were composed by visitors. In total, 90 accounts received accesses, comprising 41 accounts leaked to paste sites, 30 accounts leaked to underground forums, and 19 accounts leaked through malware. 42 accounts were blocked by Google during experiments because of suspicious activity. We were able to record activity in those accounts for some time before Google blocked them. 36 accounts were hijacked by visitors, that is, the passwords of such accounts were changed by visitors. As a result, we lost control of those accounts. We did not observe any attempt by attackers to change the default *send-from* addresses of our honey accounts. However, assuming that happened and attackers started sending spam messages, Google would block such accounts since we asked them to monitor the accounts with particular attention. A dataset containing parsed metadata of accesses to honey accounts is publicly available.[11]

4.5.2 Taxonomy of account accesses

From the activity observed in honey accounts, we devised a taxonomy of attackers/visitors based on unique accesses to the accounts. We identified four types of visitors (described next).

Curious. These accesses constitute the most basic type of access to stolen accounts. After getting hold of account credentials, people connect to those accounts to check if the credentials truly work. Afterwards, they do not carry out any additional action. The majority of observed accesses belong to this category, accounting for

[11] http://dx.doi.org/10.14324/000.ds.1508297

224 accesses. We acknowledge that this large number of curious accesses may be due in part to experienced attackers avoiding interactions with the accounts after logging in, probably after careful observations indicating that the accounts do not look entirely real. This could potentially introduce some bias into our results.

Gold diggers. When connecting to a stolen account, attackers often want to understand its worth [25]. For this reason, after logging into honey accounts, some attackers search for sensitive information such as other login credentials and financial attachments. They also seek information that may be useful in spearphishing attacks. We call these accesses "gold diggers." Previous research has shown that this practice is quite common for manual account hijackers [25]. In this paper, we confirm that finding, provide a methodology to assess the keywords that visitors search for, and analyse differences in modus operandi of gold digger accesses for credentials leaked through various outlets. In total, we observed 82 accesses of this type.

Spammers. One of the main capabilities of webmail accounts is email sending. Previous research has shown that large spamming botnets have code in their bots and C&C infrastructure to take advantage of this capability, by having the bots connect directly to compromised accounts and send spam [91]. Accesses belong to this category if they send any email. We observed 8 accounts that recorded such accesses. This low number of accounts shows that sending spam is not one of the main purposes that cybercriminals use compromised accounts for, when stolen through the outlets that we studied.

Hijackers. A stealthy criminal is likely to keep a low profile when accessing a stolen account to avoid raising suspicion. Less stealthy miscreants, however, might lock the legitimate owner out of their account by changing their password. We call these accesses "hijackers." In total, we observed 36 accesses of this type. A password change prevents us from reaching the account's visitor activity page, therefore we are unable to collect information about accesses to the account afterwards.

It is important to note that the taxonomy classes that we described are not exclu-

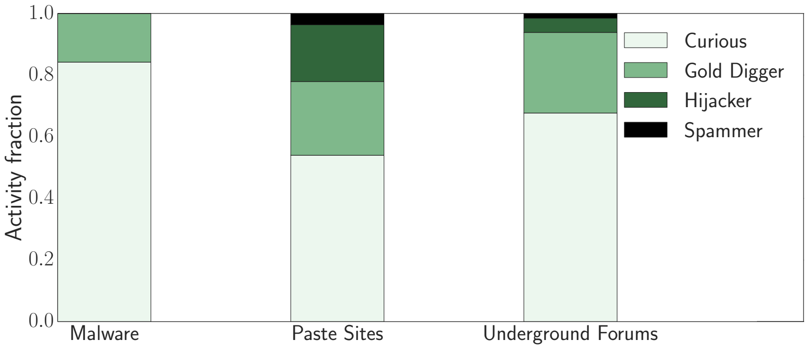


Figure 4.2: Distribution of types of accesses for various leak outlets. Most accesses belong to the “curious” category. It is possible to spot differences in types of activities for different leak outlets. For example, accounts leaked via malware do not present activity of the “hijacker” type. On the other hand, hijackers are particularly common among miscreants who obtain stolen credentials through paste sites.

sive. For example, an attacker might use an account to send spam emails (“spammer” category) and then change the password of that account (“hijacker” category). Such overlaps occurred often in the accesses recorded in our honey accounts. It is interesting to note that there was no access that behaved exclusively as “spammer.” Miscreants that sent spam through our honey accounts also acted as “hijackers” or “gold diggers.”

We set out to understand the distribution of different types of accesses in accounts that were leaked through various outlets. Figure 4.2 shows a breakdown of this distribution. Visitors who gain access to stolen accounts through malware are the stealthiest and never lock the legitimate owners out of their accounts. Instead, they limit their activity to checking if the credentials are real or searching for sensitive information in the accounts, possibly in an attempt to estimate the value of the accounts. Accounts leaked through paste sites and underground forums revealed the presence of hijackers. 20% of the accesses to accounts leaked through paste sites, in particular, belong to this category. Accounts leaked through underground forums, on the other hand, recorded the highest percentage of gold digger accesses (about 30% of all accesses).

4.5.3 Timing of activity

Here we provide a detailed analysis of unique accesses that were recorded in the honey accounts, with emphasis on their timing.

Duration of accesses. For each cookie identifier, we recorded the time that the cookie first appeared in a particular honey account as t_0 and the last time it appeared as t_{last}. From this information, we computed the duration of activity of each cookie as $t_{last} - t_0$. Note that t_{last} of each cookie is a lower bound since we cease to obtain information about cookies if the password of the honey account that is recording cookies is changed, for instance. Figure 4.3 shows Cumulative Distribution Functions (CDFs) of the duration of unique accesses of different types of visitors. The vast majority of accesses are very short, lasting only a few minutes and never coming back. Spammer accesses, in particular, tend to send emails in bursts for a certain period and then disconnect. Hijacker and gold digger accesses, on the other hand, have a long tail of about 10% accesses that keep coming back for several days in a row. The CDFs show that most curious accesses are repeated over many days, indicating that the visitors keep coming back to find out if there is new information in the accounts. This stands in conflict with the finding in [25] which states that most cybercriminals connect to a compromised webmail account once, to assess its value within a few minutes. However, [25] focused on accounts compromised via phishing pages, while we look at a broader range of ways through which criminals can obtain stolen credentials. Our results show that the modes of operation of cybercriminals vary depending on the outlets they obtain stolen credentials from.

Time between leak and first access. Next, we studied how long it takes from the time that credentials are leaked via different outlets until our infrastructure records accesses from visitors. Figure 4.4 shows CDFs of the time between leak and first access for accounts leaked through different outlets. Within the first 25 days after leak, we recorded 80% of all unique accesses to accounts leaked to paste sites, 60% of all unique accesses to accounts leaked to underground forums, and 40% of

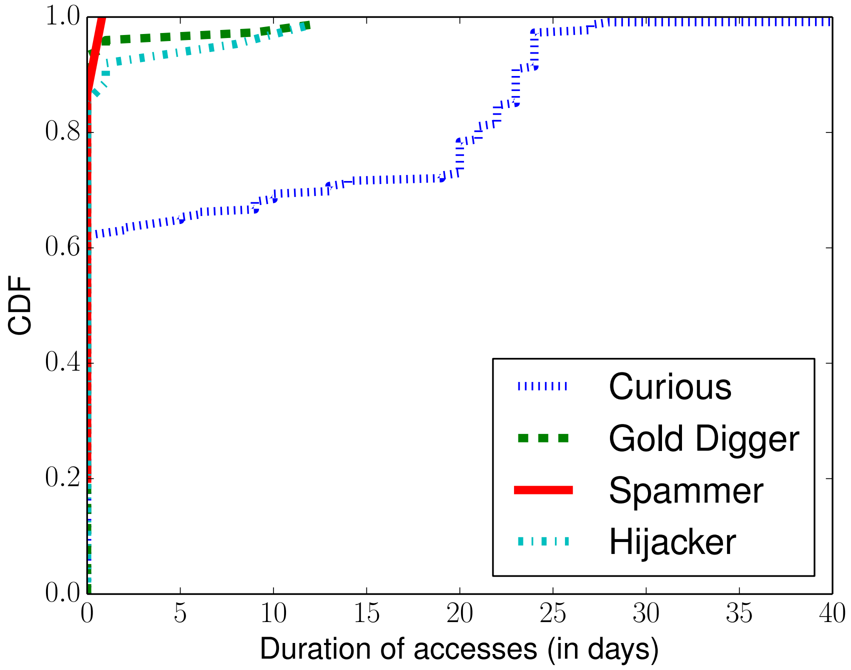


Figure 4.3: CDFs of duration of unique accesses per activity type in our honey accounts. The vast majority of unique accesses last a few minutes. Spammers tend to use accounts aggressively for a short time and then disconnect. The other types of accesses, and in particular curious ones, come back after some time, possibly to check for new activity in the honey accounts.

all unique accesses to accounts leaked via malware. A particularly interesting observation is the nature of unique accesses to accounts leaked via malware. A close look at Figure 4.4 reveals rapid increases in unique accesses to honey accounts leaked to malware, about 30 days after the leak, and also after 100 days (indicated by two sharp inflection points).

Figure 4.5 sheds more light into what happened at those inflection points. It reports the unique accesses to each honey account over time. Note that accounts that were leaked on public outlets such as forums and paste sites can be accessed by multiple visitors at the same time. Account credentials leaked through malware, on the other hand, are available only to the botmaster that stole them, until they decide to sell them or give them to someone else. Seeing bursts, in accesses to accounts leaked through malware, months after the actual leak happened indicates that the accounts were visited again by the same criminal who operated the malware infrastructure, or that the accounts were sold on an underground market and that another miscreant is now using them. This hypothesis is somewhat confirmed by the fact that these bursts in accesses were the gold digger type (we checked), while all previous accesses to the same accounts were of the curious type. In addition, Figure 4.5 shows that the majority of accounts leaked to paste sites were accessed within a few days of leak, while a particular subset was not accessed for more than two months. That subset refers to the ten credentials we leaked to Russian paste sites. Those honey accounts were not accessed for more than two months from the time of leak. This either indicates that cybercriminals are not many on Russian paste sites or they did not believe that the accounts were real.

4.5.4 System configuration of accesses

We observed a wide variety of devices and browsers in accesses to leaked accounts by leveraging Google's system fingerprinting information (available to us inside honey accounts).

Browsers. As shown in Figure 4.6, accesses to accounts leaked on paste sites

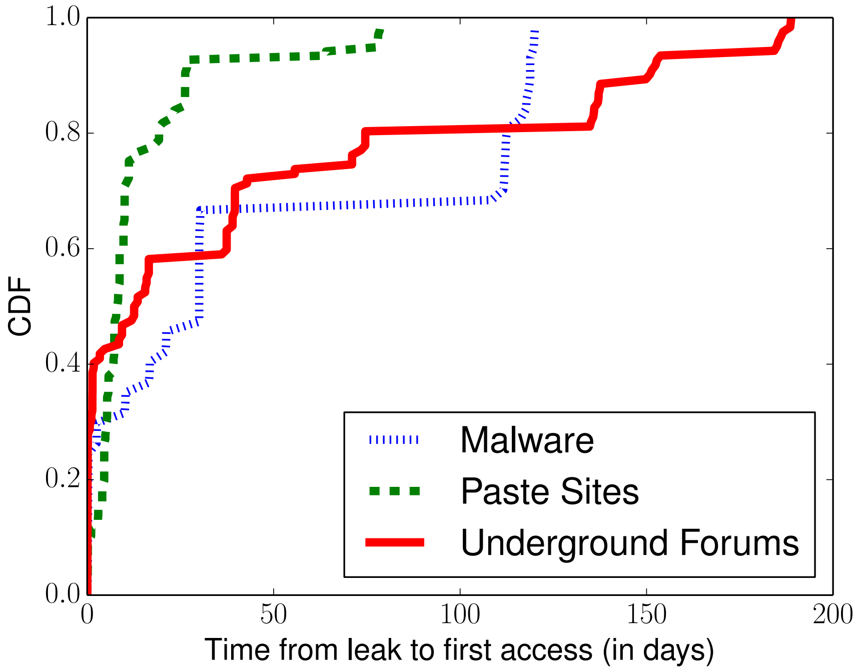


Figure 4.4: CDFs of the time between first credential leak and first visit. Accounts leaked through paste sites received accesses earlier than accounts leaked through other outlets.

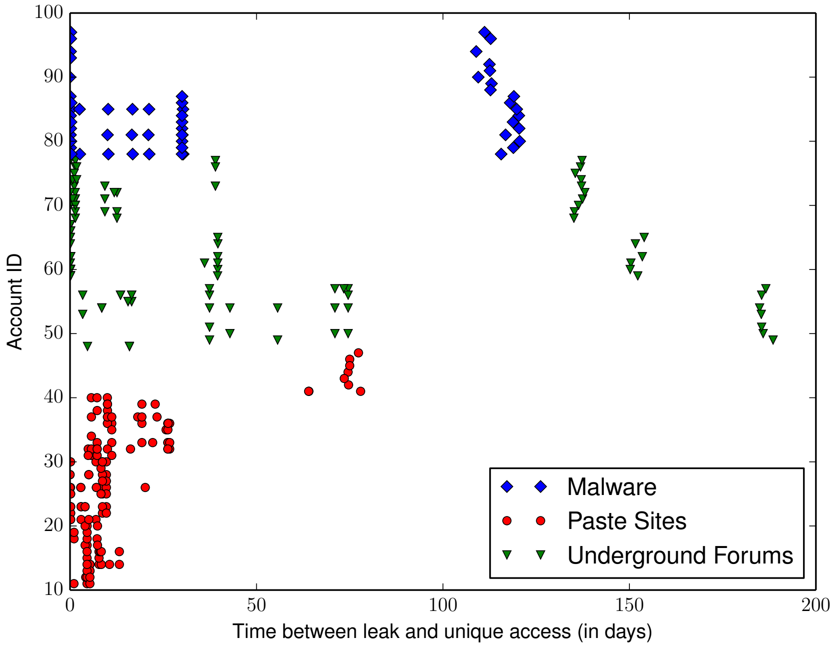


Figure 4.5: Duration between time of leak and unique accesses in accounts leaked through various outlets. Accounts leaked via malware experienced a sudden increase in unique accesses after 30 days and 100 days from the leak, indicating that they had been sold or transferred to another party by cybercriminals behind the malware C&C infrastructure.

were made through a variety of popular browsers, with Firefox and Chrome taking the lead. We also recorded many accesses from unknown browsers. It is possible for an attacker to hide browser information from Google servers by presenting an empty user agent and hiding other fingerprintable information [75]. About 50% of accesses to accounts leaked through paste sites were not identifiable. Chrome and Firefox take the lead in groups leaked in underground forums as well, but there is less variety of browsers there. Interestingly, all accesses to accounts in malware groups were made from unknown browsers. This shows that cybercriminals that accessed accounts leaked through malware were stealthier than others.

Operating systems. While analysing the operating systems on devices used by visitors, we observed that honey accounts leaked through malware mostly received accesses from Windows computers, followed by Mac OS X and Linux. This is shown in Figure 4.7. In the paste sites and underground forum groups, we observed a wider range of operating systems. More than 50% of computers in the three categories ran on Windows. It is interesting to note that Android devices were also used to connect to honey accounts in paste site and underground forum groups.

The diversity of devices and browsers in paste site and underground forum groups indicates a motley mix of cybercriminals with various motives and capabilities, compared to the malware groups that are more homogeneous. It is also obvious that attackers that steal credentials through malware make more effort to cover their tracks by evading browser fingerprinting.

4.5.5 On the origins of accesses

We recorded origin locations in accesses that were logged by our infrastructure. Our goal was to understand patterns in the locations of criminals. Out of the 326 recorded accesses, 132 came from TOR exit nodes. More specifically, 28 accesses to accounts leaked on paste sites were made via TOR, out of a total of 144 accesses to accounts leaked on paste sites. 48 accesses to accounts leaked on forums were made through TOR, out of a total of 125 accesses made to accounts leaked on

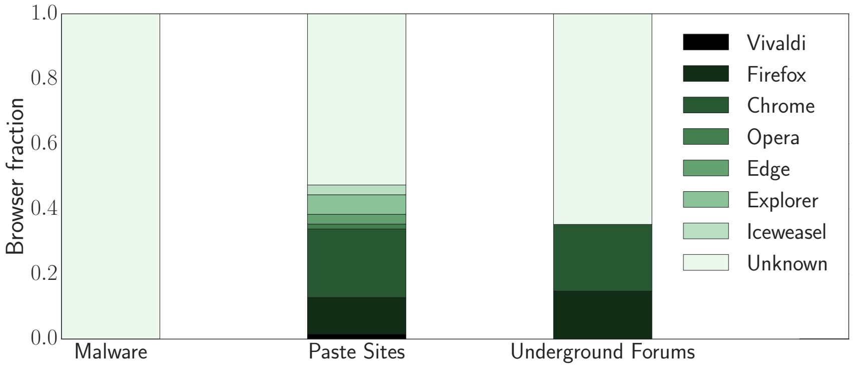


Figure 4.6: Browsers used during accesses to honey accounts.

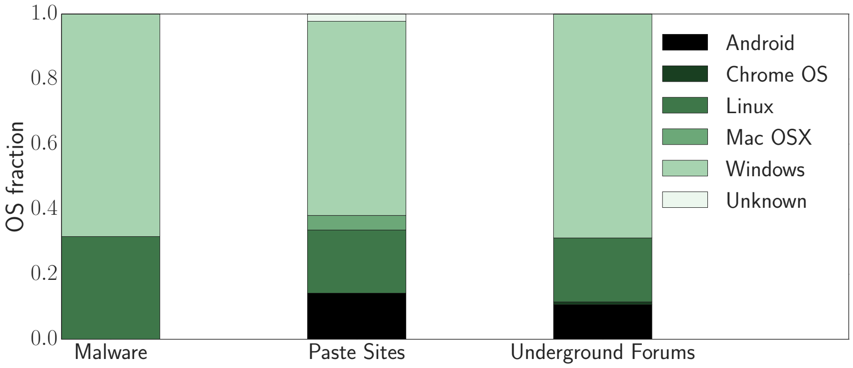


Figure 4.7: Operating systems used during accesses to honey accounts.

forums. We observed 57 accesses to accounts leaked through malware and all except one of those accesses were made via TOR. We removed these accesses from the location analysis in this section — they do not provide information on the physical location of the visitors.[12] After removing TOR exit nodes, 173 unique accesses presented location information. To determine this location information, we used the geolocation information provided by Google via visitor activity pages of honey accounts. We observed accesses from a total of 29 countries. To understand whether the IP addresses that connected to our honey accounts had been recorded in previous malicious activity, we ran checks against Spamhaus blacklist.[13] We found 20 IP addresses that accessed our honey accounts in the Spamhaus blacklist. Because of the nature of this blacklist, we believe that the addresses belong to malware-infected machines that were used by cybercriminals to connect to the stolen accounts.

One of our goals was to observe if cybercriminals would attempt to evade location-based login risk analysis systems by tweaking access origins. In particular, we wanted to assess if telling criminals the location of an account owner will influence the location that they will then connect from. Despite observing 57 accesses to honey accounts leaked through malware, we discovered that all these connections, except one, originated from TOR exit nodes. This shows that malware operators that accessed our accounts preferred to hide their location through the use of anonymising systems rather than modifying their access location based on knowledge of the usual login location of the account owner (or both).

While leaking honey credentials, we chose London and Pontiac as our decoy UK and US locations respectively. In other words, during leaks, we claimed that some honey accounts belonged to fictional persons living in either London or Pontiac. However, we realised that leaking multiple accounts with the same location might raise suspicion. Hence, we chose various UK and US locations such that London and Pontiac were the midpoints of those locations.

To observe the impact of knowledge of login location on the locations that cy-

[12] Strangely, TOR entries in visitor activity pages of honey accounts provided neither location information nor IP addresses.

[13] https://www.spamhaus.org/

bercriminals connect from, we calculated the median values of distances of the locations recorded in unique accesses from the midpoints of the advertised decoy locations in our account leaks. For accesses A to honey accounts leaked on paste sites, advertised with UK information, we extracted location information, translated them to geographical coordinates L_A, and computed the $dist\_paste\_UK$ vector as $distance(L_A, mid_{UK})$, where mid_{UK} are London's coordinates. Distances were measured in kilometres. We extracted the median values of all distance vectors and plotted concentric circles on UK and US maps, by specifying those median distances as radii of the circles, as shown in Figures 4.8 and 4.9.

Interestingly, we observe that connections to accounts with advertised locations originate from places closer to our midpoints than accounts with leaked information containing usernames and passwords only. Figure 4.8 shows that connections to accounts leaked on paste sites and forums result in smaller median circles, that is, the connections originate from locations closer to London, the UK midpoint. The smallest circle is for the accounts leaked on paste sites, with advertised UK location information (radius 1400 kilometres). In contrast, the circle of accounts leaked on paste sites without location information has a radius of 1784 kilometres. The median circle of accounts leaked in underground forums, with no advertised location information, is the largest circle in Figure 4.8, while the one of accounts leaked in underground forums, along with UK location information, is smaller.

We obtained similar results in the US plot, with some interesting distinctions. As shown in Figure 4.9, connections to honey accounts leaked on paste sites, with advertised US locations, are clustered around the US midpoint, as indicated by the circle with a radius of 939 kilometres, compared to the median circle of accounts leaked on paste sites without location information, which has a radius of 7900 kilometres. However, despite the fact that the median circle of accounts leaked in underground forums with advertised locations is smaller than that of the one without advertised location information, the difference in their radii is not as pronounced. This again supports the indication that cybercriminals on paste sites exhibit more *location malleability*, that is, they cloak their origins of accesses to appear closer

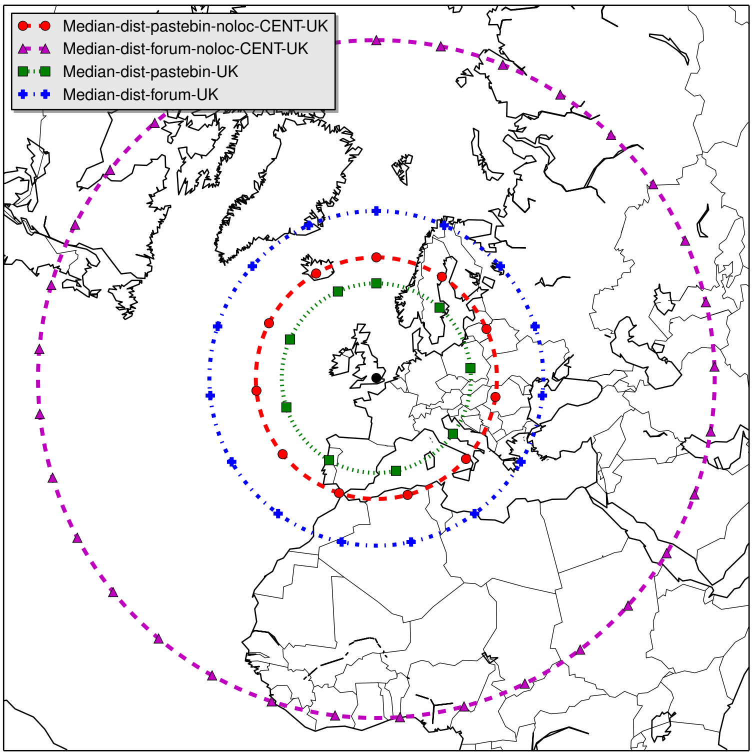

Median-dist-pastebin-noloc-CENT-UK
Median-dist-forum-noloc-CENT-UK
Median-dist-pastebin-UK
Median-dist-forum-UK

Figure 4.8: Distance of login locations from London, UK (advertised during credential leaks). Red lines indicate credentials leaked on paste sites with no location information, green lines indicate credentials leaked on paste sites with location information, purple lines indicate credentials leaked on underground forums without location information, while blue lines indicate credentials leaked on underground forums with location information.

