Honey Sheets: What Happens To Leaked Google Spreadsheets?

Martin Lazarov, Jeremiah Onaolapo, and Gianluca Stringhini
University College London
martin.lazarov.12@ucl.ac.uk,{j.onaolapo,g.stringhini}@cs.ucl.ac.uk

Abstract

Cloud-based documents are inherently valuable, due to the volume and nature of sensitive personal and business content stored in them. Despite the importance of such documents to Internet users, there are still large gaps in the understanding of what cybercriminals do when they illicitly get access to them by for example compromising the account credentials they are associated with. In this paper, we present a system able to monitor user activity on Google spreadsheets. We populated 5 Google spreadsheets with fake bank account details and fake funds transfer links. Each spreadsheet was configured to report details of accesses and clicks on links back to us. To study how people interact with these spreadsheets in case they are leaked, we posted unique links pointing to the spreadsheets on a popular paste site. We then monitored activity in the accounts for 72 days, and observed 165 accesses in total. We were able to observe interesting modifications to these spreadsheets performed by illicit accesses. For instance, we observed deletion of some fake bank account information, in addition to insults and warnings that some visitors entered in some of the spreadsheets. Our preliminary results show that our system can be used to shed light on cybercriminal behavior with regards to leaked online documents.

1   Introduction

Many useful services are cloud-based nowadays, for example Dropbox [1] and OneDrive [5] which are both file storage services. Reasons for using cloud-based storage services include ease of access to files stored in the cloud and the ability to collaborate easily on shared files. According to a study by Eurostat, 21% of EU citizens stored documents and multimedia files in the cloud as of 2014 [8]. Apart from individuals, many organizations also rely heavily on cloud-based services for office duties, such as Google Apps for Work [3], a cloud-based suite that provides business tools like official email addresses, calendars, online storage space, and document processing services.

Consequently, a considerable amount of valuable information is stored in online accounts, including sensitive personal information and business secrets. These in turn attract cybercriminals seeking to make profit from such information. To gain access to such online accounts, cybercriminals typically target owners of the accounts with phishing attacks, and sometimes, malware. Often, databases containing user credentials are also attacked, and credentials in them get stolen.

Stolen credentials are usually sold by cybercriminals in underground marketplaces, to other cybercriminals for malicious purposes. Depending on the nature of the compromised accounts, some are used to send unsolicited messages, and some are assessed by the cybercriminals with the hope of finding other valuable information that could be used to stage further attacks, either against the victim, or the victim's contacts. Such information of interest to cybercriminals include contacts lists, financial details and authentication credentials linked to other accounts.

Previous work on compromised cloud-based accounts is limited. They focus on malicious activity in compromised webmail accounts, with emphasis on spearphishing as a primary attack vector [10, 25]. It is hard for researchers to collect data on compromised accounts because there are no existing publicly available systems built for that purpose. To partially bridge this gap, we develop a system based on Google Apps Script [6] to monitor compromised online spreadsheets. Our system is able to monitor activity performed by users on Google spreadsheets, such as open events, modifications, and clicks on links. Having visibility on these events is key for researchers wanting to understand what is happening to cloud documents once they are accessed by attackers.

To demonstrate the usefulness of our system to study illicit accesses to online documents, we set up honey

spreadsheets in Google Docs and populate them with fake financial information. We also embed obfuscated links in the spreadsheets, some pointing to a domain under our control, others pointing to invalid pages on some selected banks' websites. We leak unique links pointing to the spreadsheets on a popular paste site, after configuring each spreadsheet to allow anyone with the links to edit the spreadsheets. We set up two scenarios on why these spreadsheets could be posted on paste sites: a user innocently sharing them with her colleagues, inadvertently making it accessible to anyone on the Internet, and a criminal posting a link to the spreadsheet after having obtained illicit access to it. Compromised credentials and other sensitive information are routinely posted by cybercriminals on paste sites [26], so this scenario is believable.

After posting the links, we monitor accesses to the honey spreadsheets, and log interactions and activity in them. We aim to answer the following research questions:

Question 1: What actions do attackers take on stolen cloud-based documents? Can we identify typologies of cybercriminals based on the actions they take in the documents?

Question 2: What kind of content do cybercriminals find more attractive than others, within the same document? What kind of content do they interact with more than others?

Over a period of 72 days, we conducted two experiments and observed 165 accesses, 28 modification events and 174 clicks on the embedded obfuscated links. The clicks originated from 35 different countries. The modifications we observed include deletion of fake payment information, entry of insults into a spreadsheet, and defacement of the same spreadsheet, thus rendering it unusable.

This paper makes the following contributions:

- We develop a novel system to track activity in Google spreadsheets. The system is able to observe file opening and modification of content, in addition to tracking clicks on links contained in the spreadsheets. The system also records IP addresses, request paths and HTTP headers for clicks on the embedded links.

- We deployed 5 honey spreadsheets and leaked links pointing to them on a popular paste site. We perform experiments for 72 days and measure accesses to the spreadsheets. During the experiments, visitors modified and deleted fake payment information, expanded some columns in the spreadsheets in order to get better views of the data in those columns, and also replaced an obfuscated link in a spreadsheet with a C++ code snippet.

- We discuss how our approach could be used to set up different scenarios and give the research community a better understanding of the modus operandi of cybercriminals compromising stolen online documents.