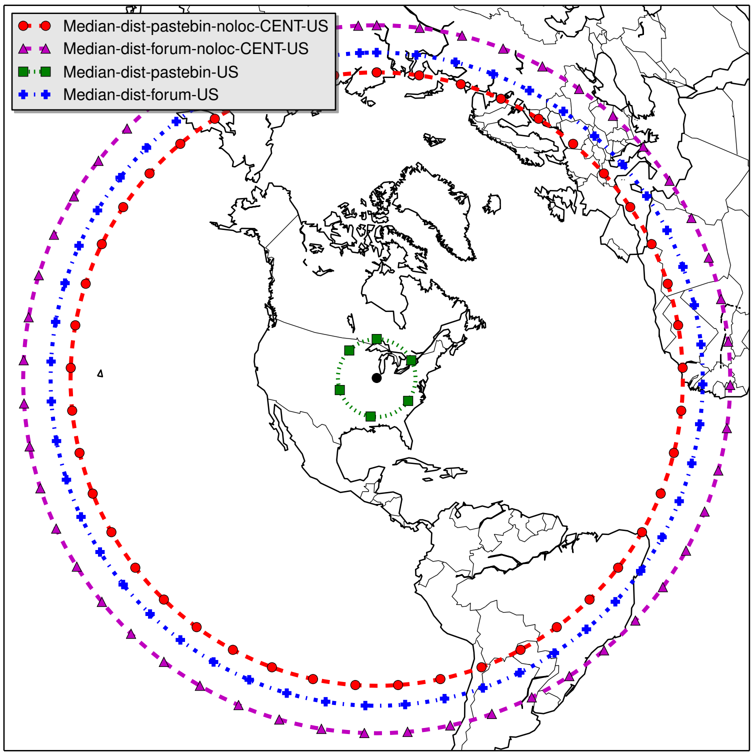

Median-dist-pastebin-noloc-CENT-US
Median-dist-forum-noloc-CENT-US
Median-dist-pastebin-US
Median-dist-forum-US

Figure 4.9: Distance of login locations from Pontiac, MI (advertised during credential leaks). Red lines indicate credentials leaked on paste sites with no location information, green lines indicate credentials leaked on paste sites with location information, purple lines indicate credentials leaked on underground forums without location information, while blue lines indicate credentials leaked on underground forums with location information.

to the account owner's location if they know it. It also shows that cybercriminals on underground forums are less sophisticated or care less than the ones on paste sites.

Statistical tests. As explained previously, Figures 4.8 and 4.9 show that accesses to leaked accounts happen closer to owners' locations if such location information is included in the leak. To confirm the statistical significance of this finding, we performed a Cramer Von Mises test [33]. The Anderson version [8] of this test can be used to understand if two vectors likely belong to the same statistical distribution or not. The p-value has to be under 0.01 for us state that it is possible to reject the null hypothesis,[14] otherwise it is not possible to state with statistical significance that both distance vectors belong to different distributions. The result of the test on paste sites vectors (p-values of 0.0017415 for UK location information *versus* no known location and 0.0000007 for US location information *versus* no known location) allows us to reject the null hypothesis, thus we state conclusively that the two vectors belong to different distributions, while we cannot say the same for tests on forum vectors (p-values of 0.272883 in the UK case and 0.272011 in the US one). Therefore, we conclusively state that criminals that use paste sites connect from closer locations when location information is provided along with leaked credentials. We cannot reach that conclusion in the case of accounts leaked to underground forums although Figures 4.8 and 4.9 indicate some location effects as well.

4.5.6 The gold digger's quest

Cybercriminals compromise online accounts because of the inherent value of target accounts. Hence, they assess accounts to decide their value and what to do with them. We decided to study the words that they likely searched for within the honey accounts, in order to understand and potentially characterise anomalous searches in the accounts. A limiting factor was that we did not have access to search logs of the honey accounts, but only to the content of the emails that were opened by visitors. To overcome this limitation, we employed Term Frequency–Inverse Document Fre-

[14] Null hypothesis — Both vectors of distances belong to the same distribution.

quency (TF-IDF). TF-IDF can be used to rank words in a corpus by importance. We relied on TF-IDF to infer the words that visitors searched for in the honey accounts. TF-IDF is a product of two metrics, namely Term Frequency (TF) and Inverse Document Frequency (IDF). It allows us to infer the words that visitors searched for, by comparing important words in the emails opened by visitors to important words in all emails within the honey accounts.

In its simplest form, TF is a measure of how frequently term t occurs in document d, as shown in Equation 4.1. IDF is a logarithmic scaling metric of the fraction of documents containing term t, as shown in Equation 4.2, where D is the set of all documents in the corpus, N is the total number of documents in the corpus, $|d \in D : t \in d|$ is the number of documents in D that contain term t. Once TF and IDF are known, TF-IDF can be computed by multiplying TF and IDF, as shown in Equation 4.3.

$$tf(t,d) = f_{t,d} \tag{4.1}$$

$$idf(t,D) = log\frac{N}{|d \in D : t \in d|} \tag{4.2}$$

$$tfidf(t,d,D) = tf(t,d) \times idf(t,D) \tag{4.3}$$

The output of TF-IDF is a weighted metric that ranges between 0 and 1. The closer the weighted value is to 1, the more important the term is. We evaluated TF-IDF on a text corpus comprising two documents, that is, all emails d_A in the honey accounts and all emails d_R opened by visitors. The intuition is that words that have high importance in the emails that have been opened by a visitor, but have lower importance in the overall dataset, are likely to be keywords that visitors searched for in the Gmail account. We preprocessed the corpus by filtering out all words that have less than 5 characters and removing all known header-related words, for instance “delivered” and “charset,” honey email handles, and also removing signalling

Searched words	$TFIDF_R$	$TFIDF_A$	$TFIDF_R - TFIDF_A$
results	0.2250	0.0127	0.2122
bitcoin	0.1904	0.0	0.1904
family	0.1624	0.0200	0.1423
seller	0.1333	0.0037	0.1296
localbitcoins	0.1009	0.0	0.1009
account	0.1114	0.0247	0.0866
payment	0.0982	0.0157	0.0824
bitcoins	0.0768	0.0	0.0768
below	0.1236	0.0496	0.0740
listed	0.0858	0.0207	0.0651

Common words	$TFIDF_R$	$TFIDF_A$	$TFIDF_R - TFIDF_A$
transfer	0.2795	0.2949	-0.0154
please	0.2116	0.2608	-0.0493
original	0.1387	0.1540	-0.0154
company	0.0420	0.1531	-0.1111
would	0.0864	0.1493	-0.0630
energy	0.0618	0.1471	-0.0853
information	0.0985	0.1308	-0.0323
about	0.1342	0.1226	0.0116
email	0.1402	0.1196	0.0207
power	0.0462	0.1175	-0.0713

Table 4.3: Top 10 words sorted by $TFIDF_R - TFIDF_A$ (upper part) and top 10 words sorted by $TFIDF_A$ (lower part). The words in the upper part are the ones that have the highest difference in importance between the emails opened by visitors and emails in the entire corpus. Hence, they are the words that visitors likely searched for while looking for sensitive information in the stolen accounts. The words in the lower part, on the other hand, are the ones that have the highest importance in the entire corpus.

information that our monitoring infrastructure introduced into the emails. After running TF-IDF on the remaining terms in the corpus, we obtained their TF-IDF values as vectors $TFIDF_A$ and $TFIDF_R$, the TF-IDF values of all terms in the corpus $[d_A, d_R]$. We proceeded to compute their difference as $TFIDF_R - TFIDF_A$. The top 10 words by $TFIDF_R - TFIDF_A$ compared to the top 10 words by $TFIDF_A$ are presented in Table 4.3. Words that have $TFIDF_R$ values that are higher than $TFIDF_A$ values will rank higher in the list, and those are the words that visitors likely searched for.

As seen in Table 4.3, the top 10 important words by $TFIDF_R - TFIDF_A$ are sensitive words, such as “Bitcoin,” “family,” and “payment.” Comparing these words with the most important words in the entire corpus reveals that visitors likely searched for sensitive information, especially financial information. In addition, words with high importance in the entire corpus (for example, “company” and “energy”), shown in the lower part of Table 4.3, have much lower importance in the emails opened by visitors, and most of them have negative $TFIDF_R - TFIDF_A$ values. This is a strong indication that the emails opened in honey accounts were not opened at random, but were the result of searches for sensitive information.

Originally, the Enron dataset had no “Bitcoin” term. That term was introduced into the opened emails document d_R through the actions of one of the criminals that accessed some honey accounts. The criminal attempted to send blackmail messages from some honey accounts to victims of the Ashley Madison dating website scandal,[15] requesting ransoms in Bitcoin in exchange for silence. In the process, many draft emails containing Bitcoin information were created and abandoned by the criminal, and other visitors opened them during later accesses. Hence, our honeypot infrastructure picked up Bitcoin-related terms and they rank high in Table 4.3 (the upper part), showing that visitors indicated a lot of interest in those emails.

4.5.7 Sophistication of attackers

From the accesses recorded in honey accounts, we identified three peculiar behaviours of cybercriminals that indicate their level of sophistication: *configuration hiding* — for instance by hiding user agent information, *location filter evasion* — by connecting from locations close to the account owner’s location if known, and *stealth* — avoiding clearly malicious actions such as hijacking and spamming. Attackers accessing honey accounts leaked via different outlets exhibit different types of sophistication. Those accessing accounts leaked through malware are stealthier than others — they do not hijack the accounts and they do not send spam from them. They also access the accounts via TOR network and hide their system configuration, for instance, their web browsers are not fingerprintable by Google. Attackers

[15] https://www.wired.com/2015/08/happened-hackers-posted-stolen-ashley-madison-data/

accessing accounts leaked on paste sites tend to connect from locations closer to the account owners' locations if known. They do so to evade detection. Attackers accessing accounts leaked in underground forums do not make significant attempts to stay stealthy or to connect from closer locations. These differences in sophistication can be used to characterise attacker behaviour in future work.

4.6 Interesting case studies

In this section, we present some interesting case studies that we encountered during experiments. They help to shed more light on the actions of cybercriminals on compromised webmail accounts.

First, attempts were made to send multiple blackmail messages to victims of the Ashley Madison dating website scandal[16] from three honey accounts. In the emails, which were not delivered[17] to the intended recipients, the blackmailer threatened to expose victims unless they made some payments in Bitcoin to a specified Bitcoin wallet. Tutorials on how to make Bitcoin payments were also included in the messages. The blackmailer created and abandoned many drafts emails targeted at more Ashley Madison scandal victims. Second, two honey accounts received notification emails about the hidden Google Apps Script "using too much computer time." The notifications were opened by a visitor and we received notifications about the opening of those "computer time" notifications. Finally, an attacker registered on a carding forum using one of the honey accounts as registration email address. As a result, registration confirmation information was sent to the honey account. This shows that some of the accounts were used as stepping stones by cybercriminals to perform further illicit activity.

[16] https://www.wired.com/2015/08/happened-hackers-posted-stolen-ashley-madison-data/

[17] Recall that we set up a mail sinkhole mechanism to trap outgoing emails in Section 4.4.

4.7 Summary

In this chapter, we presented our honeypot system that can monitor the activity of cybercriminals who gain illicit access to Gmail accounts. Our system is publicly available to encourage researchers to set up additional experiments and improve the knowledge of our community regarding what happens to stolen webmail accounts.[18] We set up and ran experiments involving 100 honey accounts, leaked them via paste sites, underground forums, and virtual machines infected with malware, and provided detailed analyses of the activity of cybercriminals and other visitors to the accounts. Our findings will help the research community to gain better understanding of the ecosystem of stolen online accounts, and potentially help researchers and online services to develop better detection and mitigation systems to make online accounts safer for everyone.

[18] https://bitbucket.org/gianluca_students/gmail-honeypot

Chapter 5

Stolen Social Accounts

5.1 Contributions

First, we devised a method to instrument and monitor compromised social network accounts, following the general honey assets approach proposed earlier in Chapter 3. Second, we created, instrumented, and deployed more than 1000 Facebook accounts in our experimental setup, incorporating age range and gender variations in the accounts, to observe resulting differences in accesses. Third, we present detailed measurements and analyses of accesses and actions performed by visitors in Facebook accounts, and shed light on what happens in stolen social accounts. To the best of our knowledge, this is the first large-scale Facebook honeypot to that effect. Our work in this chapter has won a “Secure the Internet” grant from Facebook.[1]

Collaborators. We express our heartfelt gratitude to Nektarios Leontiadis, Despoina Magka, and Mark Atherton (all in Facebook Inc.), henceforth referred to as our Facebook contacts, for helping us to scale up experiments, especially during the process of creating Facebook accounts. They also helped to establish friend connections among the accounts. It is important to note that our Facebook contacts did not share any proprietary data or methods with us before, during, or after experiments.

[1] https://research.fb.com/facebook-awards-more-than-800000-in-secure-the-internet-grants/

5.2 Overview

Social accounts are almost indispensable in our daily lives. Discovering old and new friends on Facebook, curating news on Twitter, and securing the next job on LinkedIn are a few of the many activities that social accounts facilitate. It goes without saying that individuals, businesses, and other entities find social accounts useful for personal and commercial purposes. Like other types of online accounts, social accounts accumulate personal information, sentimental value, and sometimes, financial value, over time. Compared to webmail accounts (studied in Chapter 4), social accounts provide features that transcend messaging.

How much latent value exists in a social account? Honan, staff at Wired Magazine, learned the answer to that question in a terrifying way. In 2012, he was the victim of a chain of attacks by hackers that sought to take over his Twitter account. His Google and Apple ID accounts were also stolen during the attacks in which he lost a lot of data.[2] This clearly highlights the value of social accounts. It also emphasises the importance of understanding what attracts cybercriminals to social accounts and what they do within the accounts after breaking in. This knowledge will help social network service providers to develop better detection and mitigation systems.

Other problems plaguing social network platforms and their users include the proliferation of misinformation and disinformation (also known as *fake news*) [112, 111, 113], fake accounts (also known as *Sybils*) [109], and hate speech and cyberbullying [50], among others. However, we do not study those problems in this chapter. Instead, we focus on the problem of data breaches, specifically, credential theft. We aim to understand what happens to social accounts after cybercriminals acquire credentials to those accounts through illicit means. In other words, we seek to understand their accesses and the actions they perform in the accounts. This will help the security community in two ways. First, we will shed light on an understudied domain (it is hard to study compromised accounts without being in control of a

[2] https://www.wired.com/2012/08/apple-amazon-mat-honan-hacking/

large online service, hence academic literature is sparse in this area). Second, our findings will help online services to tune and improve their detection and mitigation tools.

To this end, we built a system to understand what happens to Facebook accounts post-compromise, by leveraging the general honey assets method in Chapter 3. We then created and deployed 1008 realistic decoy Facebook accounts (for ethical reasons, it is not possible for us to study accounts that belong to real persons, to avoid harming them). To lure visitors into interacting with the accounts, we leaked credentials of a subset of the accounts on the Surface Web and Dark Web, mimicking the modus operandi of cybercriminals that distribute stolen account credentials. We monitored the accounts for one month, extracted comprehensive activity records of people visiting the accounts, and analysed those records offline.

We observed 215 unique accesses to 235 accounts that resulted in 478 actions in those accounts. We show the different types of actions that visitors performed in the accounts, and analyse the search terms they entered in the accounts (this reveals the type of content that they were interested in). We also show the content that they posted in the accounts. Finally, we present the locations that logins originated from, and describe the devices that connected to the accounts. These detailed measurements paint a picture of the activity of visitors in Facebook accounts, and will be useful in developing better tools and techniques to secure social accounts.

Research questions. Our research questions are as follows. Will differences in account demographics (age and gender) affect the activity of visitors in compromised social accounts? How long do they stay in social accounts after logging in? What is the nature of content that they search for in social accounts? What is the nature of content that they post in social accounts?

5.3 Background

In this section, we describe the features and functionalities of regular Facebook accounts and a special type of sandboxed Facebook accounts.

5.3.1 Facebook accounts

A potential Facebook user first creates an account and an associated *profile*. Afterwards, they can send *friend requests* to their peers. They can post updates on their profile *timeline*, for instance, by writing text, uploading a photo, or posting a URL (or a combination of those actions). Facebook also allows users to send private messages to their friends via *Messenger* (Facebook's messaging application). Users can click *like* (and other reactions) on posts, photos, and other content of interest to them. Facebook usage is not limited to individual users. Informal groups, businesses, and corporate entities can also maintain Facebook presence by creating *pages* and *groups*. Users can search for, and connect to, friends, groups, and pages they are interested in. These features, among others, highlight the social nature of Facebook.

5.3.2 Whitehat accounts

In addition to regular accounts, Facebook provides sandboxed accounts that are disconnected from regular accounts. These accounts, known as *whitehat accounts*, have similar functionality and visual similarity to real accounts, but exist in an isolated environment (a sandbox). Hence, they cannot connect to regular accounts. They are often used for testing purposes, for instance, security vulnerability testing.[3] Figure 5.1 shows the profile header of a whitehat account (one of the experimental accounts deployed later in this chapter). It looks similar to the profile header of a regular Facebook account. The inherent isolation of whitehat accounts makes them particularly suitable for our studies into understanding malicious activity in compromised social accounts, since it ensures that real users will not be harmed in any way during experiments, and this matches our ethics requirement[4] for studies of this nature. We discuss these ethical considerations in Section 5.4.5.

Facebook also provides a dashboard for managing whitehat accounts. The dashboard, which is accessible only from a real Facebook account, allows the account

[3] https://www.facebook.com/whitehat/info/

[4] Recall our ARMER requirements in Chapter 3. "E" stands for ethical.

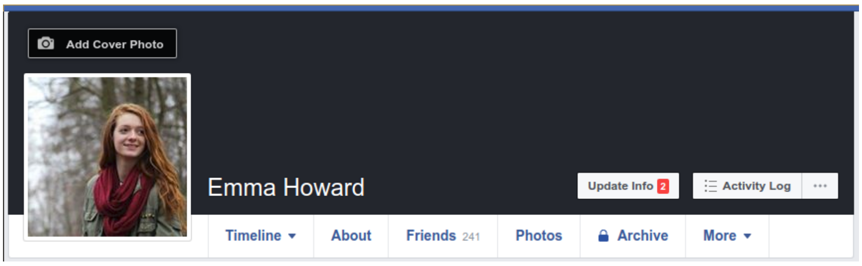
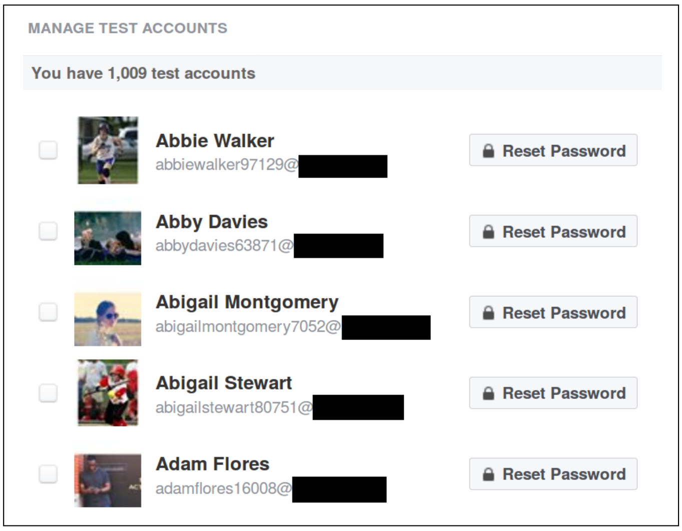


Figure 5.1: This is the profile header of a whitehat account. Similar to a regular Facebook account, it features a profile photo, the name of the account owner, and additional information about the account.



MANAGE TEST ACCOUNTS

You have 1,009 test accounts

□ Abbie Walker
abbiewalker97129@[redacted]
Reset Password

□ Abby Davies
abbydavies63871@[redacted]
Reset Password

□ Abigail Montgomery
abigailmontgomery7052@[redacted]
Reset Password

□ Abigail Stewart
abigailstewart80751@[redacted]
Reset Password

□ Adam Flores
adamflores16008@[redacted]
Reset Password

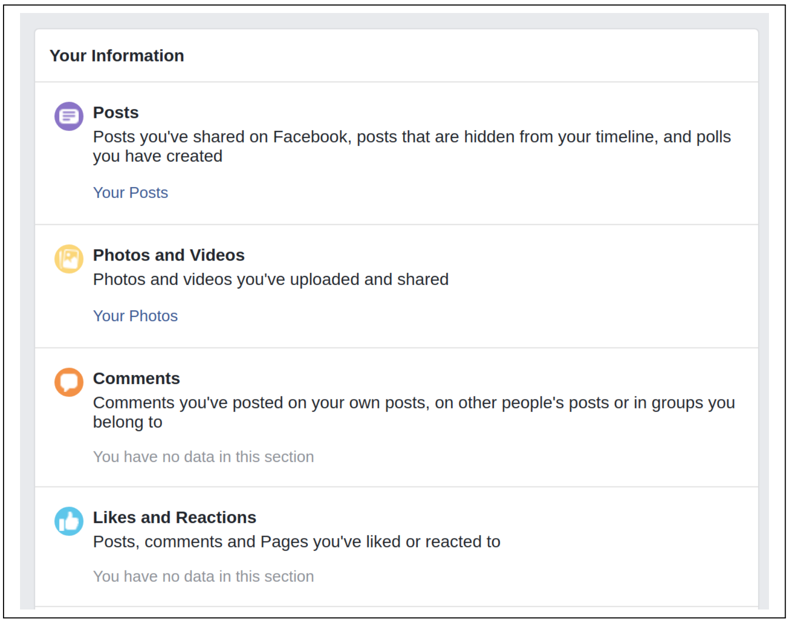
Figure 5.2: Facebook's whitehat dashboard allows the manager of whitehat accounts to reset passwords of accounts under their control.

manager to reset passwords of whitehat accounts under their control. Figure 5.2 shows an example whitehat dashboard.

5.3.3 Download Your Information (DYI)

A Facebook user may desire to download and review their own account data and activity. To facilitate this, Facebook accounts present a built-in tool known as Download Your Information (DYI)[5] which allows users to request and download a compressed archive containing their account data and activity over time. Alternatively, this data can be downloaded in JavaScript Object Notation format (JSON). The DYI tool is

[5] https://www.facebook.com/help/1701730696756992



Your Information

Posts
Posts you've shared on Facebook, posts that are hidden from your timeline, and polls you have created
Your Posts

Photos and Videos
Photos and videos you've uploaded and shared
Your Photos

Comments
Comments you've posted on your own posts, on other people's posts or in groups you belong to
You have no data in this section

Likes and Reactions
Posts, comments and Pages you've liked or reacted to
You have no data in this section

Figure 5.3: An example home page in an uncompressed DYI archive downloaded from one of our honey accounts. It is organised in clickable sections – the user can click through those sections to view detailed information about them.

available via the *Settings* menu of Facebook accounts.

After requesting and downloading the compressed archive (DYI archive), the user can then uncompress the archive offline and peruse its contents. It is usually structured like an offline web site organised in directories (sections) and web pages that can be viewed offline in a web browser. Those pages contain detailed account and activity records about the user, for instance, uploaded photos, IP addresses, and private messages. Figure 5.3 shows an example home page in an uncompressed DYI archive downloaded from one of our honey accounts. It is organised in clickable sections – the user can click through those sections to view detailed information about them.

Given the wealth of activity information and account data present in DYI archives, they play a central role in the honeypot infrastructure presented in this chapter, as explained further in Section 5.4.2. In other words, we rely on DYI functionality in Facebook accounts to retrieve activity data from honey accounts at the end of experiments.

5.4 Method and experimental setup

In this section, we describe our honey accounts. We also present the data collection infrastructure that retrieves and processes data from honey accounts.

5.4.1 Setting up honey accounts

Demographics. Lévesque et al. [64] examined gender and age, among other demographic factors, as risk factors in malware infections. Inspired by their approach, we designed personas around two demographic attributes, namely age range (teen/adult) and gender (male/female). We wanted to observe differences or similarities in the behaviour of visitors to the honey accounts, depending on the demographic attributes of the accounts. To this end, we created 1008 profiles in total, comprising equal numbers of female adult, male adult, female teen, and male teen accounts.

Profile names and passwords. We assigned first and last names to the profiles by generating random combinations of names using the API of *Random User Generator*.[6] We then assigned passwords to the profiles by randomly selecting passwords from the publicly available *RockYou* password list, comprising 32 million passwords that were exposed during a December 2009 data breach.[7] Finally, we created 1008 Facebook whitehat accounts based on the profiles described earlier. To increase the realism of the accounts, we established friend connections among them to mimic the social nature of real Facebook accounts.

Profile photos. We sourced profile photos for the accounts by downloading Creative Commons (CC) stock photos from *Pixabay*,[8] *Flickr*,[9] *Pexels*,[10] and *Unsplash*.[11] We chose only CC0-licensed photos from those sources, that is, the photos that can be used for any purpose (they also do not require attribution). We manually matched

[6] https://randomuser.me/

[7] https://www.theregister.co.uk/2010/01/21/lame_passwords_exposed_by_rockyou_hack/

[8] https://pixabay.com/

[9] https://www.flickr.com/

[10] https://www.pexels.com/

[11] https://unsplash.com/

photos to accounts, taking care to ensure that each profile photo represented the previously designated demographic attributes of its host account. For instance, for a female adult account, we chose a profile photo that shows an adult woman. Finally, we uploaded the curated profile photos to honey accounts using a photo upload automation tool that we built for this purpose. Thus, at a glance, the demographic label of any given account can be inferred by anyone that connects to the account.

Timeline data. To further mimic real Facebook accounts, we posted some content on the timelines of honey accounts. To this end, we collected publicly available tweets containing popular hashtags using the Twitter Streaming API.[12] These popular hashtags, identified in previous work [6], include `#sports`, `#music`, and `#news`, among others. We removed personally identifiable information (PII) from the tweets and wrote the sanitised text snippets on timelines of honey accounts using an automation tool that we built for this purpose. Hence, the honey accounts display diverse content on topics that people usually post on social networks, and are more convincing, as a result. We also considered populating the accounts with popular song lyrics, but discarded the idea because of copyright restrictions on musical lyrics.

5.4.2 Data collection infrastructure

In this section, we present the data collection infrastructure that we built to retrieve activity data from honey accounts.

DYI feature. As described earlier, Facebook accounts, including whitehat accounts, provide a feature for account owners to download a compressed archive containing comprehensive records of their activity on Facebook. We rely on this feature to collect activity records of visitors to honey accounts. Hence, at the end of experiments, we downloaded each account's DYI archive and parsed it offline. We then analysed this data to gain insight into the modus operandi of visitors to compromised social accounts.

[12] https://developer.twitter.com/en/docs

DYI archive parser. As stated earlier, DYI archives comprise web pages containing activity details for offline viewing. We built a parser to automatically extract and categorise the data presented in those web pages. Some of these details include login and logout information, device information, and password changes, among others.

Account health inspector. Visitors sometimes hijack honey accounts during experiments by changing the passwords of such accounts. It is therefore necessary to keep track of the *health status* of accounts, to know the ones that are still accessible and the ones that have been hijacked (in other words, *unhealthy*). To this end, we developed a tool (account health inspector) to periodically connect to all our honey accounts and report their statuses. This inspector connects to each account, navigates to its *activity log* page, and records that page for offline parsing. Note that this is different from fetching a DYI archive. The inspector allows us to check two things. First, we can verify that the accounts are healthy, and carry out remedial actions otherwise, for instance, by resetting their passwords through the whitehat dashboard. Second, the recorded activity provides some information about actions in the accounts, but it is not as comprehensive as a DYI archive. Nevertheless, it gives early insights into activity in the accounts, pending DYI downloads at the end of experiments.

Email notifications. While setting up whitehat accounts, we associated certain email addresses to the honey accounts. Those email addresses point to a mail server under our control. On that mail server, we receive real-time email notifications from honey accounts about password changes, incoming friend requests, and received private messages, among others.

In summary, DYI archives, account health inspector reports, and email notifications from the honey accounts provide us with a comprehensive view of honey accounts. Figure 5.4 shows the interconnections among the above listed components of our data collection infrastructure.

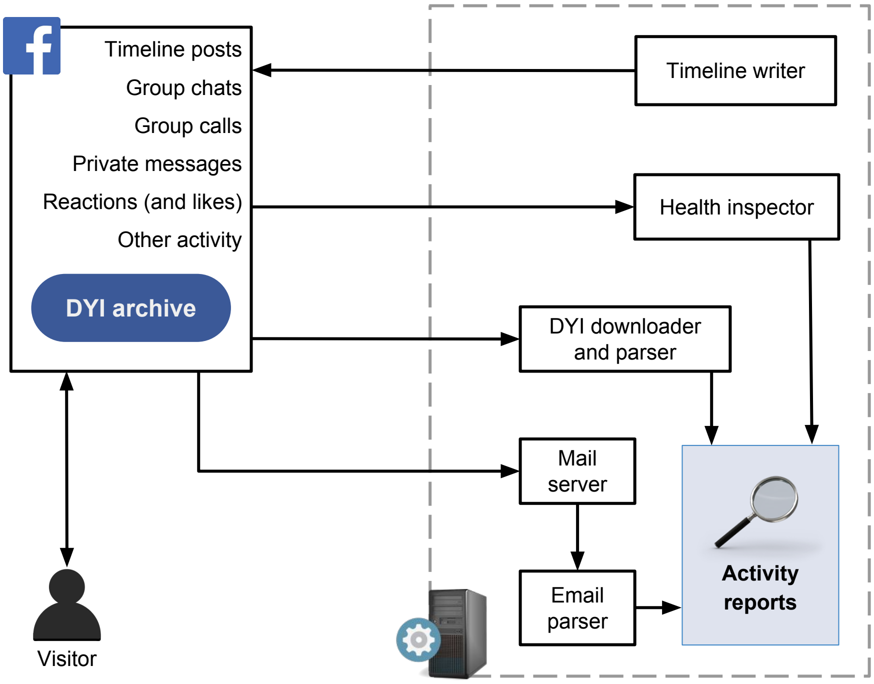


Figure 5.4: Social honeypot infrastructure. Honey accounts report activity records to us via the DYI downloader, mail server, and health inspector.

5.4.3 Leaking honey credentials

Stolen credentials are often distributed on paste sites and other outlets by cybercriminals [91]. Hence, we mimicked the credential-leaking approach to attract cybercriminals to our honey accounts by leaking their credentials via paste sites on the Surface Web and the Dark Web (see details in Table 5.1). These paste sites are ideal outlets because they allow public pastes and show recent pastes to all visitors. Besides, paste sites have successfully attracted visitors to honey assets in previous work [76, 62, 16].

We did not leak the entire population of honey accounts. Instead, we leaked two-thirds of them, in other words, only 672 credentials out of the entire set of 1008 credentials. We did this to observe if visitors will attempt to compromise the accounts that were not leaked by leveraging existing friend connections among the accounts. For instance, they might send phishing messages or malicious links to accounts that we did not leak.

Table 5.1: To lure visitors to honey accounts, we leaked account credentials through paste sites on the Surface Web and the Dark Web. We chose these paste sites because they allow public pastes and successfully attracted visitors to honey accounts in previous work [76, 62, 16].

Name	*Type*	*URL*
Pastebin	Surface Web	`https://pastebin.com/`
Paste.org.ru	Surface Web	`http://paste.org.ru/`
Stronghold	Dark Web (via TOR)	`http://nzxj65x32vh2fkhk.onion/`

Given the large number of credentials that we leaked (672 accounts), we divided them into seven chunks, each chunk comprising a maximum of 100 credentials. Note that the recent pastes feature of paste sites imposes a fading effect on the visibility of leaks. Hence, to ensure that our leaks favour paste site visitors from multiple timezones that differ from ours, we leaked credentials twice daily. Finally, to ensure that the credentials were adequately exposed during leaks, we randomised the order of credentials in each chunk prior to leaking them. Our assumption is that most visitors that see the leaks will pay more attention to credentials at the top of each chunk than the ones that appear later in the chunk. To compensate for this potential effect, we ensured that the credentials appeared in a different (random) sequence in each leak instance. This has an unintended positive effect — each leak instance appears unique to the human eye due to the random order of elements.

5.4.4 Threats to validity

We acknowledge that there are some factors that may affect the validity of our findings. First, the content of the honey accounts comprise stock photos and other publicly available data, which might be obvious under close scrutiny. Also, a close look might reveal that the honey accounts were created fairly recently — this can possibly influence the credibility of our accounts. Second, recall that we used sandboxed accounts (whitehat accounts) that are disconnected from regular Facebook accounts. A close observation may reveal the presence of features that differ slightly from real accounts. However, this does not pose a major risk to experiments. Third, we leaked credentials through paste sites only. Our findings may not be representative of malicious activity in social accounts stolen via other outlets, for instance, malware or underground forums. Despite these factors, this chapter offers insights into malicious activity in stolen social accounts and will help in developing detection and mitigation systems and techniques.

5.4.5 Ethics

We carefully considered the ethical implications of this study while setting up and running experiments. First, we used accounts that are isolated from regular Facebook accounts to avoid harming legitimate Facebook users. This sandboxing approach is in line with common practices in malware research, which is related to our work [82]. Second, we used publicly available stock photos and tweets to populate the accounts. We did this to ensure that no private information was leaked in this study. Third, by leveraging the whitehat dashboard, we ensured that account passwords could be changed easily by us, to lock visitors out, if we observed attempts to harm people via honey accounts. Fourth, we asked our Facebook contacts to keep an eye on the accounts with a view to shutting down any account that violates Facebook's policies during experiments. Finally, since our experiments involved deceiving criminals to interact with decoy accounts, we sought and obtained ethics approval from UCL prior to starting experiments.

5.5 Data analysis

In this section, we provide an overview of the activity of visitors in honey accounts. In detail, we discuss the types of accesses that visitors made to the accounts and show differences in account activity. We also summarise the system configuration of observed accesses (browsers, operating systems, and IP addresses of the devices that connected to the accounts).

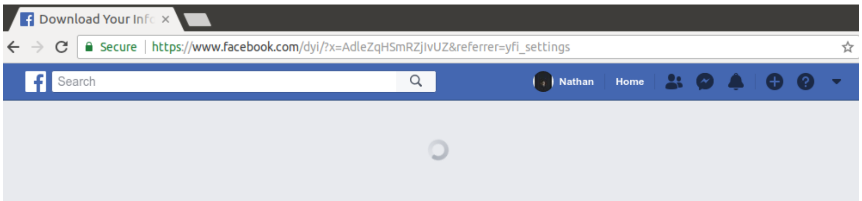

Download Your Inf
Secure
https://www.facebook.com/dyi/?x=AdleZqHSmRZjIvUZ&referrer=yfi_settings
Search
Nathan
Home

Figure 5.5: This is an example of a defective honey account. It presents an infinite spinning GIF, indicating page load, but never loads content. It was impossible to download DYI archives from defective honey accounts, hence we excluded them from data analysis.

5.5.1 Discarding defective accounts

As described in Section 5.4.2, our data collection process involves downloading DYI archives from all honey accounts. While downloading those archives, we discovered that 158 accounts were defective. They presented spinning GIFs, indicating infinite page load, instead of presenting page content. It was impossible to download DYI archives from those accounts so we excluded them from data analysis. Figure 5.5 shows an example of a defective account. We have reached out to our Facebook contacts to look into the accounts. The presence of defective accounts reduced the effective number of honey accounts under analysis from 1008 to 850 fully functional accounts. These functional accounts comprise 428 adult accounts and 422 teen accounts (from the age range point of view), or 427 female accounts and 423 male accounts (from the gender point of view). Finally, the effective number of (functional) leaked accounts reduced from 672 to 569 (after excluding the defective accounts).

5.5.2 Accesses and associated actions

Facebook accounts record unique accesses to accounts, and each access is labelled with a unique string identifier known as a *cookie*. Cookies can be found in the login records section of DYI archives. An access is recorded when a visitor connects to a honey account. Note that access identifiers (cookies) can persist across logins into different accounts. For instance, if a visitor connects to account A and then connects to another account B using the same device and browser within a short time, the same cookie will be recorded in both accounts. After logging in, a visitor performs zero or more *actions*, for instance, sending a private message or writing a status update. In other words, an access results in zero or more actions in a honey account. In this chapter, we use the terms *cookie* and *access* interchangeably. We observed various types of accesses in the accounts and named them according to the action(s) they performed in the accounts. These types of accesses, codified into a taxonomy of accesses, are described next.

5.5.3 Taxonomy of accesses

As earlier mentioned, accesses can be described by the action(s) linked to them. We observed the following types of accesses in honey accounts. Note that we have more access types listed here, for Facebook accounts, than Gmail accounts (presented in Chapter 4). This is because social accounts present more features and nuance than webmail accounts.

Curious. A curious access is recorded when a visitor connects to a honey account and does nothing. This implies that the visitor was likely just checking to see that accounts are real. In other words, a curious access has no associated action.

Hijacker. A hijacker access is recorded when the password of a honey account (or its email address) is changed.

Chatty. This type of access is recorded when a visitor sends private messages, creates group chats, posts an update on the timeline of another account, or posts on its own timeline.

Emotional. An emotional access is recorded during clicks on a Facebook “like” button (or any other reaction) on photos and posts.

Gold digger. This type of access is recorded when a visitor enters search terms in the search bar.

Profile editor. Profile editor accesses are recorded when a visitor edits the profile of a honey account (for instance, by changing the profile photo or other profile information about the account owner).

Table 5.2: Summary of actions in honey accounts grouped by access type. Note that the curious type is excluded from this table. This is because curious accesses do not perform any action in honey accounts. Gold digger and friend modifier accesses are responsible for the vast majority of recorded actions (they account for 47% and 22% of all actions respectively).

Access type	*Number of actions*	*Percentage*
Gold digger	224	46.86
Friend modifier	104	21.76
Chatty	90	18.83
Hijacker	31	6.49
Profile editor	15	3.14
Emotional	14	2.93
Total	**478**	**100.00**

Friend modifier. This type of access is recorded when a visitor adds or removes a friend from a honey account.

These types of accesses are not mutually exclusive, except for the curious type. A single access with one or more actions can have one or more access types, excluding the curious type, which is reserved for accesses that do not have any associated action. For instance, an access that is chatty can also be emotional, depending on its actions.

5.5.4 Actions

In total, we observed 215 unique accesses to 235 accounts, which resulted in 478 actions in those accounts. Table 5.2 shows a summary of actions grouped by access type. Recall that a unique access can be responsible for zero or more actions. Table 5.2 excludes accesses of the curious type since they are not responsible for any action. Gold digger and friend modifier accesses dominate the table of actions, responsible for 47% and 22% of all actions respectively. Profile editor and emotional accesses are the least active types. This shows that visitors are mostly interested in searching for information through the Facebook search bar (details can be found in Section 5.5.7), and adding or removing friends from accounts. Next, we study the timing of activity in honey accounts, with particular emphasis on how long the recorded accesses stayed connected to the accounts.

5.5.5 Timing of account activity

We set out to understand the time patterns of accesses to accounts. To this end, we measured how long it took visitors to connect to the accounts after we leaked account credentials, and how long they stayed connected to the accounts. These measurements were carried out across all accounts, and also on groups of accounts (by age range and gender), to observe differences in activity patterns across different types of accounts. We present detailed measurements next.

Leaks to logins. Recall that we leaked credentials of honey accounts via paste sites to attract visitors to them. To observe how long it took them to connect to accounts after the leaks, we computed time lags between the first leak and the first login (access) recorded in each account. Note that account credentials were leaked simultaneously multiple times. In this analysis, we focused on the first leak (dated 1st June, 2018). As the CDF in Figure 5.6 shows, accounts were mostly not accessed instantly. Instead, visitors connected to them gradually for several days after the first leak. 40% of the accounts were accessed in 350 hours or less, after the first leak (in other words, within 15 days).

Access duration. To understand how long visitors stayed in honey accounts, we computed the durations of their accesses. To achieve this, we recorded the time that a cookie first appeared in an account as t_0, and the last time it appeared in that account as t_{last}. Given this information, access duration can be computed as $t_{last} - t_0$ for each access. Figure 5.7 shows CDFs of access duration grouped by access type. Curious and emotional accesses tend to be short-lived compared to the remaining types of accesses that stay connected to the accounts for longer periods of time. The CDFs in Figure 5.7 corroborate Table 5.2 — gold digger, friend modifier, and chatty accesses spend more time in accounts and are responsible for more actions than emotional and hijacker accesses, for instance. We also computed access durations by age range to see if there were differences in access durations in adult accounts compared to teen accounts. The CDFs in Figure 5.8 show that visitors spend slightly more time in teen accounts than adult accounts. Finally, we

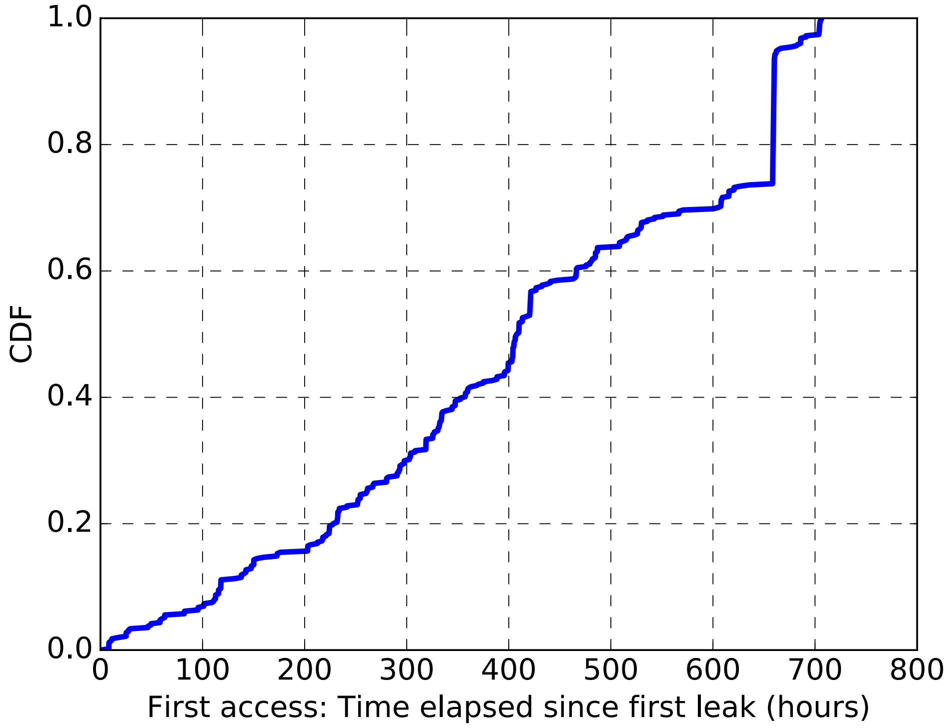


Figure 5.6: CDF of the time difference between the first instance of credential leaks (across all outlets) and first connections made to accounts by visitors. 40% of the leaked accounts were accessed within 350 hours or less (in other words, within 15 days).

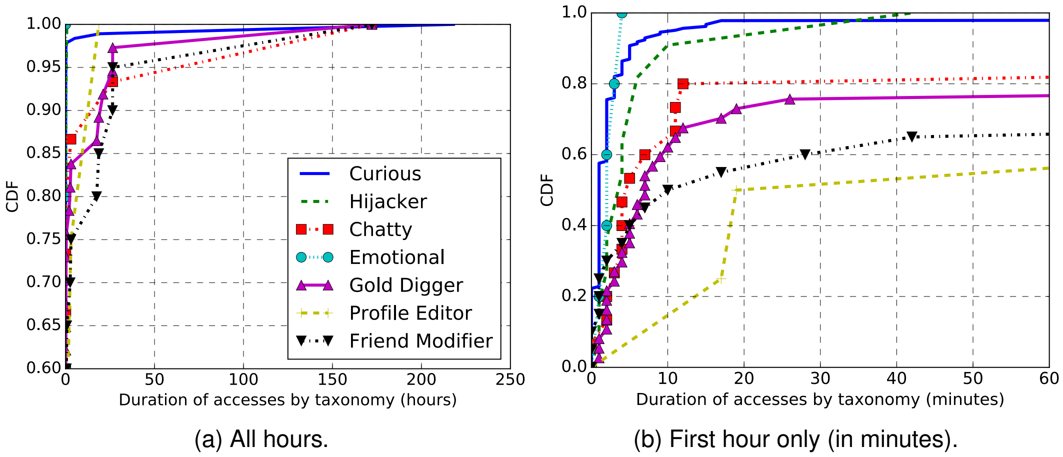
Table 5.3: We conducted KS tests to compare access durations for each access type to all access types combined.

Type (access durations)	*KS statistic*	*P-value*
Gold digger	0.3894438	0.0000618
Profile editor	0.8731884	0.0016149
Curious	0.1684783	0.0032443
Friend modifier	0.3826087	0.0058528
Chatty	0.3905797	0.0183591
Hijacker	0.2457181	0.4853071
Emotional	0.2862319	0.7395308

computed access durations by gender, to see if there were differences in access durations in female accounts compared to male accounts. The CDFs in Figure 5.9 show that visitors spend slightly more time in female accounts than male accounts.

Statistical tests. To test the statistical significance of differences in access duration by type, age range, and gender, we relied on the two-sided Kolmogorov-Smirnov (KS) test. The null hypothesis is that both samples under examination belong to identical statistical distributions. The output of the test is a KS statistic and p-value. A small KS statistic or high p-value shows that we cannot reject the null hypothesis. First, we tested the access durations of each access type against all access durations, to see the access types for which we can reject the null hypothesis. As Table 5.3 shows, gold digger accesses differ most from the distribution of all accesses (in other words, we can clearly reject the null hypothesis), while emotional accesses differ least (we cannot reject the null hypothesis).

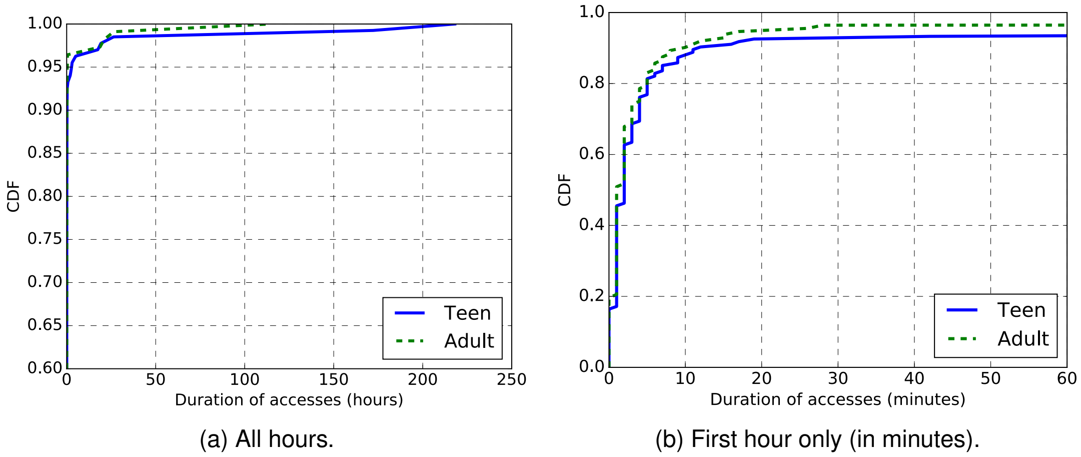
Next, we set out to determine whether adult and teen access durations belong to the same distribution or not (*null hypothesis* — adult and teen access duration vectors belong to the same distribution; statistic=0.055, p-value=0.992, cannot reject null hypothesis). Likewise, we conducted another KS test to see if the female and male access duration vectors belong to the same distribution or not (*null hypothesis* — female and male access duration vectors belong to the same distribution; statistic=0.102, p-value=0.532, cannot reject null hypothesis). In both tests, the null hypothesis cannot be rejected.



(a) All hours.

(b) First hour only (in minutes).

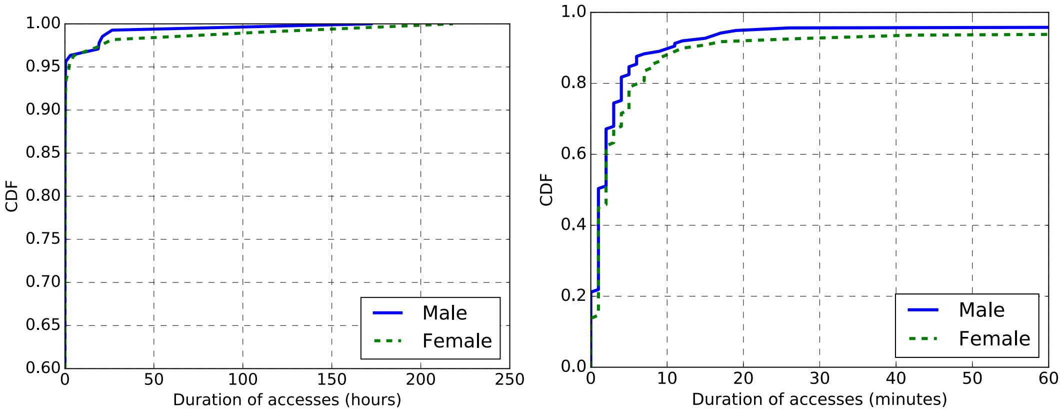
Figure 5.7: CDFs of access duration per access type. 5.7a shows the entire duration of experiments while 5.7b shows the first hour only. To enhance the visibility of the curves, the y-axis of 5.7a shows only the 60th to the 100th percentile ticks, while 5.7b shows all percentile ticks. Curious and emotional accesses tend to be short-lived compared to the remaining types of accesses that stay logged in longer.



(a) All hours.

(b) First hour only (in minutes).

Figure 5.8: CDFs of access duration per age range. 5.8a shows the entire duration of experiments while 5.8b shows the first hour only. They show that visitors spend slightly more time in teen accounts than adult accounts. To enhance the visibility of the curves, the y-axis of 5.8a displays only the 60th to the 100th percentile ticks.



(a) All hours.

(b) First hour only (in minutes).

Figure 5.9: CDFs of access duration per gender. 5.9a shows the entire duration of experiments while 5.9b shows the first hour only. To enhance the visibility of the curves, the y-axis of 5.9a shows only the 60th to the 100th percentile ticks. The CDFs show that visitors spend more time in female accounts than male accounts.

5.5.6 Further demographic analysis

We wanted to understand differences in the types of accesses recorded in the accounts per age range and gender. To this end, we calculated the proportions of access types in each range range and gender. As Figure 5.10 shows, teen accounts present more chatty and emotional accesses than adult accounts, while adult accounts show more friend modifier accesses than teen accounts. Figure 5.11 shows that female accounts present more friend modifier accesses than male accounts (proportionally). Male accounts present some profile editor accesses, while female accounts present none. Finally, male accounts present more chatty and gold digger accesses than female accounts.

Having observed many instances of friend requests among honey accounts during experiments, we decided to study differences in friend request behaviour among the accounts. To this end, we plotted CDFs of received friend requests (with emphasis on age range and gender). Figure 5.12 shows that female accounts receive a few more friend requests than male accounts. Similarly, we observed minor differences in received friend requests in adult and teen accounts.

Statistical tests. To test the statistical significance of the minor differences in re-

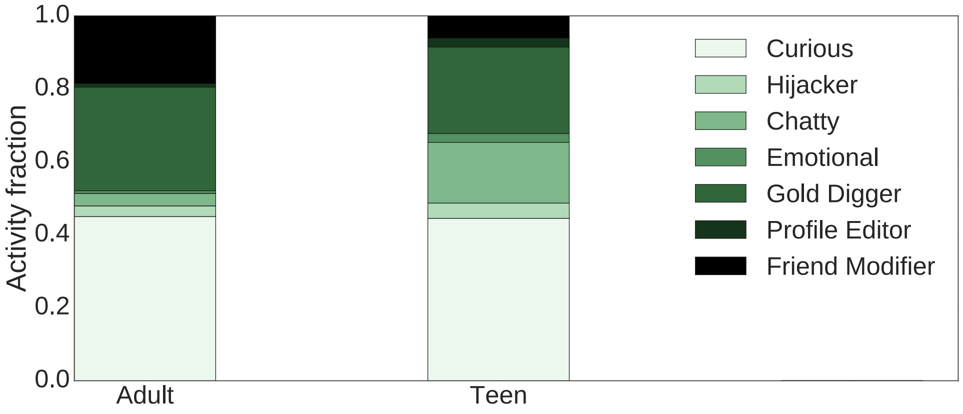


Figure 5.10: Types of accesses per age range. Teen accounts present more chatty and emotional accesses than adult accounts, while adult accounts show more friend modifier accesses than teen accounts.

ceived friend requests by age range and gender, we once again performed the two-sided Kolmogorov-Smirnov (KS) test. Recall that the output of the test is a KS statistic and p-value. A small KS statistic or high p-value shows that we cannot reject the null hypothesis. The first KS test was to determine if received friend requests in adult and teen accounts belong to the same distribution or not (*null hypothesis* — adult and teen vectors of received friend requests belong to the same distribution; statistic=0.010, p-value=1.000). Likewise, we conducted another KS test to see if the female and male vectors of received friend requests belong to the same distribution or not (*null hypothesis* — female and male vectors of received friend requests belong to the same distribution; statistic=0.063, p-value=0.359). In both tests, the null hypothesis cannot be rejected.

5.5.7 What gold diggers seek

As shown in Table 5.2, gold digger accesses were responsible for a substantial share of actions in honey accounts (47%). Various search terms were recorded in 52 accounts (those search terms were entered in the Facebook search bar of honey accounts). To understand what visitors were searching for, we analysed the search logs in DYI archives and found many varieties of search terms.

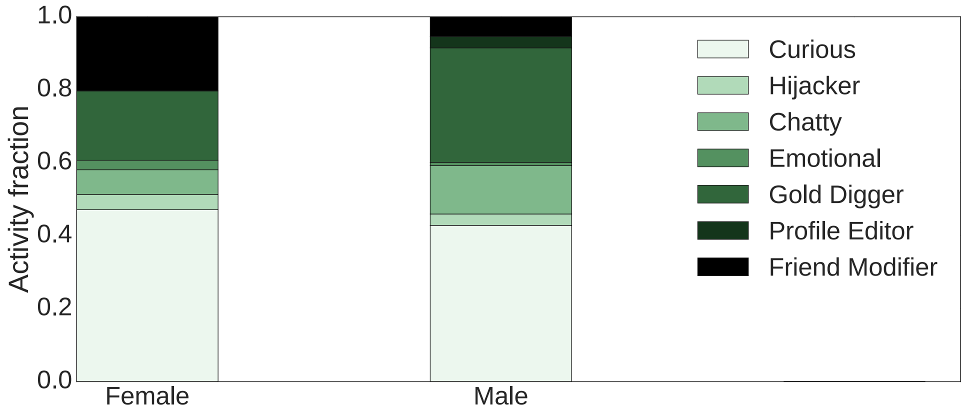
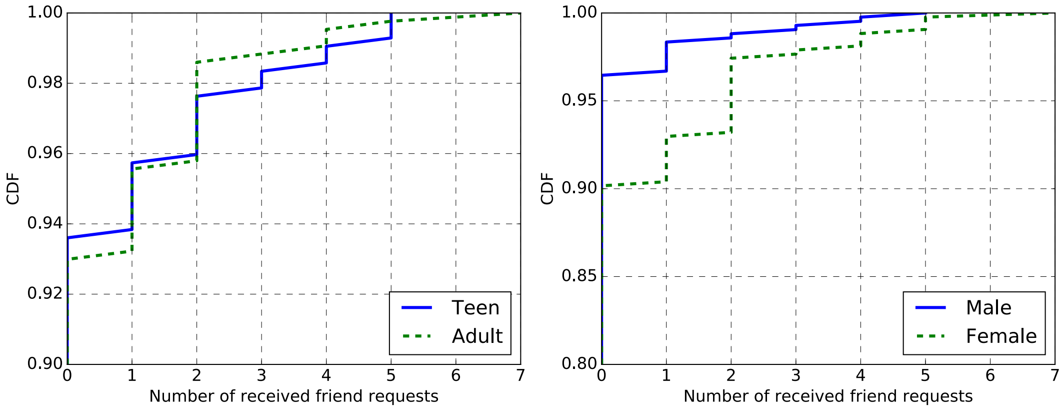


Figure 5.11: Types of accesses per gender. Female accounts present more friend modifier accesses than male accounts (proportionally). Male accounts present some profile editor accesses while female accounts present none. Also, male accounts present more chatty and gold digger accesses than female accounts.



(a) Per age range.

(b) Per gender.

Figure 5.12: CDFs showing the distribution of received friend requests. 5.12b shows that female accounts received more friend requests than male accounts, while 5.12a indicates minor differences in the number of received friend requests in teen and adult accounts. Note that both plots display only very high percentile ticks, for visibility reasons.

Table 5.4: Top ten words among the search terms entered in honey accounts. These include atheism- and religion-related words in Spanish (or Portuguese) as a result of a visitor's numerous searches for debates on atheism and religion. Other interesting search terms that showed up include "india vs england live" and "bin carding."

Searched words	*TF-IDF*
debates	0.4293
ateísmo	0.3578
bihar	0.3578
religiões	0.3220
robson	0.2326
india	0.1431
oaxaca	0.1252
salina	0.1252
cruz	0.1252
fajar	0.1252

We relied on Term Frequency – Inverse Document Frequency (TF-IDF),[13] a natural language processing metric, to analyse the search logs, similar to our approach in previous work [76, 16]. Given a text corpus, TF-IDF ranks words in the corpus by assigning weights to them, between 0 and 1. Words weighted close to 1 are more important in the corpus than words weighted close to 0. To observe the top words that visitors searched for in the accounts, we used TF-IDF to obtain the top 10 words in the search logs as shown in Table 5.4. They include atheism- and religion-related words in Spanish (or Portuguese) as a result of a visitor's numerous searches for debates on atheism and religion. Other interesting search terms that showed up in search logs include "india vs england live," "bin carding," and "marvel cinematic universe." These search terms reveal the presence of a wide variety of benign and malicious interests in search terms. Note that word stemming was not applied during this analysis because the corpus contained a mixture of words in multiple languages. Stemming is best done on a corpus of text in a single language.

5.5.8 (Anti)social chatter

Recall that Table 5.2 shows that chatty accesses were responsible for 19% of all recorded actions. What did chatty visitors post in honey accounts? We observed

[13] Chapter 4 contains a detailed mathematical explanation of TF-IDF.

Table 5.5: Top ten words extracted from the text corpus comprising comments, private messages, and posts made in the honey accounts. Greetings showed up in the corpus (“hello” and “hi baby,” for instance). We also observed some posts alerting account owners about data breaches (unknown to the posters, we leaked credentials intentionally). Some visitors, in apparent moments of awareness, also posted comments that our honey accounts were fake.

Chatty words	*TF-IDF*
hi	0.3842
baby	0.2744
hii	0.2744
my	0.2744
you	0.2195
fake	0.1646
password	0.1646
am	0.1646
change	0.1646
better	0.1098

chatty behaviour in 29 accounts. These comprise attempted group calls, “waves,” private messages, posts on own timeline and other timelines. Private messages ranged from the “hello” and “hi” types to sexually explicit messages. Timeline posts ranged from short meaningless posts to morbid posts (for instance, “killing my family with an assault rifle from ww2”). There were some posts warning account owners about data breaches including leaked credentials (unknown to the posters, we leaked credentials intentionally). Finally, some comments stated that the accounts were fake. Surprisingly, we did not observe any post containing phishing or malware-laden links. To observe the top words in the chatty text corpus, we once again applied the TF-IDF technique (previously described in Section 5.5.7). The top 10 chatty words are shown in Table 5.5. Note that we also did not perform stemming because of the presence of multiple languages in the chatty text corpus.

5.5.9 System configuration of accesses

Leveraging the user-agent string information in DYI archives, we extracted browser and operating system information from the observed accesses. A wide range of browsers and operating systems were used to access honey accounts. Table 5.6 shows a summary of those browsers. Chrome and Android Browser top the list of browsers, at 36% and 29% respectively. A small percentage of accesses were made using PhantomJS,[14] a web automation tool. Table 5.6 shows that some connections to honey accounts were made manually, while others were made automatically. Table 5.7 shows an overview of the operating systems on the devices that connected to honey accounts. Windows and Android dominate the list (55% and 34% respectively). A small fraction of accesses were also made with iPhones.

5.5.10 On the origins of accesses

In total, we observed 209 IP addresses (IPv4 and IPv6 addresses) from 47 countries. Of these IP addresses, 49 were TOR exit nodes. It is possible that some of the remaining IP addresses were proxies or VPN nodes. To understand the geographical locations that accesses originated from, we extracted all IP addresses associated with accesses from the DYI archives. We then carried out IP geolocation using *IP-API*,[15] an IP geolocation service that provides timezone and location information about IP addresses.

Figure 5.13 shows a world map with markers showing the locations that accesses originated from. As the map shows, connections originated from many locations around the world. It is interesting to note a particularly dense cluster of accesses from Europe. No access originated from China — note that Facebook is banned in China. It is possible that visitors connected to some accounts via proxies or VPNs. However, we did not observe any evidence confirming or refuting this.

5.6 Interesting case studies

As shown in Section 5.5.9, many accesses were made via mobile devices (especially Android devices and iPhones). We observed three cases in which visitors (inadvertently) synchronised their mobile phone contacts, comprising names, phone numbers, and occasionally, email addresses, to our honey accounts. In the first ac-

[14] `http://phantomjs.org/`

[15] `http://ip-api.com`

Table 5.6: Various browsers were used to connect to the accounts, including desktop and mobile variants. The presence of PhantomJS, a web automation tool, shows that some accesses were made using automation tools. This wide variety of browsers (including mobile browsers) reveals a mix of manual and automated accesses.

Browser	*Percentage*
Chrome	35.98
Android Browser	29.13
Firefox	26.95
Unknown Browser	2.34
Edge	2.34
Safari	2.02
Opera	0.62
Internet Explorer	0.31
PhantomJS	0.31
Total	**100.00**

Table 5.7: Distribution of operating systems in accesses to honey accounts. The vast majority of accesses were made using Windows and Android devices. More than one-third of accesses were made via mobile devices (Android and iOS devices).

OS	*Percentage*
Windows	54.98
Android	33.80
Linux	4.05
MacOS	2.65
Unknown OS	2.65
iPhone iOS	1.87
Total	**100.00**

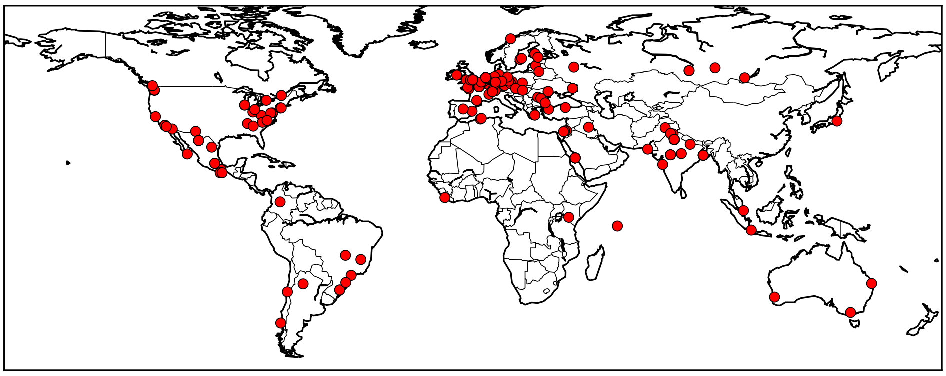


Figure 5.13: Origins of accesses determined via IP geolocation. Each marker indicates the origin of a connection to a honey account. Connections were made to honey accounts from many locations around the world. There is a dense cluster of accesses that originated from Europe. No accesses originated from China — recall that Facebook is banned in China.

count, we observed a contact list comprising 830 phone contacts. The country code prefix (+91) of the phone numbers indicates that the phone numbers are domiciled in India. In the second account, we observed a different contact list comprising 57 contacts, once again domiciled in India (country code prefix +91). In the third account, we observed yet another contact list comprising 27 Mexican phone contacts (country code prefix +52).

For ethical reasons, we did not investigate the identities associated with these phone numbers or their relationships with the visitors that connected to the honey accounts and synchronised their phone contacts. In other words, it is possible to potentially unmask them and their personal networks, but we decided against doing so, for ethical reasons. A wider implication of this observation is that visitors (and other actors) can be tricked into exposing vital information (PII) about themselves if they can be convinced to connect to honey accounts using their mobile devices. In addition to the potential harm from PII leakage, this can be potentially damaging for actors that intend to keep their own identities private (along with identities of people in their networks), for example, journalists and politicians. The countermeasure is simple — do not connect to untrusted accounts on personal mobile devices to avoid PII leakage and associated problems.

Finally, two honey accounts were used to authenticate to Instagram, according to the information we collected via DYI archives. This indicates the possibility of multi-platform attacks — cybercriminals can compromise accounts on an online service and then use those accounts to carry out further illicit activity on other services.

5.7 Summary

We presented a method to study compromised social accounts without being in control of a large online service, and implemented and deployed our honeypot infrastructure on Facebook, one of the largest social network platforms. We showed detailed measurements and analyses of accesses and actions of visitors connecting to compromised Facebook accounts under our control. We also explored differences in visitor behaviour across two demographic attributes of social accounts, namely age range and gender. Finally, we presented our findings on the origins of accesses and devices that connected to the accounts. Our approach to studying social accounts can be deployed by researchers to study other problems facing social accounts, for instance, demographic risk factors in cyberbullying, which is one of the persistent problems facing users of social accounts.

Chapter 6

Compromised Cloud Documents

6.1 Contributions

First, we introduced some improvements to the cloud document monitor infrastructure originally proposed in a 2016 USENIX workshop paper [62], following the general honey assets method presented in Chapter 3. Second, to understand what happens to compromised cloud documents, we created, instrumented, and leaked 100 decoy Google spreadsheets comprising 1000 financial records of fictional individuals. We henceforth refer to them as *sheets*. For comparison, only five decoy sheets were deployed in the pilot study [62]. In other words, we scaled up experiments by a factor of 20 in this chapter. Third, we present detailed measurements and analyses of resulting activity in the sheets, and provide insights into what happens within compromised cloud documents. The findings in this chapter will help researchers to understand what happens to compromised cloud documents and help providers of cloud services to understand ways to secure accounts and assets on such cloud services.

Collaborators. We express our appreciation to Martin Lazarov (erstwhile UCL student) for implementing and testing the first version of our cloud document monitor system under the supervision of Gianluca Stringhini and the author of this thesis. The idea of a cloud document monitor was conceived by the author while its initial implementation was assigned to Martin Lazarov, and he did an excellent job. Afterwards, the author implemented some improvements to the cloud document monitor system and performed a large-scale experiment on compromised cloud documents. This chapter presents the results of that large-scale experiment.

6.2 Overview

It is hard to imagine life nowadays without online accounts, for instance, webmail accounts for business and personal communication, e-commerce accounts for online shopping, and cloud storage accounts for convenient document storage and sharing.

Cloud documents can help to increase the productivity of teams in organisations by allowing them to collaborate easily and edit documents in real time without requiring their physical presence. As of 2014, 21% of EU citizens relied on cloud accounts to store their documents.[1] This shows the widespread adoption of cloud storage platforms, for instance, Dropbox, Google Drive, and Microsoft OneDrive.

However, there is a downside to the use of cloud accounts. Like most other online accounts, cloud accounts often accumulate sensitive information over time, for instance, financial and personal secrets. This makes them attractive targets to cybercriminals seeking to steal and monetise such sensitive information [91]. It is therefore important for researchers and cloud service providers to study and understand what happens to compromised cloud accounts and the documents they guard, as one of the necessary steps towards securing such accounts. It is hard to study attacker behaviour in online accounts unless one is in control of a large online service (as discussed in Chapter 2). Hence, there is limited research literature in this space. This research gap motivated the work presented in this chapter, with specific focus on understanding compromised cloud documents.

Previous work has shown that cybercriminals target online accounts and services to steal financial information from them, and trade stolen information via various outlets [49]. Such financial information includes payment card information,

[1] https://ec.europa.eu/eurostat/statistics-explained/index.php/Internet_and_cloud_services_-_statistics_on_the_use_by_individuals

cryptocurrency wallets, and online banking details. In recent times, there has been massive public interest in cryptocurrencies and blockchain technologies such as Bitcoin, Ethereum, and Litecoin, which provide alternative means of facilitating online transactions, among other uses. However, the advent of cryptocurrencies also introduced a new wave of cybercriminals that steal digital money (cryptocurrency wallets), sometimes leading to huge losses, as seen in the 2014 high-profile attack on a cryptocurrency exchange known as Mt. Gox ($460 million in losses).[2]

To understand what happens to compromised cloud documents containing financial information, we created decoy documents and inserted fake traditional bank payment information and cryptocurrency details in them. In other words, following the general honey assets method proposed in Chapter 3, we set up 100 fake payroll sheets comprising comprising 1000 fake records of fictional individuals. We populated the sheets with fake traditional bank payment information, fake cryptocurrency details, and fake payment links. We also installed scripts in the sheets to notify us about the activity of visitors in them and configured fake payment links in the sheets to record information about clicks on them. Unlike the pilot study [62] in which five sheets were deployed, we conducted experiments on a much larger scale in this chapter (100 sheets in total). Note that the pilot study [62] did not include any cryptocurrency information unlike the work in this chapter. To lure cybercriminals and other visitors into visiting the sheets, we leaked links pointing to the sheets via paste sites. By doing so, we mimicked the modus operandi of cybercriminals that steal and distribute stolen financial information online. This approach has been used successfully in previous work by the author [76, 62, 16]. We then recorded accesses to the sheets and tracked clicks on the fake payment URLs inside them.

Our research questions, related to the questions in the pilot study [62], are as follows. First, which actions do cybercriminals carry out on compromised cloud documents? Second, in a given document, will there be differences in the interactions of cybercriminals with different types of financial information, namely traditional bank payment information and cryptocurrency information? Third, will cybercriminals at-

[2] https://www.wired.com/2014/03/bitcoin-exchange/

tempt to carry out further attacks outside the leaked documents, for instance, by visiting payment URLs in the documents? Fourth, can we characterise the devices that cybercriminals use to connect to stolen cloud documents?

We ran experiments for a month and collected data using the infrastructure presented in Section 6.4.2. We observed 235 accesses across 98 sheets. Two sheets were not opened. We also recorded 38 modifications in 7 sheets. In Section 6.5, we present detailed measurements and analysis of accesses, modifications, edits, and devices that visited URLs in the sheets (with emphasis on IP addresses, browsers, and operating systems). In summary, we present a comprehensive picture of what happens to compromised Google spreadsheets. The findings presented in this chapter will help other researchers seeking to understand what happens to stolen cloud documents and providers of cloud services looking to understand ways to secure accounts and assets on those cloud services. This is essential because our daily activities depend heavily on cloud services.

6.3 Background

In this section, we describe cloud documents, with specific focus on Google Sheets, and explain why Google Sheets constitutes a good fit for our experiments in this chapter.

6.3.1 Cloud documents

Word processing, desktop publishing, and data processing tasks, which are ever-present business and personal tasks, can be carried out on local machines using desktop tools such as Microsoft Word, Scribus, and Apache OpenOffice Calc, among others. It is also possible and easy to use cloud-based tools for such tasks. They usually do not require complex installation processes unlike their desktop counterparts. They also allow users to collaboratively edit documents from any location unlike their desktop counterparts. Examples of cloud-based tools for creating and editing cloud documents include Google Sheets, Microsoft Office 365, and Zoho

Office Suite. These tools offer remote document hosting and editing services, and are accessible via a web browser. Next, we describe Google Sheets, the cloud-based platform that supported our experiments in this chapter.

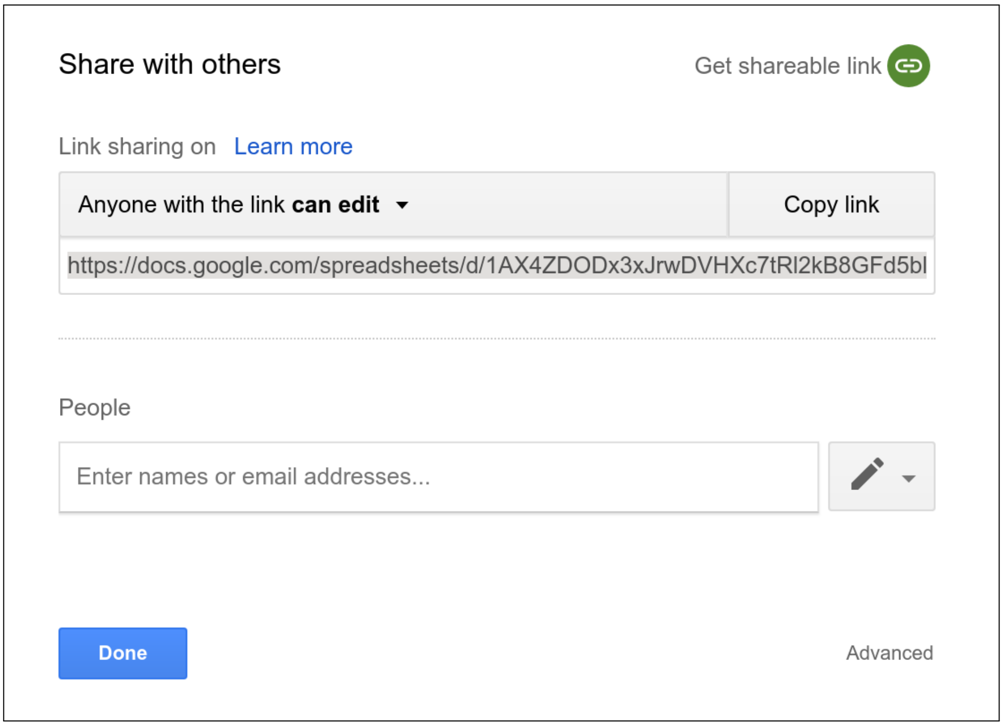
6.3.2 Google Sheets

Here, we focus on Google Sheets, a cloud-based data processing tool that allows users to create and modify sheets, and carry out data processing tasks on those sheets. Google Sheets also enables users to extend the functionalities of their sheets by incorporating scripts in them, leveraging the power of Google Apps Script[3] (a scripting engine for building lightweight web applications and augmenting Google Apps). This makes Google Sheets a good fit for the experiments in this chapter since the embedded Google Apps Script engine allows us to instrument sheets to "phone home" (report activity data), in line with requirements of the general honey assets method proposed in Chapter 3. Google Apps Script (within Google Sheets) thus plays an important role in the data collection module of the honeypot infrastructure instance presented in this chapter, similar to the one in Chapter 4 (we described how Google Apps Script works in Section 4.3.2). In this chapter, our honeypot infrastructure relies on time-driven and event-driven triggers in a custom app hidden in decoy documents, to track and report accesses and changes in those documents to us. Note that this method can be applied to other types of online documents as well, not just Google Sheets.

To create sheets, a user will first have to set up at least one Google account to host sheets. Afterwards, the user can create new sheets via a web browser. Alternatively, users can upload existing sheet data, for instance, comma-separated values (CSV) files that already contain data formatted in rows and columns. Users can edit cells in the sheets, delete rows and columns of cells, perform computations and transformations on cells, and delete entire sheets, among other operations.

For collaborative purposes, the owner of a sheet can configure the sheet to allow other users or visitors to view, comment on, or edit the sheet. Inviting collaborators

[3] https://developers.google.com/apps-script/overview



Share with others

Link sharing on

Anyone with the link **can edit**

https://docs.google.com/spreadsheets/d/1AX4ZDODx3xJrwDVHXc7tRl2kB8GFd5bl

People

Enter names or email addresses...

Figure 6.1: One of the ways to share a sheet with collaborators is by generating a long link that points to the sheet. In this example, anyone that knows the long link (highlighted in grey) can view and edit this sheet. Alternatively, the sheet owner can explicitly enter collaborators via the “People” field.

to such sheets usually involves explicitly granting them specific permissions (*view* or *edit*). The sheet owner can also generate a *long link* that points to the sheet, such that anyone that knows the long link can view or edit the sheet, depending on the privilege level assigned to the long link. The sheet owner can then send the long link to collaborators. They will visit the long link to gain access to the sheet. Figure 6.1 shows an example of a sheet configuration setting that allows visitors with knowledge of the long link to edit the sheet.

6.4 Method and experimental setup

In this section, we describe the process of creating and instrumenting honey sheets prior to experiments, and how we exposed the sheets to cybercriminals. We also describe the data collection infrastructure that powered experiments in this chapter.

6.4.1 Creating honey sheets

To observe what happens in compromised cloud documents, we created sheets containing two types of financial information, namely traditional bank payment information (bank account numbers and sort codes) and cryptocurrency information (Bitcoin addresses). We designed the sheets to look like payroll spreadsheets including salary information, following the approach employed in [62]. It is important to note that the sheets in [62] did not include any cryptocurrency information, while half of our sheets in this chapter did. Next, we describe honey sheet data in detail.

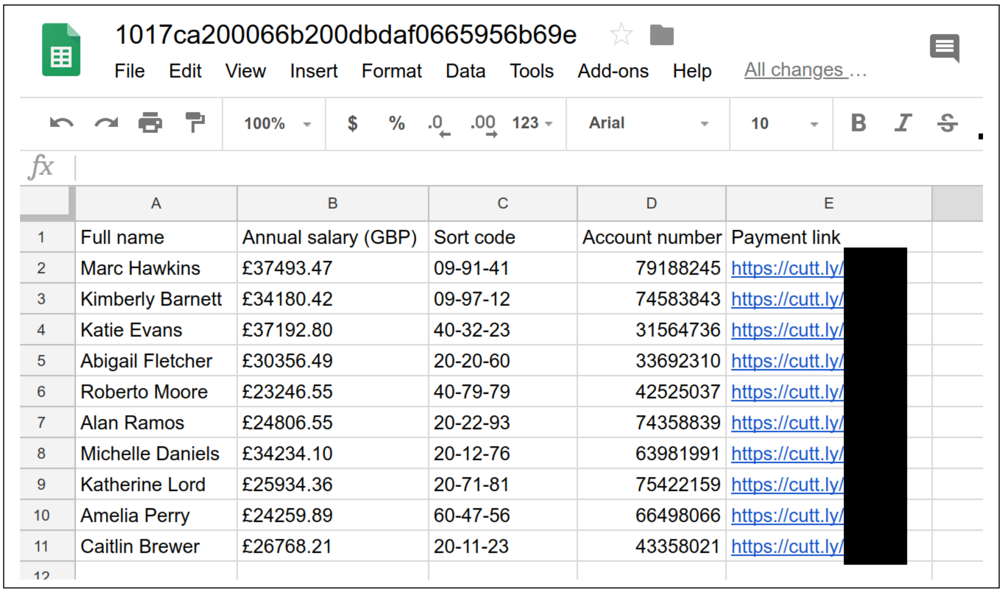
Fake data in cells. In total, we created 100 sheets for the experiments in this chapter. Prior to creating them, we manually created 10 Google accounts to host them, based on fake personal data that was generated using *Random User Generator*.[4] Note that a similar approach was employed to generate fake personal data for honey social accounts in Chapter 5. Next, we generated data to fill the rows and columns of honey sheets. Using the same Random User Generator, we created 1000 fake personal profiles. We then entered the first names and last names of those profiles in the honey sheets. We also included salary information sourced from `Monster.co.uk`,[5] a website that provides job search, career advice, and salary information to the general public. In half of the sheets, we inserted randomly-generated fake banking information and inserted fake cryptocurrency information in the other half. We did this to observe differences in accesses to documents that contain traditional banking information compared to documents that contain cryptocurrency information. To convert salary values from British Pound (GBP) to Bitcoin (XBT), we relied on `XE` tool[6] for currency conversion. We configured all sheets in a way that visitors can edit them without authentication, in other words, anyone in possession of a long link (described in Section 6.3.2) that points to a sheet can visit and edit such a sheet.

Fake banking information. As mentioned earlier, we included traditional banking

[4] https://randomuser.me/

[5] https://www.monster.co.uk/career-advice/article/uk-average-salary-graphs

[6] https://www.xe.com/currencyconverter/



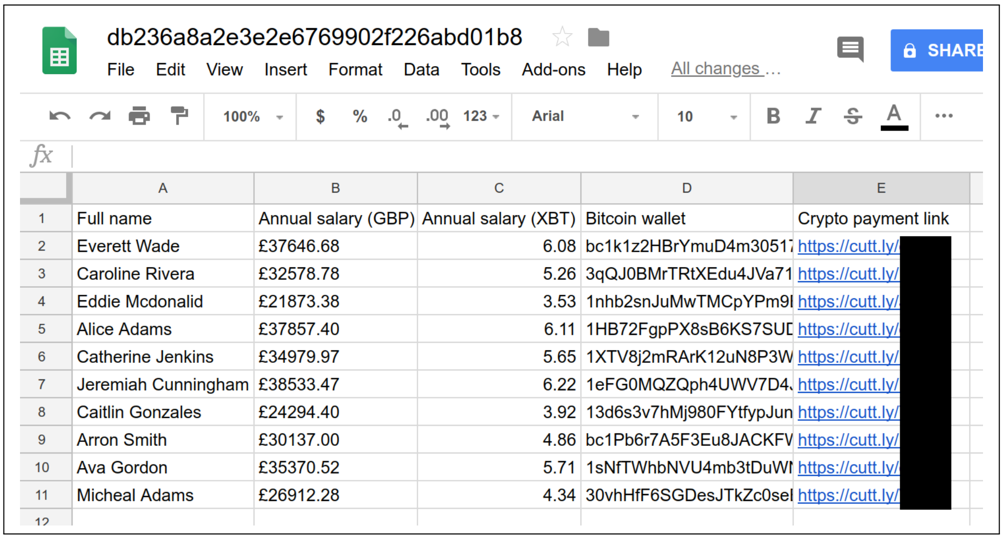
	A	B	C	D	E
1	Full name	Annual salary (GBP)	Sort code	Account number	Payment link
2	Marc Hawkins	£37493.47	09-91-41	79188245	https://cutt.ly/[redacted]
3	Kimberly Barnett	£34180.42	09-97-12	74583843	https://cutt.ly/[redacted]
4	Katie Evans	£37192.80	40-32-23	31564736	https://cutt.ly/[redacted]
5	Abigail Fletcher	£30356.49	20-20-60	33692310	https://cutt.ly/[redacted]
6	Roberto Moore	£23246.55	40-79-79	42525037	https://cutt.ly/[redacted]
7	Alan Ramos	£24806.55	20-22-93	74358839	https://cutt.ly/[redacted]
8	Michelle Daniels	£34234.10	20-12-76	63981991	https://cutt.ly/[redacted]
9	Katherine Lord	£25934.36	20-71-81	75422159	https://cutt.ly/[redacted]
10	Amelia Perry	£24259.89	60-47-56	66498066	https://cutt.ly/[redacted]
11	Caitlin Brewer	£26768.21	20-11-23	43358021	https://cutt.ly/[redacted]
12					

Figure 6.2: This sheet contains fake traditional bank payment information. Even though bank names were not explicitly included in the spreadsheets, account numbers and sort codes were generated to appear similar to real bank accounts. Short URLs (parts redacted) in the sheet point to nonexistent pages on websites of real banks.

information in half of the sheets. We selected 5 popular UK banks, namely HSBC, Lloyds Bank, Santander, Barclays, and Standard Chartered. We then generated fake sort codes and bank account numbers corresponding to those banks, following their conventions. For instance, HSBC sort codes have the form *40-xx-xx*, and Barclays *20-xx-xx*, where *xx* stands for any number between 11 and 99. Figure 6.2 shows an example sheet containing fake banking information.

Fake Bitcoin addresses. We needed fake but realistic-looking cryptocurrency information for the other half of honey sheets. To this end, we generated fake Bitcoin addresses following the specifications described on a Bitcoin wiki.[7] Specifications — most Bitcoin addresses comprise random digits and alphabets excluding characters that can result in visual ambiguity, for instance, digit zero (*0*) and uppercase letter *O*. The length of a Bitcoin address varies between 26 and 35 characters. In addition, they usually start with *1, 3,* or *bc1.* Following these specifications, we generated 500 fake Bitcoin addresses and included them in 50 sheets (Bitcoin sheets).

[7] https://en.bitcoin.it/wiki/Address



	A	B	C	D	E
1	Full name	Annual salary (GBP)	Annual salary (XBT)	Bitcoin wallet	Crypto payment link
2	Everett Wade	£37646.68	6.08	bc1k1z2HBrYmuD4m30517	https://cutt.ly/[redacted]
3	Caroline Rivera	£32578.78	5.26	3qQJ0BMrTRtXEdu4JVa71	https://cutt.ly/[redacted]
4	Eddie Mcdonalid	£21873.38	3.53	1nhb2snJuMwTMCpYPm9l	https://cutt.ly/[redacted]
5	Alice Adams	£37857.40	6.11	1HB72FgpPX8sB6KS7SUD	https://cutt.ly/[redacted]
6	Catherine Jenkins	£34979.97	5.65	1XTV8j2mRArK12uN8P3W	https://cutt.ly/[redacted]
7	Jeremiah Cunningham	£38533.47	6.22	1eFG0MQZQph4UWV7D4	https://cutt.ly/[redacted]
8	Caitlin Gonzales	£24294.40	3.92	13d6s3v7hMj980FYtfypJun	https://cutt.ly/[redacted]
9	Arron Smith	£30137.00	4.86	bc1Pb6r7A5F3Eu8JACKFV	https://cutt.ly/[redacted]
10	Ava Gordon	£35370.52	5.71	1sNfTWhbNVU4mb3tDuWl	https://cutt.ly/[redacted]
11	Micheal Adams	£26912.28	4.34	30vhHfF6SGDesJTkZc0sel	https://cutt.ly/[redacted]
12					

Figure 6.3: This sheet contains fake Bitcoin addresses in addition to other financial information. To convert salary values from British Pound (GBP) to Bitcoin (XBT), we relied on `XE` tool for currency conversion. Short URLs (parts redacted) point to nonexistent pages on real cryptocurrency exchanges.

Figure 6.3 shows an example sheet containing fake Bitcoin addresses.

Honey URLs. To observe if visitors to the sheets were going to carry out attacks on the "account owners" listed in the sheets, we included some fake payment URLs, which we refer to as *honey URLs*, in the sheets. These honey URLs, which point to nonexistent pages on bank websites and cryptocurrency exchanges, allow us to track clicks on them. To track clicks, we leveraged the functionality that *link shorteners* provide. Link shorteners are often used to generate short (and convenient and easy-to-use) URLs that contain fewer characters than the original URL, yet point to the original URL. Short URLs are handy on social networks since it is important to keep posts, messages, and URLs as short as possible. Examples of link shorteners include `bit.ly`, `goo.gl` (recently discontinued), and `cutt.ly`. By including short URLs (honey URLs) in the sheets instead of actual destination URLs, we achieve our goal of click tracking (via click analytics functionality provided by link shortening services) and hide the true destination of honey URLs. See Table 6.1 for illustrative examples of honey URLs. Visitors are compelled to click on honey URLs if they

Original URL	*Honey URL*
https://www.hsbc.co.uk/?passkey=d********	https://cutt.ly/C***
https://bittylicious.com/?auth=6********	https://cutt.ly/8***
http://[bouncy.domain]/banking-8102-f********	https://cutt.ly/J***
http://[bouncy.domain]/crypto-8102-1********	https://cutt.ly/w***

Table 6.1: Examples of honey URLs and the original URLs that were shortened to derive them (parts redacted). Note that click analytics data can be retrieved from all honey URLs, but additional information (such as IP addresses) can be retrieved only in the case of honey URLs that point to the bouncy web server (bouncy.domain stands in for the real domain that we used) explained in Section 6.4.2. Note that the URLs in this table are illustrative examples only, they are not literal examples of specific URLs that we included in the sheets.

wish to visit "payment pages" that they point to.[8] In the pilot experiment on honey sheets [62], goo.gl was used to shorten URLs and achieve click tracking. However, goo.gl was discontinued recently by Google, so we opted for cutt.ly instead. We briefly considered using bit.ly, another popular URL shortener, but it allows the external public to easily de-obfuscate the destination URL by simply appending a "+" to the short URL and visiting the resulting URL. In addition to revealing the destination URL, this also exposes details of the short URL's click analytics. Hence, we opted for cutt.ly whose destination URLs are harder to de-obfuscate and also provides private analytics. It provides a free click analytics dashboard and an API that allows easy download of click analytics data.

This concludes the process of creating honey sheets and adding fake financial data to them. Next, we describe the data collection infrastructure that we deployed to monitor honey sheets.

6.4.2 Data collection

In this section, we present the honeypot infrastructure that was deployed to collect data from honey sheets. We describe its components, what they do, and how they interact. Figure 6.4 shows an overview of the honeypot infrastructure and we describe its key components next.

Safehouse webmail account. Similar to the approach employed in [62], we in-

[8] Alternatively, to visit honey URLs, visitors can copy and paste them in a web browser.

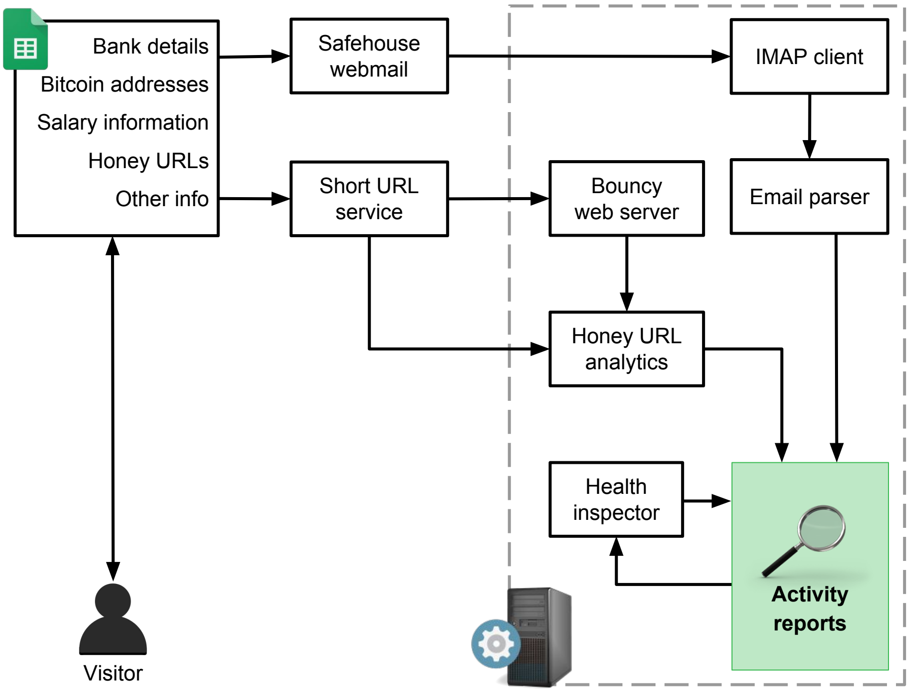


Figure 6.4: Our updated honey sheets infrastructure. Some components, especially the bouncy web server, have been updated since the experiments in previous work [62].

stalled scripts (Google Apps Script) in each sheet to report changes in the sheet back to us via a dedicated *safehouse webmail account*. Precisely, the scripts send notification emails containing periodic snapshots of sheets to the safehouse webmail account every 4 hours, by leveraging a time-driven Apps Script trigger. Also included in those notification emails are the edits made by visitors (changes in sheet state) between snapshots. We automatically record them by leveraging event-driven triggers and writing those changes to *Properties*, a persistent buffer for storing simple key-value pairwise data.[9] This approach (minimising the number of emails sent from sheets by relying on an implicit buffer) helped us to avoid script failure by staying under Google's quotas and limits (previously mentioned in Chapter 4, under Section 4.3.2). Next, we retrieve those notification emails via an email client (IMAP) and parse them to compare snapshots of sheets automatically. This allows us to record differences in snapshots and changes in sheets over time. For instance, differences in snapshots reveal modifications made to sheets by visitors (a modification involves changing the structure of a sheet or editing its cells).

Honey URL analytics. As mentioned in Section 6.4.1, short URLs in honey sheets provide information about clicks on them. This includes information about click origin (country), click count, and device information (that is, the device that was used to click on the link). We collected click analytics data once daily by leveraging `cutt.ly` analytics API (recall that we used `cutt.ly` link shortener service to create honey URLs), and stored it locally in JSON files for offline parsing.

Bouncy web server. `cutt.ly` analytics API provides useful click analytics data but does not reveal IP addresses of people that visit honey URLs. To overcome this limitation, we configured a third of the `cutt.ly`-generated honey URLs to point to a custom web server under our control, otherwise known as a *bouncy web server*. This web server enables us to record IP addresses and additional header information (which short URL analytics do not provide). On receiving a request for a web resource, the bouncy web server parses the request path and redirects the visitor to

[9] https://developers.google.com/apps-script/reference/properties/properties-service

a bank website if the request path contains the token "banking-8102," a cryptocurrency exchange website if the request path contains "crypto-8102," or `google.com` if the request path does not contain either of those tokens.[10] Table 6.1 shows example honey URLs that contain such tokens. The "bouncy" behaviour of the web server helps to keep up the appearance of visiting "payment links" and hides the existence of the bouncy web server.

Health inspector. To inspect the state of the honeypot system (to ensure that all components work as expected), we periodically run the health inspector to check that latest activity reports have been retrieved from the safehouse webmail account. It also examines click analytics data for recency. Out-of-date data indicates that one or more components of the honeypot infrastructure have failed. The health inspector reported sound system health throughout the experiments in this chapter.

6.4.3 Leaking long links

Previous work has shown that cybercriminals often post samples of their loot via online outlets usually to brag about their prowess or attract potential buyers [91]. Mimicking their modus operandi and following the general honey assets method proposed in Chapter 3, we leaked long links pointing to the sheets (not to be confused with honey URLs) on paste sites to lure cybercriminals to visit the sheets. To avoid ambiguity, note that long links, for leaking, are of the form `https://docs.google.com/spreadsheets/d/1AX4ZDODx3J...` while honey URLs, included in sheet data, look like `https://cutt.ly/B...`. Figure 6.1 shows an example of a long link that points to a sheet. We leaked long links via the same paste sites employed in Chapter 5 (listed in Table 6.2). Note that paste sites have successfully attracted visitors to honey assets in previous work [76, 62, 16] — this makes them suitable for the experiments in this chapter. Each long link was leaked along with a short description, for instance, "leaked payroll" or "bitcoin payment lists." Recall that we configured each sheet in a way that anyone can access and edit it, provided they

[10] In case the reader wonders what "8102" stands for in request paths, it has no special significance. It is simply year "2018" written backwards.

Table 6.2: To lure visitors to honey sheets, we leaked long links pointing to the sheets through paste sites on the Surface Web and the Dark Web. We chose these paste sites because they allow public pastes and have successfully attracted visitors to honey assets in previous work [76, 62, 16].

Name	*Type*	*URL*
Pastebin	Surface Web	`https://pastebin.com/`
Paste.org.ru	Surface Web	`http://paste.org.ru/`
Stronghold	Dark Web (via TOR)	`http://nzxj65x32vh2fkhk.onion/`

know the long link that points to it.

Prior to leaking the 100 long links, we divided them into five chunks, each chunk comprising 20 long links. We leaked all chunks twice daily to ensure good temporal coverage on paste sites, thus compensating for timezone differences among visitors to the paste sites. We also randomised the order of links in each chunk prior to leaking, thus ensuring that each long link had a fair chance of being visited. After leaking the long links, we recorded accesses to sheets and tracked clicks on honey URLs inside them.

6.4.4 Threats to validity

We acknowledge that there are some factors that may affect the validity of our findings. First, our honey sheet data comprises fake financial data, including fake Bitcoin addresses, which may be obvious under close scrutiny, and can possibly influence the behaviour of visitors. Second, our honey URLs (embedded in sheet data) are short URLs, and short URLs are generally treated with suspicion. This may negatively affect the perception of visitors to honey sheets. Third, we leaked long links pointing to the sheets through paste sites only. Our findings may not be representative of malicious activity in cloud documents stolen via other outlets, for instance, malware-laden endpoints or underground forums.

Finally, there is also the possibility that automated tools (crawlers) visited the long links in addition to human visitors. This may affect the validity of our findings. To mitigate this risk in future work, it is possible to incorporate an additional CAPTCHA-like step in the process of accessing the sheets to ensure that only manual accesses by humans pass through. This can be achieved by leaking links that point to a web domain under our control which will serve up an interstitial page containing the CAPTCHA. If a visitor passes the CAPTCHA (and thus prove they are indeed human), they will be redirected to the sheet that they were trying to access in the first place. However, this approach may discourage visitors from proceeding because of the increased cognitive workload that CAPTCHA solving involves.

Despite these concerns, this chapter offers insights into malicious activity in compromised cloud documents and presents new ways to study such activity.

6.4.5 Ethics

The experiments in this chapter involve deceiving cybercriminals into interacting with cloud documents. In line with standard ethical practices, we took the following precautions. First, we used fake financial data (randomly generated) in the sheets. Thus, we ensured that no real person or account was harmed in our experiments. Second, to avoid spamming other accounts, we did not leak credentials of the Google accounts that hosted our honey sheets. We only leaked the long links that point to honey sheets, thus limiting the possible harm that our experiments may cause otherwise. Third, we obtained approval from UCL's ethics committee prior to running experiments.

6.5 Data analysis

In this section, we present an overview of our observations during experiments and detailed measurements of visitor activity in honey sheets.

6.5.1 Activity overview

We conducted experiments from 11th July, 2018 until 14th August, 2018. During this period, 98 sheets were opened 235 times. These sheets comprise 48 sheets containing banking information and 50 sheets containing cryptocurrency information. We recorded 38 modification events during which 7 sheets were modified by visitors. We observed 219 clicks on honey URLs. Those clicks originated from 30 countries.

6.5.2 What is a sheet access?

Unlike the studies presented in Chapter 4 (webmail accounts) and Chapter 5 (social accounts) in which an access requires authenticating to an account using leaked username and password information, we present a different definition of an access in this chapter. By design, accessing any of our honey sheets does not require authentication. Instead, the visitor requires knowledge of a long link that points to a sheet (and we expose those long links by leaking them on paste sites, as explained in Section 6.4.3). This makes it easier for visitors to access the honey sheets — all they need to do is visit the long link. Hence, in this chapter, we define an access as a file open event (in other words, a sheet open event). The downside of this unauthenticated approach is that there are no strong unique identifiers of accesses (in other words, no cookies, unlike Chapters 4 and 5 in which we recorded and analysed cookies). This also implies that we are unable to build a taxonomy of accesses in this chapter, since doing so requires cookies. Finally, it is important to note that spreadsheets present fewer functionalities than webmail accounts and social accounts (hence, fewer measurements are possible). Nevertheless, we successfully analysed open events and modification events, and tracked clicks on honey URLs in the sheets. Next, we present our findings.

6.5.3 Timing of activity in sheets

Leak to first access. First, we set out to understand how long it took for visitors to access the sheets after we leaked long links pointing to them. Let us denote the time of first leak as t_{leak}. For each opened sheet, we record the time of its first open event (first access) as t_0 and compute the time lag between leak and first access as $t_0 - t_{leak}$. Figure 6.5 shows a CDF of time lags. Less than 10% of opened sheets were visited within the first 22 hours since first leak. However, accesses increased rapidly afterwards — by the 25th hour since first leak, 80% of the sheets had been opened. It is possible that the initial time lags between first leak and first accesses were due to reluctance of visitors to visit links, since links can be potentially malicious, generally speaking. However, it is hard to explain the rapid uptake that started around the 23rd hour since first leak.

Timeline of accesses. Next, we set out to understand the spatial patterns (with respect to time) of all accesses during experiments. 98 sheets received 235 accesses. These comprise 48 bank sheets and 50 cryptocurrency sheets. Let us denote the time of a given access as t_a and the time of first leak as t_{leak}. For each access, we computed its relative access time as $t_a - t_{leak}$. We then plotted a timeline of accesses (see Figure 6.6), with the time of first leak t_{leak} as the reference point. Figure 6.6 corroborates our previous findings in Figure 6.5 — it shows sparse accesses during the first day since the initial leak. From the beginning of the second day, it shows a sharp increase in accesses to bank sheets and cryptocurrency sheets. Figure 6.6 also shows that accesses to cryptocurrency sheets spanned a longer time period than accesses to bank sheets — the last access we recorded in a bank sheet was on the 25th day after first leak, whereas we observed accesses in cryptocurrency sheets afterwards. Next, we study the modifications that visitors made to some of the honey sheets.

6.5.4 Modifications and edits in sheets

We observed 38 modifications in 7 cryptocurrency sheets. No bank sheet was modified. A closer look at the modified sheets revealed that most of the modifications were recorded when visitors resized columns in sheets (changes made to sheet structure are recorded as modifications). This happened in cryptocurrency sheets because visitors wanted to view Bitcoin addresses, which are long strings, partly obscured in the default states of the cryptocurrency sheets. See Figure 6.3 for an example cryptocurrency sheet containing Bitcoin addresses — note that the addresses were not displayed in full. Interested visitors thus had to resize that column

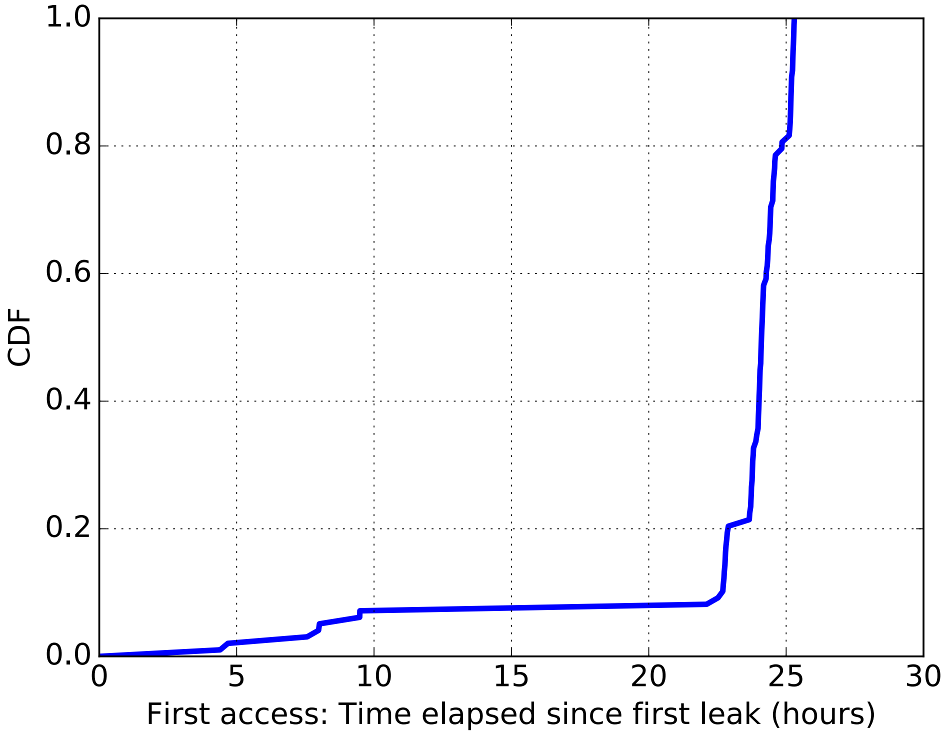


Figure 6.5: CDF of time lags between first leak and first access. Less than 10% of the opened sheets were visited within the first 22 hours since first leak, indicating initial reluctance to visit the sheets. However, accesses increased rapidly afterwards. By the 25th hour since first leak, 80% of the sheets had been opened.

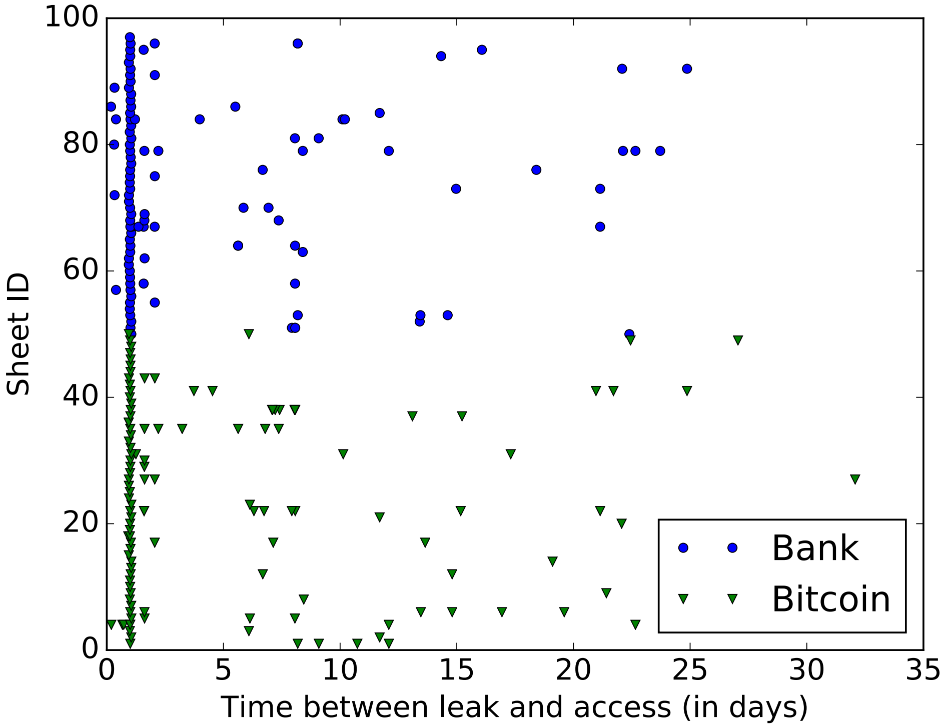


Figure 6.6: Timeline of accesses. 98 sheets received 235 accesses. These comprise 48 bank sheets (“Bank”) and 50 cryptocurrency sheets (“Bitcoin”). Note that accesses to cryptocurrency sheets spanned a longer time period than accesses to bank sheets.

for a better view.

Next, we studied modifications that resulted in changes to values of cells in sheets (otherwise known as *edits*). We observed that a Bitcoin address in cell D4 of a cryptocurrency sheet was replaced with another Bitcoin address. We looked up the new Bitcoin address on a Bitcoin address verification tool (`blockchain.info`), but it returned no result. We also looked up our list of fake Bitcoin addresses to see if it was copied from another cryptocurrency sheet and this lookup also yielded no result. This indicates that the Bitcoin address entered by the visitor was either a yet-to-be-used Bitcoin address that belonged to them (with intent to commit fraud by receiving payments meant for the original recipient listed on the compromised sheet), or a fake Bitcoin address made up by them.

In another cryptocurrency sheet, we observed that one of its records (fields B10 – E10) had been replaced with values that were exactly the same record. In other words, cell edits were recorded with no changes in values. This indicates that a visitor (accidentally) “cut” the original values and pasted them back in the sheet. On the same sheet, the next record was modified similarly, with most values intact, except for the Bitcoin address field. The Bitcoin address of that record was replaced with a different string[11] that did not fit the specification of Bitcoin addresses and was also absent from the list of fake Bitcoin addresses we initially generated. We observed another edit in a separate cryptocurrency sheet in which the Bitcoin address of one of its records was replaced by a copy of the string mentioned previously. This indicates that the same visitor modified both sheets (by pasting that string in both sheets).

In summary, the majority of sheet modifications comprised column resizing actions by visitors, while actual edits involved changes to Bitcoin addresses. Next, we study the patterns of clicks on honey URLs within the sheets.

[11] Pasted string — `qzpweklwh85u0h2x44ffv4tsfhxww96v8c7kylnwyu`. We are yet to figure out what it stands for.

6.5.5 Click activity

Recall that we included two types of honey URLs in the honey sheets, namely bank URLs and cryptocurrency URLs. In this section, we present measurements of clicks on those URLs. We recorded 219 clicks on honey URLs comprising 135 clicks on bank URLs and 84 clicks on cryptocurrency URLs. Those clicks originated from 30 countries. We present a detailed summary of click counts in Table 6.3.

Click counts. We wanted to observe differences in clicks on bank URLs and cryptocurrency URLs. To this end, we counted those clicks, by link type, and plotted CDFs of click counts. Contrary to our expectations, honey URLs of the bank type consistently received more clicks than honey URLs of the cryptocurrency type. We expected the opposite to happen, given the recent surge in interest of the general public in cryptocurrencies and blockchain technologies. The bank link with the highest click count recorded 18 clicks while the cryptocurrency link with the highest click count recorded 14 counts, as Figure 6.7 shows.

Statistical test. To test the statistical significance of differences in click counts by link type, we relied on the two-sided Kolmogorov-Smirnov (KS) test to examine the CDFs in Figure 6.7. The null hypothesis states that both samples under examination belong to identical statistical distributions. The output of the test is a KS statistic and p-value. A small KS statistic or high p-value shows that we cannot reject the null hypothesis. The KS test returned an inconclusive result (statistic=0.4667, p-value=0.0515).

Click locations. During the analysis of `cutt.ly` click analytics data (on honey URLs), we collated a list of countries that clicks originated from. We also carried out geolocation (country resolution) of IP addresses that visited our bouncy web server. We used *IP-API*,[12] an IP geolocation service that provides timezone and location information for IP addresses, to achieve this. We then plotted the resulting locations, comprising 30 countries, on a world map as shown in Figure 6.8. As the map shows, most of the countries are located in Europe. It is possible that some

[12] http://ip-api.com

Table 6.3: Summary of clicks on honey URLs. Direct honey URLs lead visitors directly to the destination URL (bank or cryptocurrency page) while bouncy URLs surreptitiously route visitors through our bouncy web server before redirecting them to the destination URL. Surprisingly, bank URLs received more clicks than cryptocurrency URLs despite the recent interest of the general public in cryptocurrencies and blockchain technologies.

Type of honey URL	*Click count*
Direct bank	98
Bouncy bank	37
Direct Bitcoin	69
Bouncy Bitcoin	15
Total	**219**

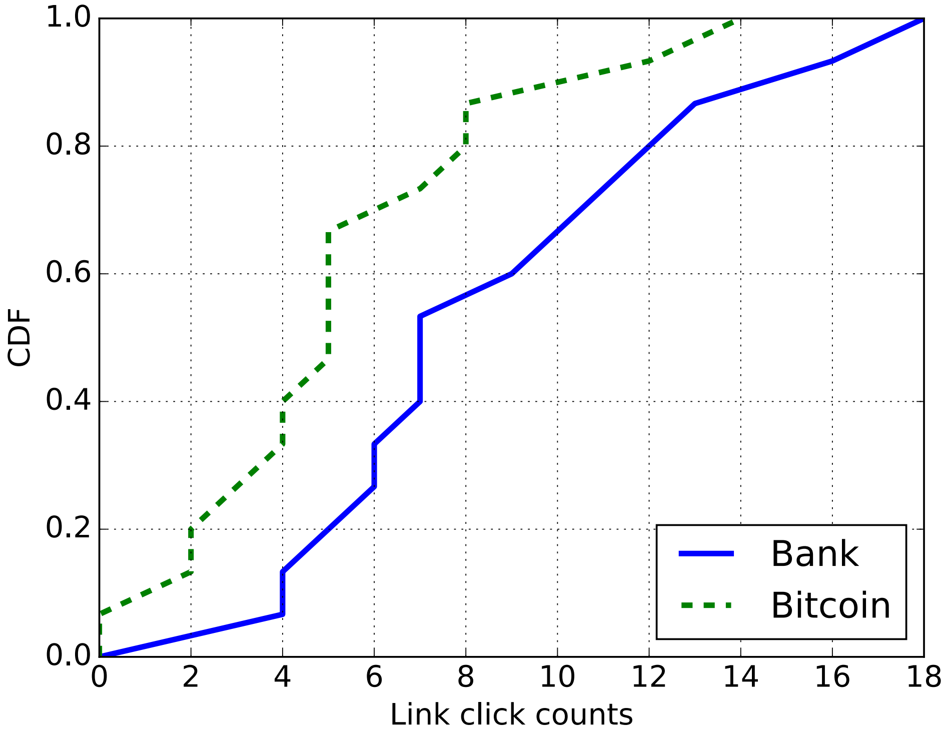


Figure 6.7: URL click counts. Contrary to our expectations, honey URLs of the bank type received more clicks than cryptocurrency honey URLs. The bank link with the highest click count recorded 18 clicks while the cryptocurrency link with the highest click count recorded 14 counts.

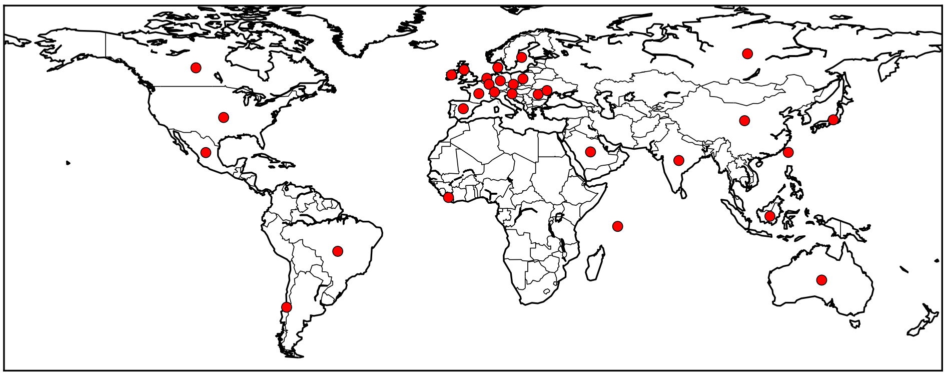


Figure 6.8: Origins of clicks on honey URLs in sheets. These locations comprise a comprehensive list of countries found during the parsing of click analytics data and IP geolocation. Most of the countries are located in Europe. It is possible that some visitors connected to sheets and clicked on honey URLs via proxies or VPNs.

visitors connected to the sheets and clicked on honey URLs via proxies or VPNs. We found some TOR exit nodes (see Section 6.5.6) among the IP addresses that visited the bouncy web server via honey URLs.

The IP geolocation process yielded 20 countries. Intuitively, these countries should be a subset of the list of countries recorded by the `cutt.ly` click analytics tool. However, this was not entirely the case. We found 14 common countries (that is, they exist in click analytics and IP geolocation datasets), while 6 countries in the IP geolocation dataset were absent from the click analytics dataset. This shows that some visitors visited honey URLs, recorded the (de-obfuscated) destination URLs, and directly visited those destination URLs later.

6.5.6 System configuration of accesses

In this section, we study the devices that visitors used while clicking on honey URLs in sheets. We discuss their IP addresses, browsers, and operating systems.

IP addresses and TOR exit nodes. Recall that a subset of honey URLs point to our bouncy web server which allows us to collect IP addresses of visitors clicking on them, in addition to click analytics. We recorded 35 IP addresses that visited the bouncy web server from 20 countries. 12 of the IP addresses were TOR exit nodes.

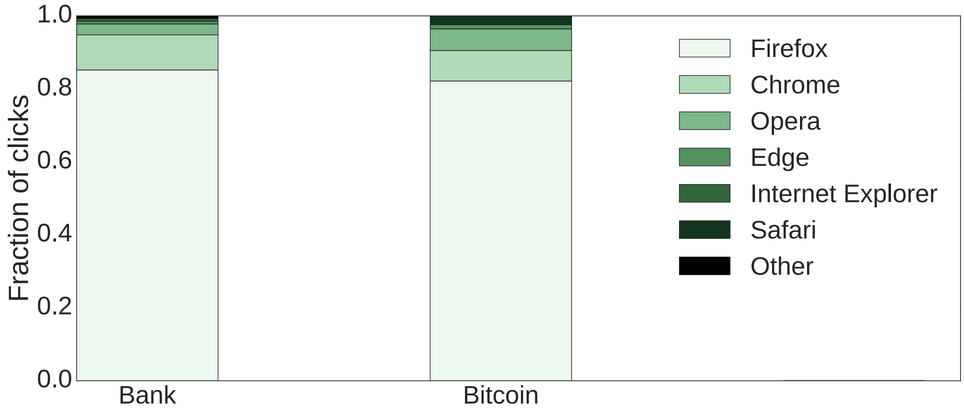


Figure 6.9: Distribution of browsers that visitors used while clicking on honey URLs in the sheets. Firefox leads the pack, with more than 80% share in clicks on bank and cryptocurrency URLs. We also observed clicks from Chrome, Opera, and other browsers.

Note that this is only a subset of the IP addresses that visited the honey sheets. In other words, not all visitors to honey sheets click on honey URLs in the sheets. Also, only a third of our honey URLs, the ones that point to the bouncy web server, can track IP addresses. Hence we have a partial view of IP addresses. Nevertheless, it is surprising that 34% of the recorded IP addresses were TOR exit nodes. It shows that one out of every three persons that visited our honey URLs covered their tracks while doing so, by visiting via the TOR network.

Browsers. We wanted to understand the distribution of browsers that visitors used to connect to honey URLs in the sheets. To this end, we extracted browser information from click analytics data and grouped browser-clicks by URL type. We computed the percentages of clicks made via the different browsers observed. Figure 6.9 shows the distribution of browsers that were used to visit honey URLs. Visitors that clicked on honey URLs had an unusual preference for Firefox — the top browser responsible for more than 80% of clicks on bank and cryptocurrency URLs. We also observed clicks from Chrome, Opera, and other browsers.

Operating Systems. We wanted to know the operating systems of devices that connected to honey URLs. We extracted this information from the click analytics dataset. Visitors that clicked on honey URLs had a preference for Windows de-

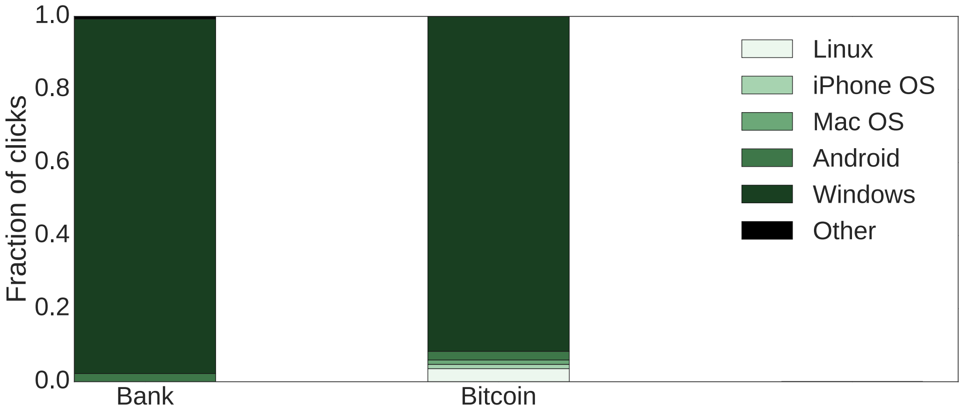


Figure 6.10: Visitors to honey URLs had a preference for Windows devices in clicks on honey URLs. In Bitcoin URL clicks, even though Windows devices dominate, we observe a slightly wider range of operating systems compared to bank URLs. Bitcoin URLs also attract more handheld devices (by percentage share) than bank URLs, indicating a more diverse set of visitors.

vices, in bank and cryptocurrency honey URL clicks, as shown in Figure 6.10. In cryptocurrency URL clicks, even though Windows devices dominate, we observe a slightly wider range of operating systems than devices that clicked on bank URLs. In both URL types, we observed a small fraction of clicks from Android devices. Cryptocurrency URLs recorded a tiny fraction of visits from iPhones and Linux devices, both absent from clicks on bank URLs. This indicates that cryptocurrency URLs attracted a slightly more diverse set of visitors than bank URLs.

6.6 Summary

In this chapter, we studied the activity of visitors in compromised cloud documents. First, we presented a method for instrumenting and deploying honey cloud documents. Next, we deployed 100 payroll sheets comprising 1000 financial records. In half of the sheets, we included fake banking information, and included fake cryptocurrency information in the other half, to observe differences in accesses and activity in the sheets. We included honey URLs in the sheets to track clicks on them. We ran experiments for a month and recorded 235 accesses across 98 sheets. We observed 38 modifications in 7 sheets. Finally, we presented detailed measurements and analysis of devices that visited honey URLs, with emphasis on IP addresses, browsers, and operating systems. In summary, we have presented a comprehensive picture of what happens to compromised spreadsheets. The findings presented in this chapter will be useful for other researchers seeking to understand what happens to stolen cloud documents and providers of cloud services seeking to understand ways to secure user accounts and assets.

Chapter 7

Discussions and Conclusion

In Chapters 4 (webmail accounts), 5 (social accounts), and 6 (cloud documents), we studied what happens to compromised online accounts and documents. In this chapter, we position our findings in a broader context. First, we discuss what our findings imply for current detection and mitigation techniques against malicious activity in online accounts. Second, we discuss differences and similarities in activity across the online services under study. Third, we discuss lessons learned from the credential dissemination vectors that were employed in this thesis. We then highlight some implementation-specific limitations of honeypot instances that were deployed in this thesis. Finally, we present some ideas for future work.

7.1 Implications for detection systems

We made multiple findings that provide the research community with a better understanding of what happens when online accounts are compromised. In Chapter 4 (webmail accounts), we discovered that attackers that know the locations of webmail account owners tend to connect from places that are closer to those locations. We infer that this is an attempt to evade current security mechanisms employed by online services to discover suspicious logins. Such systems often rely on the origin of logins to assess how suspicious those login attempts are. Our findings show that there is an arms race going on, with attackers attempting to actively evade the location-based anomaly detection systems employed by Google and other online services. We also observed that a considerable number of accesses to webmail accounts, social accounts, and cloud documents were routed through TOR exit nodes, so it is hard to determine the exact origins of accesses, since TOR network masks true origins of accesses. This problem shows the necessity of defence-in-depth approaches in protecting online services, in which multiple layers of detection systems are deployed simultaneously to identify and block criminals.

Potential improvements. Despite confirming evasion techniques in use by cybercriminals, our experiments also highlighted interesting behaviours that can be used to develop or improve systems to detect malicious activity. For example, our observations about the words searched for by cybercriminals show that behavioural modelling could work in identifying anomalous behaviour in online accounts (Chapters 4 and 5). Anomaly detection systems could be trained adaptively on words being searched for by the legitimate account owner during regular use. A deviation of future search behaviour would then be flagged as anomalous, indicating that the account may have been compromised. Online service providers, when tuning detection systems, can also apply our findings on differences in search terms — in Chapter 4 (webmail accounts), we observed that search terms mostly contained financial/sensitive information while search terms recorded in Chapter 5 (social accounts) indicated less interest in financial information.[1] Similarly, anomaly detection systems could be trained by building statistical models on the durations of accesses (measured in Chapters 4 and 5). Deviations from known access patterns could then be flagged as anomalous (potentially malicious) and they would trigger additional automatic checks and reviews by human operators. In other words, it is possible to develop and train tools based on machine learning methods that do not require balanced training datasets of positive and negative examples (one-class Support Vector Machines, for instance) on “normal” document activity or account accesses. Malicious accesses will likely deviate from normal activity and will thus be flagged as such. Our datasets, collected during experiments, are insufficient for this purpose

[1] Caveat — We encountered some limitations in our analysis of search terms in Chapter 4.

(that is, they are not large enough to train machine learning tools). Online services with direct access to much larger datasets of account accesses and activity, for instance, Google and Facebook, are in a better position to develop such tools.

7.2 Differences across services under study

We observed differences in external usage of honey assets. For instance, in Chapter 4 (webmail accounts), we observed the use of a honey webmail account as a registration address during account creation on a carding forum (financial usage). Similarly, in Chapter 5, we observed the use of two honey social accounts for authentication to Instagram (social usage). This shows differences in motivation of cybercriminals that visit different types of online accounts — webmail account visitors appear to be more interested in financial usage of honey assets than visitors to social accounts.[2] Overall, the use of honey assets on external platforms indicates the possibility of multi-platform attacks in which cybercriminals compromise accounts on an online service and use those accounts to carry out further illicit activity on other services.

As we discussed in Chapter 5 (social accounts), differences in account demographics influence malicious activity in accounts. For instance, we observed slight differences in recorded activity across different age ranges and genders. Anomaly detection systems could be trained to be sensitive to differences in account activity per demographic attribute. These detection systems could be trained to be more sensitive to chatty and emotional behaviour in teen accounts more than adult accounts, for instance. Similarly, a related study (not included in this thesis) shows that language differences in account content affects the behaviour of cybercriminals in webmail accounts [16]. This knowledge, along with other key findings presented in this thesis, could be applied when sourcing and partitioning training and test data for automatic detection systems.

In cloud documents (Chapter 6), we observed differences in document modifi-

[2] We observed a similar distinction in search terms across webmail and social accounts, as mentioned earlier.

cations, depending on the content of the documents. Particularly, documents that contained cryptocurrency information were subject to more modifications than documents containing banking information, despite receiving fewer accesses than sheets that contained bank information. Similarly, we observed differences in URL clicking behaviour across different types of URLs in documents. This knowledge can be used during the development and training of detection systems to protect cloud documents. Such detection systems could be built to adapt their statistical models depending on document type and content.

7.3 Common trends across services

We also observed common trends across online services. First, we observed some usage of TOR browser/network in accesses to honey assets in all platforms under study (webmail, social, and cloud document services). As discussed previously, this makes it difficult to determine the true origin of accesses to online accounts. Second, we recorded account modification or defacement activity on all the platforms under study. In webmail accounts, for instance, we observed many abandoned email drafts. In social accounts, we recorded bizarre timeline posts and private messages. In cloud documents, we observed an instance in which a meaningless string was pasted in two sheets, among other modifications. Such behaviour likely deviates from regular everyday use of online accounts and could potentially help in identifying anomalous behaviour in such accounts.

On a related note, we observed potentially destructive behaviour across all types of honey assets under study. In webmail and social accounts, for instance, we observed hijacking attacks (password changes) and deletion of friends (along with addition of new friends). In cloud documents, we observed “cut-and-paste” activity on sheet content. Recall that there was no way to observe hijacking incidents in cloud documents because our experimental system intentionally allowed unauthenticated access to documents. Even though such actions occur during benign account usage, a surge in potentially destructive activity could be flagged as an anomaly by detection systems.

Finally, we observed search activity in webmail and social honey assets.[3] This shows that searching for information within an account is yet another common activity of criminals that connect to stolen accounts on various platforms. This also corroborates previous work [25] and shows that we should pay more attention to anomalous search patterns in online accounts in the race for better detection and mitigation systems to make accounts safer for users.

7.4 Lessons learned from dissemination vectors

Recall that we leveraged multiple dissemination vectors, otherwise known as outlets for credential leaks, namely paste sites, underground forums, and malware. In this section, we summarise some general lessons we learned while using those dissemination vectors.

Paste sites and others. During the webmail study (Chapter 4), we leveraged paste sites and underground forums on the Surface Web and found that paste sites resulted in higher *yield* than underground forums (we controlled for leak outlets). Hence, for high yield (more accesses from potential criminals), it is beneficial to focus more on paste sites than other outlets. Note that we excluded malware leak outlets from this discussion because they do not fit into the Surface Web/Dark Web dichotomy.

Surface Web and Dark Web outlets. Unlike Chapter 4 (webmail study) in which we used Surface Web and malware outlets only, we leveraged a combination of Surface and Dark Web outlets (paste sites) in Chapters 5 (social accounts) and 6 (cloud documents). This combination of Surface Web and Dark Web outlets enabled us to record more accesses than we would have recorded if we had relied on the Surface Web only. This approach also helps to improve the diversity of visitors (potential criminals) that access honey assets.

[3] Even though sheets present search functionality, we do not have a way to record search logs in them yet.

Malware outlets. It takes a skilled criminal with malicious intent to operate information-stealing malware infrastructure. The process extends beyond mere curiosity, for instance, just testing to see if leaked accounts are real or not – it involves actively compromising victim endpoint devices and covertly stealing sensitive information from them. Hence, when compared to paste sites and underground forums that attract a combination of potentially benign visitors, potential criminals, and other interested parties, malware outlets comprise dissemination vectors that are founded on real criminal intent and they likely provide the "purest" view to malicious activity in online accounts. Hence, it may be beneficial to focus more on malware outlets in future work seeking to understand sophisticated cybercriminal operations. However, the main downside of studying malware outlets is the difficulty of obtaining live malware samples with active C&C infrastructure (malware outlets are fickle, as we learned during the webmail study). In addition, advanced information-stealing malware, for instance Dridex, often incorporate evasive mechanisms that prevent them from executing in sandboxed environments. Despite these challenges, malware outlets hold a lot of promise in shining light on criminal activity in compromised accounts.

7.5 Implementation-specific limitations

In Chapter 3, we discussed our honey assets method, ARMER requirements for honeypots, and our honeypot development life cycle approach. We also discussed its limitations. In this section, we discuss the limitations of specific honeypot instances that were deployed during the experiments in this thesis.

We were able to leak honey assets on a few outlets, namely paste sites, underground forums, and malware. In particular, we could only target underground forums that were open to the public and for which registration was free. In Chapter 4 (webmail accounts), we could not study recent families of information-stealing malware, such as Dridex, because they would not execute in our virtual environment (evasive malware). Attackers could find the scripts we hid in the webmail accounts and remove them, and make it impossible for us to monitor their activity. This is an intrinsic limitation of the webmail honeypot infrastructure, but this limitation does not apply to the Facebook honeypot system in Chapter 5, since we did not have to hide scripts in Facebook accounts. In principle, studies similar to ours could be performed by online service providers themselves, for instance, Google or Facebook. By having access to the full logs of their systems, such entities would have no need to set up monitoring scripts and it would be impossible for attackers to evade their scrutiny.

Chapter 4 revealed that the vast majority of accesses to our webmail accounts, especially the ones leaked via underground forums and malware, resulted in no action ("curious" visitors). Our honey accounts were possibly not convincing enough for them to take action after logging in. This can affect the ecological validity of our findings. Alternatively, it is possible that potential criminals that use underground forums and malware are generally less active than those that use paste sites.

In Chapter 4 (webmail accounts), while evaluating what cybercriminals were looking for in honey accounts, we were able to observe the emails that they were interested in, not everything they searched for. This happened because we did not have access to search logs of the webmail accounts under study. However, in Chapter 5 (social accounts), we did not face this limitation. We had access to the full search logs of Facebook accounts that were deployed during experiments and recorded exact search terms in them.

In Chapter 6, we had limited visibility into the sheets because of unauthenticated accesses (by design). As a result, we were able to record only a subset of IP addresses that visited the documents by recording the ones that clicked on honey URLs in the documents. Similarly, we succeeded in recording times of accesses to cloud documents, but not the durations of accesses, unlike Chapters 4 (webmail accounts) and Chapter 5 (social accounts) in which we recorded access durations, in addition to IP addresses.

Across all services under study, we leaked credentials via public outlets, with the exception of the subset of webmail credentials that were leaked through information-stealing malware, implying that they ended up in private lists of the actors running those malware C&C servers. In other words, all *public* leaks, by design, were known to multiple potential criminals simultaneously. There is the possibility that the accessibility of credentials to multiple potential criminals diminished the perceived value of those accounts. This may influence their willingness to carry out the usual actions they would have carried out if the accounts were privately held. In view of this, the subset of webmail credentials that were leaked via malware likely present better ecological validity (since they were privately held) than the remaining sets of leaked credentials.

Finally, recall that we do not have control of the online services that host our honey assets. Hence, the installation of scripts inside honey assets that require them (see Chapters 4 and 6) must be done perfectly prior to experiments. This is because it is hard to update scripts in honey assets — such updates must be carried out and tested manually across all honey assets. This is hard to do, but even harder once experiments are in motion (such updates, if carried out during experiments, may taint the findings). Hence, we carried out rigorous testing on honey assets prior to experiments to avoid having to update them during experiments.

Despite these limitations, we have succeeded in shedding light on malicious activity in online accounts and cloud documents. Our honeypots also provide new ways for researchers and service providers to carry out related work and add to the knowledge of the security community.

7.6 Future work

In the future, we plan to continue exploring the ecosystem of stolen accounts and gaining a better understanding of the underground economy surrounding them. We will explore ways to make honey assets more believable, to attract more cybercriminals and keep them engaged. We intend to set up additional scenarios, such as studying attackers who have a specific motivation, for example, compromising accounts that belong to political activists. We also intend to carry out further studies on the impact of other demographic attributes (including employment status, religious affiliation, and political affiliation, among others) of online accounts on the behaviour of cybercriminals that gain illicit access to them. It is also possible to deploy social honeypots to understand demographic risk factors that influence cyberbullying incidents. These will provide comprehensive insights into attackers' motivations and resulting activity.

7.7 Conclusion

In this thesis, we reviewed existing work on malicious online activity, developed novel methods to study malicious activity in compromised online accounts, carried out experiments, and presented our findings. We discovered attempts by cybercriminals to evade existing defence systems. We also observed patterns of accesses and behaviour that could be used to characterise malicious activity to help improve existing defence systems. We discussed the implications of our findings, especially for online service providers, and bridged the research gap we observed prior to the work in this thesis. Finally, we discussed the limitations of our work and highlighted potential future work. Parts of the work in this thesis have been peer-reviewed and presented in top conferences and workshops. In addition, some parts have received considerable press coverage. This shows that our work has contributed to the research community and increased the awareness of the general public about compromised online accounts. Overall, this will lead to safer online activity for everyone.

Bibliography

[1] Sherly Abraham and InduShobha Chengalur-Smith. An overview of social engineering malware: Trends, tactics, and implications. *Technology in Society*, 32(3):183–196, 2010.

[2] Stefan Achleitner, Thomas F. La Porta, Patrick D. McDaniel, Shridatt Sugrim, Srikanth V. Krishnamurthy, and Ritu Chadha. Cyber Deception: Virtual Networks to Defend Insider Reconnaissance. In *ACM CCS Workshop on Managing Insider Security Threats (MIST@CCS)*, 2016.

[3] Sadia Afroz, Aylin Caliskan Islam, Ariel Stolerman, Rachel Greenstadt, and Damon McCoy. Doppelgänger finder: Taking stylometry to the underground. In *IEEE Symposium on Security and Privacy*, 2014.

[4] Ali Al Mazari. Cyber-bullying taxonomies: Definition, forms, consequences and mitigation strategies. In *Conference on Computer Science and Information Technology (CSIT)*, pages 126–133, 2013.

[5] Eric Alata, Vincent Nicomette, Mohamed Kaâniche, Marc Dacier, and Matthieu Herrb. Lessons learned from the deployment of a high-interaction honeypot. In *European Dependable Computing Conference (EDCC)*, 2006.

[6] David Alvarez-Melis and Martin Saveski. Topic Modeling in Twitter: Aggregating Tweets by Conversations. In *AAAI Conference on Weblogs and Social Media (ICWSM)*, 2016.

[7] Ross Anderson, Chris Barton, Rainer Böhme, Richard Clayton, Michel JG Van Eeten, Michael Levi, Tyler Moore, and Stefan Savage. Measuring the

cost of cybercrime. In *The economics of information security and privacy*, pages 265–300. Springer, 2013.

[8] Theodore W. Anderson and Donald A. Darling. Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *The Annals of Mathematical Statistics*, 1952.

[9] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the Mirai Botnet. In *USENIX Security Symposium*, 2017.

[10] Paul Baecher, Markus Koetter, Thorsten Holz, Maximillian Dornseif, and Felix Freiling. The nepenthes platform: An efficient approach to collect malware. In *Symposium on Recent Advances in Intrusion Detection (RAID)*, 2006.

[11] Julia Barlińska, Anna Szuster, and Mikołaj Winiewski. Cyberbullying among adolescent bystanders: Role of the communication medium, form of violence, and empathy. *Journal of Community & Applied Social Psychology*, 23(1):37–51, 2013.

[12] Timothy Barron and Nick Nikiforakis. Picky Attackers: Quantifying the Role of System Properties on Intruder Behavior. In *Annual Computer Security Applications Conference (ACSAC)*, 2017.

[13] Fabrício Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. Detecting Spammers on Twitter. In *Conference on Email and Anti-Spam (CEAS)*, 2010.

[14] Fabrício Benevenuto, Tiago Rodrigues, Virgílio Almeida, Jussara Almeida, and Marcos Gonçalves. Detecting spammers and content promoters in online video social networks. In *ACM conference on Research and development in information retrieval*, 2009.

[15] Maya Bercovitch, Meir Renford, Lior Hasson, Asaf Shabtai, Lior Rokach, and Yuval Elovici. HoneyGen: An automated honeytokens generator. In *IEEE Conference on Intelligence and Security Informatics (ISI)*, 2011.

[16] Emeric Bernard-Jones, Jeremiah Onaolapo, and Gianluca Stringhini. BABELTOWER: How Language Affects Criminal Activity in Stolen Webmail Accounts. In *Companion Proceedings of the The Web Conference (WWW)*, 2018.

[17] Michael S. Bernstein, Andrés Monroy-Hernández, Drew Harry, Paul André, Katrina Panovich, and Greg Vargas. 4chan and /b/: An Analysis of Anonymity and Ephemerality in a Large Online Community. In *AAAI Conference on Weblogs and Social Media (ICWSM)*, 2011.

[18] Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *World Wide Web Conference (WWW)*, 2009.

[19] Hamad Binsalleeh, Thomas Ormerod, Amine Boukhtouta, Prosenjit Sinha, Amr Youssef, Mourad Debbabi, and Lingyu Wang. On the analysis of the Zeus botnet crimeware toolkit. In *Privacy, Security and Trust (PST)*, 2010.

[20] Jeremy Blackburn and Haewoon Kwak. STFU NOOB! Predicting Crowdsourced Decisions on Toxic Behavior in Online Games. In *World Wide Web Conference (WWW)*, 2014.

[21] Yazan Boshmaf, Ildar Muslukhov, Konstantin Beznosov, and Matei Ripeanu. The socialbot network: when bots socialize for fame and money. In *Annual Computer Security Applications Conference (ACSAC)*, 2011.

[22] Brian M. Bowen, Shlomo Hershkop, Angelos D. Keromytis, and Salvatore J. Stolfo. Baiting inside attackers using decoy documents. In *Security and Privacy in Communication Networks (SecureComm)*, 2009.

[23] Brian M. Bowen, Vasileios P. Kemerlis, Pratap V. Prabhu, Angelos D. Keromytis, and Salvatore J. Stolfo. Automating the injection of believable de-

coys to detect snooping. In *ACM Conference on Wireless Network Security (WiSec)*, 2010.

[24] Brian M. Bowen, Pratap V. Prabhu, Vasileios P. Kemerlis, Stelios Sidiroglou, Angelos D. Keromytis, and Salvatore J. Stolfo. BotSwindler: Tamper Resistant Injection of Believable Decoys in VM-Based Hosts for Crimeware Detection. In *Symposium on Recent Advances in Intrusion Detection (RAID)*, 2010.

[25] Elie Bursztein, Borbala Benko, Daniel Margolis, Tadek Pietraszek, Andy Archer, Allan Aquino, Andreas Pitsillidis, and Stefan Savage. Handcrafted Fraud and Extortion: Manual Account Hijacking in the Wild. In *ACM Internet Measurement Conference (IMC)*, 2014.

[26] Juan Caballero, Chris Grier, Christian Kreibich, and Vern Paxson. Measuring Pay-per-Install: The Commoditization of Malware Distribution. In *USENIX Security Symposium*, 2011.

[27] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. Aiding the Detection of Fake Accounts in Large Scale Social Online Services. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.

[28] Bill Cheswick. An Evening with Berferd in which a cracker is Lured, Endured, and Studied. In *Winter USENIX Conference, San Francisco*, 1992.

[29] Cho-Yu Jason Chiang, Yitzchak M. Gottlieb, Shridatt James Sugrim, Ritu Chadha, Constantin Serban, Alexander Poylisher, Lisa M. Marvel, and Jonathan Santos. ACyDS: An adaptive cyber deception system. In *IEEE Military Communications Conference (MILCOM)*, 2016.

[30] W. Y. Chin, Evangelos P. Markatos, Spyros Antonatos, and Sotiris Ioannidis. HoneyLab: Large-Scale Honeypot Deployment and Resource Sharing. In *International Conference on Network and System Security (NSS)*, 2009.

[31] Nicolas Christin. Traveling the silk road: a measurement analysis of a large anonymous online marketplace. In *World Wide Web Conference (WWW)*, 2013.

[32] Jessica Colnago, Summer Devlin, Maggie Oates, Chelse Swoopes, Lujo Bauer, Lorrie Faith Cranor, and Nicolas Christin. “It’s not actually that horrible”: Exploring Adoption of Two-Factor Authentication at a University. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2018.

[33] H. Cramèr. On the composition of elementary errors. *Skandinavisk Aktuarietidskrift*, 1928.

[34] Jon Crowcroft, Hamed Haddadi, and Tristan Henderson. Responsible Research on Social Networks: Dilemmas and Solutions, 2017.

[35] Anupam Das, Joseph Bonneau, Matthew Caesar, Nikita Borisov, and XiaoFeng Wang. The Tangled Web of Password Reuse. In *Symposium on Network and Distributed System Security (NDSS)*, 2014.

[36] Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker, and Alex C. Snoeren. Tripwire: Inferring Internet Site Compromise. In *ACM Internet Measurement Conference (IMC)*, 2017.

[37] Rachna Dhamija, J. Doug Tygar, and Marti Hearst. Why phishing works. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2006.

[38] Kelly P Dillon and Brad J Bushman. Unresponsive or un-noticed?: Cyberbystander intervention in an experimental cyberbullying context. *Computers in human behavior*, 45:144–150, 2015.

[39] Harris Drucker, Donghui Wu, and Vladimir N Vapnik. Support vector machines for spam categorization. *IEEE Transactions on Neural Networks*, 10(5):1048–1054, 1999.

[40] Marilyn A Dyrud. I brought you a good news: An analysis of Nigerian 419 letters. In *Association for Business Communication Annual Convention*, 2005.

[41] Manuel Egele, Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. COMPA: Detecting Compromised Accounts on Social Networks. In *Symposium on Network and Distributed System Security (NDSS)*, 2013.

[42] Manuel Egele, Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. Towards Detecting Compromised Accounts on Social Networks. In *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2015.

[43] Gerald D Everett and Raymond McLeod Jr. *Software testing: testing across the entire software development life cycle*. John Wiley & Sons, 2007.

[44] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y Zhao. Detecting and characterizing social spam campaigns. 2010.

[45] Chris Grier, Kurt Thomas, Vern Paxson, and Michael Zhang. @ spam: the underground on 140 characters or less. In *ACM Conference on Computer and Communications Security (CCS)*, 2010.

[46] William G Halfond, Jeremy Viegas, and Alessandro Orso. A classification of SQL-injection attacks and countermeasures. In *IEEE Symposium on Secure Software Engineering*, 2006.

[47] Xiao Han, Nizar Kheir, and Davide Balzarotti. PhishEye: Live Monitoring of Sandboxed Phishing Kits. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.

[48] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Alexander G Gray, and Sven Krasser. Detecting Spammers with SNARE: Spatio-temporal Network-level Automatic Reputation Engine. In *USENIX Security Symposium*, 2009.

[49] Andreas Haslebacher, Jeremiah Onaolapo, and Gianluca Stringhini. All your cards are belong to us: Understanding online carding forums. In *APWG Symposium on Electronic Crime Research (eCrime)*, 2017.

[50] Gabriel Emile Hine, Jeremiah Onaolapo, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Riginos Samaras, Gianluca Stringhini, and Jeremy

Blackburn. Kek, Cucks, and God Emperor Trump: A Measurement Study of 4chan's Politically Incorrect Forum and Its Effects on the Web. In *AAAI Conference on Weblogs and Social Media (ICWSM)*, 2017.

[51] Homa Hosseinmardi, Amir Ghasemianlangroodi, Richard Han, Qin Lv, and Shivakant Mishra. Analyzing Negative User Behavior in a Semi-anonymous Social Network. In *IEEE/ACM Conference on Advances in Social Network Analysis and Mining (ASONAM)*, 2014.

[52] JingMin Huang, Gianluca Stringhini, and Peng Yong. Quit Playing Games with My Heart: Understanding Online Dating Scams. In *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. 2015.

[53] Paul Hunton. Data attack of the cybercriminal: Investigating the digital currency of cybercrime. *Computer Law & Security Review*, 28(2):201–207, 2012.

[54] Paul Hyman. Cybercrime: it's serious, but exactly how serious? *Communications of the ACM*, 56(3):18–20, 2013.

[55] Philip Inglesant and Martina Angela Sasse. The true cost of unusable password policies: password use in the wild. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.

[56] Jelena Isacenkova, Olivier Thonnard, Andrei Costin, Aurélien Francillon, and Davide Balzarotti. Inside the SCAM jungle: A closer look at 419 scam email operations. *EURASIP J. Information Security*, 2014:4, 2014.

[57] Gregoire Jacob, Engin Kirda, Christopher Kruegel, and Giovanni Vigna. PUBCRAWL: Protecting Users and Businesses from CRAWLers. In *USENIX Security Symposium*, 2012.

[58] Tom N Jagatic, Nathaniel A Johnson, Markus Jakobsson, and Filippo Menczer. Social Phishing. *Communications of the ACM*, 50(10):94–100, 2007.

[59] John P John, Alexander Moshchuk, Steven D Gribble, and Arvind Krishnamurthy. Studying Spamming Botnets Using Botlab. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2009.

[60] Alexander Kedrowitsch, Danfeng (Daphne) Yao, Gang Wang, and Kirk Cameron. A First Look: Using Linux Containers for Deceptive Honeypots. In *Workshop on Automated Decision Making for Active Cyber Defense*, 2017.

[61] Bryan Klimt and Yiming Yang. Introducing the Enron Corpus. In *Conference on Email and Anti-Spam (CEAS)*, 2004.

[62] Martin Lazarov, Jeremiah Onaolapo, and Gianluca Stringhini. Honey Sheets: What Happens to Leaked Google Spreadsheets? In *USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2016.

[63] Kyumin Lee, James Caverlee, and Steve Webb. The social honeypot project: protecting online communities from spammers. In *World Wide Web Conference (WWW)*, 2010.

[64] Fanny Lalonde Lévesque, Jude Nsiempba, José M. Fernandez, Sonia Chiasson, and Anil Somayaji. A clinical study of risk factors related to malware infections. In *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[65] Bingshuang Liu, Zhaoyang Liu, Jianyu Zhang, Tao Wei, and Wei Zou. How many eyes are spying on your shared folders? In *ACM Workshop on Privacy in the Electronic Society (WPES)*, 2012.

[66] Lila A Loos and Martha E Crosby. Cognition and Predictors of Password Selection and Usability. In *International Conference on Augmented Cognition*, 2018.

[67] Niels Provos Panayiotis Mavrommatis and Moheeb Abu Rajab Fabian Monrose. All your iframes point to us. In *USENIX Security Symposium*, pages 1–16, 2008.

[68] Gibson Mba, Jeremiah Onaolapo, Gianluca Stringhini, and Lorenzo Cavallaro. Flipping 419 Cybercrime Scams: Targeting the Weak and the Vulnerable. In *Companion Proceedings of the World Wide Web Conference (WWW)*, 2017.

[69] Marti Motoyama, Damon McCoy, Kirill Levchenko, Stefan Savage, and Geoffrey M. Voelker. An analysis of underground forums. In *ACM Internet Measurement Conference (IMC)*, 2011.

[70] Collin Mulliner, Steffen Liebergeld, and Matthias Lange. Poster: Honeydroid-creating a smartphone honeypot. In *IEEE Symposium on Security and Privacy*, 2011.

[71] Vinita Nahar, Sanad Al-Maskari, Xue Li, and Chaoyi Pang. Semi-supervised learning for cyberbullying detection in social networks. In *Australasian Database Conference*, 2014.

[72] Vinita Nahar, Sayan Unankard, Xue Li, and Chaoyi Pang. Sentiment analysis for effective detection of cyber bullying. In *Asia-Pacific Web Conference*, 2012.

[73] B Sri Nandhini and JI Sheeba. Online social network bullying detection using intelligence techniques. *Procedia Computer Science*, 45:485–492, 2015.

[74] Jose Nazario. Phoneyc: A virtual client honeypot. In *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2009.

[75] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *IEEE Symposium on Security and Privacy*, 2013.

[76] Jeremiah Onaolapo, Enrico Mariconti, and Gianluca Stringhini. What Happens After You Are Pwnd: Understanding the Use of Leaked Webmail Credentials in the Wild. In *ACM Internet Measurement Conference (IMC)*, 2016.

[77] Rodmonga Potapova and Denis Gordeev. Determination of the internet anonymity influence on the level of aggression and usage of obscene lexis. *arXiv preprint arXiv:1510.00240*, 2015.

[78] Stéphanie A Prince, Kristi B Adamo, Meghan E Hamel, Jill Hardt, Sarah Connor Gorber, and Mark Tremblay. A comparison of direct versus self-report measures for assessing physical activity in adults: a systematic review. *International Journal of Behavioral Nutrition and Physical Activity*, 5(1):56, 2008.

[79] Niels Provos et al. A Virtual Honeypot Framework. In *USENIX Security Symposium*, 2004.

[80] Niels Provos and David Mazières. A Future-Adaptable Password Scheme. In *USENIX Annual Technical Conference*, pages 81–91, 1999.

[81] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[82] Christian Rossow, Christian J Dietrich, Chris Grier, Christian Kreibich, Vern Paxson, Norbert Pohlmann, Herbert Bos, and Maarten van Steen. Prudent practices for designing malware experiments: Status quo and outlook. In *IEEE Symposium on Security and Privacy*, 2012.

[83] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization*, 1998.

[84] Malek Ben Salem and Salvatore J Stolfo. Decoy document deployment for effective masquerade attack detection. In *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. 2011.

[85] Richard Shay, Iulia Ion, Robert W Reeder, and Sunny Consolvo. My religious aunt asked why I was trying to sell her viagra: Experiences with account hi-

jacking. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2014.

[86] David Silver, Suman Jana, Dan Boneh, Eric Yawei Chen, and Collin Jackson. Password Managers: Attacks and Defenses. In *USENIX Security Symposium*, 2014.

[87] Lance Spitzner. Honeypots: Catching the insider threat. In *Annual Computer Security Applications Conference (ACSAC)*, 2003.

[88] Frank Stajano. Pico: No more passwords! In *International Workshop on Security Protocols*, 2011.

[89] Clifford Stoll. The cuckoo's egg: Tracing a spy through the maze of computer espionage, 1989.

[90] Brett Stone-Gross, Marco Cova, Lorenzo Cavallaro, Bob Gilbert, Martin Szydlowski, Richard Kemmerer, Christopher Kruegel, and Giovanni Vigna. Your Botnet is My Botnet: Analysis of a Botnet Takeover. In *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[91] Brett Stone-Gross, Thorsten Holz, Gianluca Stringhini, and Giovanni Vigna. The underground economy of spam: A botmaster's perspective of coordinating large-scale spam campaigns. In *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2011.

[92] Gianluca Stringhini, Oliver Hohlfeld, Christopher Kruegel, and Giovanni Vigna. The harvester, the botmaster, and the spammer: on the relations between the different actors in the spam landscape. In *ACM Symposium on Information, Computer and Communications Security*, 2014.

[93] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. Detecting Spammers on Social Networks. In *Annual Computer Security Applications Conference (ACSAC)*, 2010.

[94] Gianluca Stringhini and Olivier Thonnard. That Ain't You: Blocking Spearphishing Through Behavioral Modelling. In *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, 2015.

[95] Yuchun Tang, Sven Krasser, Paul Judge, and Yan-Qing Zhang. Fast and effective spam sender detection with granular SVM on highly imbalanced mail server behavior data. In *Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2006.

[96] Bradley Taylor. Sender Reputation in a Large Webmail Service. In *Conference on Email and Anti-Spam (CEAS)*, 2006.

[97] Daniel R. Thomas, Sergio Pastrana, Alice Hutchings, Richard Clayton, and Alastair R. Beresford. Ethical issues in research using datasets of illicit origin. In *ACM Internet Measurement Conference (IMC)*, 2017.

[98] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson. Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *USENIX Security Symposium*, 2013.

[99] Kurt Thomas, Chris Grier, Dawn Song, and Vern Paxson. Suspended accounts in retrospect: an analysis of Twitter spam. In *ACM Internet Measurement Conference (IMC)*, 2011.

[100] Charles Tive. *419 scam: Exploits of the Nigerian con man*. iUniverse, 2006.

[101] Nikos Virvilis, Bart Vanautgaerden, and Oscar Serrano Serrano. Changing the game: The art of deceiving sophisticated attackers. In *International Conference on Cyber Conflict (CyCon)*, 2014.

[102] Michael Vrable, Justin Ma, Jay Chen, David Moore, Erik Vandekieft, Alex C. Snoeren, Geoffrey M. Voelker, and Stefan Savage. Scalability, fidelity, and containment in the potemkin virtual honeyfarm. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2005.

[103] Ding Wang, Zijian Zhang, Ping Wang, Jeff Yan, and Xinyi Huang. Targeted Online Password Guessing: An Underestimated Threat. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.

[104] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y Zhao. You Are How You Click: Clickstream Analysis for Sybil Detection. In *USENIX Security Symposium*, 2013.

[105] Yi-Min Wang, Doug Beck, Xuxian Jiang, Roussi Roussev, Chad Verbowski, Shuo Chen, and Sam King. Automated web patrol with strider honeymonkeys. In *Symposium on Network and Distributed System Security (NDSS)*, 2006.

[106] Steve Webb, James Caverlee, and Calton Pu. Social Honeypots: Making Friends With A Spammer Near You. In *Conference on Email and Anti-Spam (CEAS)*, 2008.

[107] Monica T Whitty and Tom Buchanan. The online romance scam: A serious cybercrime. *CyberPsychology, Behavior, and Social Networking*, 15(3):181–183, 2012.

[108] Chao Yang, Jialong Zhang, and Guofei Gu. A taste of tweets: reverse engineering Twitter spammers. In *Annual Computer Security Applications Conference (ACSAC)*, 2014.

[109] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben. Y Zhao, and Yafei Dai. Uncovering Social Network Sybils in the Wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1):2:1–2:29, February 2014.

[110] Jim Yuill, Mike Zappe, Dorothy Denning, and Fred Feer. Honeyfiles: deceptive files for intrusion detection. In *IEEE SMC Information Assurance Workshop*, 2004.

[111] Savvas Zannettou, Tristan Caulfield, Emiliano De Cristofaro, Nicolas Kourtelris, Ilias Leontiadis, Michael Sirivianos, Gianluca Stringhini, and Jeremy

Blackburn. The web centipede: understanding how web communities influence each other through the lens of mainstream and alternative news sources. In *ACM Internet Measurement Conference (IMC)*, 2017.

[112] Savvas Zannettou, Tristan Caulfield, Emiliano De Cristofaro, Michael Sirivianos, Gianluca Stringhini, and Jeremy Blackburn. Disinformation Warfare: Understanding State-Sponsored Trolls on Twitter and Their Influence on the Web. *arXiv preprint arXiv:1801.09288*, 2018.

[113] Savvas Zannettou, Michael Sirivianos, Jeremy Blackburn, and Nicolas Kourtellis. The Web of False Information: Rumors, Fake News, Hoaxes, Clickbait, and Various Other Shenanigans. *arXiv preprint arXiv:1804.03461*, 2018.