2 Background

2.1 Cloud-based spreadsheets

As stated earlier, cloud-based accounts are popular nowadays, and a large amount of documents particularly are stored in cloud-based storage facilities. We focus our study on Google spreadsheets, particularly on actions taken by cybercriminals when they compromise the spreadsheets. We chose Google spreadsheets since the Google platform offers scripting functionality that allows us to boost the capability of the spreadsheets, to monitor activity in them. Our methodology can be applied to other cloud-based documents as well. In this section, we summarize the capabilities that spreadsheets provide, and focus on Google spreadsheets specifically.

When logged in to a Google account, users can create a new spreadsheet, or import an existing one. Each spreadsheet comprises cells arranged in rows and columns. Users can delete existing rows and columns of cells, move existing rows and columns around, and insert new ones as well.

For purposes of collaboration, users can share spreadsheets with other users usually by listing their email addresses, or by sharing a unique link pointing to the spreadsheet. Depending on permission settings specified by the document owner, collaborators can have view-only, comment-only, or edit access.

Spreadsheet owners can extend the functionality of their spreadsheets by leveraging Google Apps Script [6], a cloud-based Javascript engine for attaching scripts to documents in Google Docs platform. The scripts may be triggered by events such as opening of the spreadsheet or modification of its contents. Such scripts may also be triggered at some specified time intervals. Google Apps Script is commonly used for development of minimal web applications, in addition to extending the functionality of Gmail accounts and other Google products [6]. Google Apps Script can be configured to send out notifications when triggered, among other tasks.

2.2 Threat model

Cloud-based documents can get compromised in a number of ways. Cybercriminals mount phishing and spearphishing attacks on online accounts; this is the most common way by which cybercriminals manually hijack online accounts, according to Bursztein et al. [10].

Information-stealing malware also capture login credentials safeguarding online accounts, for instance after drive-by downloads or clicking on spurious links and attachments in emails. Attackers also compromise accounts by hacking databases containing login credentials.

Poor file access configuration, for example granting "allow-all" permissions, rather than specifying a whitelist of permitted parties, can also expose online documents to threats. Attackers have easy access if they get hold of pointers to a file, for instance by eavesdropping on communications about shared files, and grabbing links pointing to such files.

Attackers can also mount Man-in-the-Cloud attacks on documents stored online, by stealing authentication tokens and impersonating legitimate parties that have access to the compromised documents [9].

Previous work focused on what happens in webmail accounts compromised via phishing attacks [10]. However, our work focuses on attacks on documents in the cloud. Similarly, Stolfo et al. proposed a method of mitigating insider threats to confidential data stored in the cloud [21]. Their work, however, focuses on insider threats, while we consider a broader threat model.

3 Methodology

Our aim was to gain insight into what attackers do after gaining unauthorized access to leaked spreadsheets. To achieve this, we developed a system based on Google Apps Script [6] to monitor events performed by users on Google spreadsheets, such as accesses, modifications, and click on links. We then set up experiments in which we created five Google spreadsheets and added honey URLs in them, some pointing to a domain under our control, others pointing to non-existent bank pages. Figure 1 shows an overview of the honeypot infrastructure we set up, to monitor accesses to the honey spreadsheets. We leaked links pointing to the honey spreadsheets on pastebin.com [7], a popular paste site. The idea is to observe what captures the attention of attackers, and how they interact with the spreadsheets.

3.1 Honey spreadsheets

We set up a Google account and created five spreadsheets in it. The spreadsheets were populated with randomly selected names and fake bank account details, including fake International Bank Account Numbers (IBANs), sort codes, and links masquerading as funds transfer pointers apparently connected to some of the fake accounts. Some of the links point to a domain under our control. The idea is to log attempts made by cybercriminals to transfer
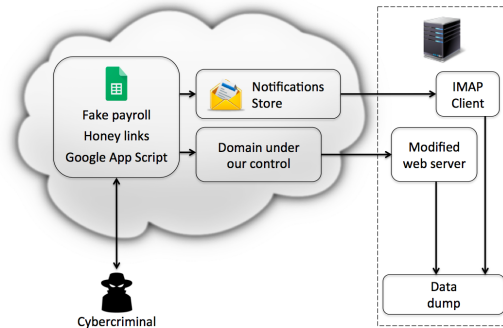

Figure 1: A simplified overview of our honeypot infrastructure. Visitors interact with honey spreadsheets, and information about their actions is sent to the webmail account that receives notifications. If certain links in the spreadsheets are clicked, our modified web server receives the generated HTTP requests, logs information about the requests, and redirects them to google.com [2]. We also monitor clicks on all embedded links through Google Analytics, since we used Google's URL shortener to obfuscate those links before inserting them in the honey spreadsheets.

funds out of the non-existent honey accounts. One of the honey spreadsheets is shown in Figure 2.

Figure 2: A partial view of a leaked honey spreadsheet, containing fake payroll information. Note that shortened URLs are also in the same spreadsheet, but not shown here.

On creating a spreadsheet on the Google Docs platform, a unique link is generated by Google. Anyone with the link can thus access the spreadsheet, either with read-only permission or edit permission, depending on the access settings in the spreadsheet. We configured each spreadsheet to grant edit permission to anyone in possession of its unique link. In addition, the spreadsheets were instrumented with scripts to monitor activity, for instance, file open and edit events. The hidden scripts report those events back to us by sending email notifications to a webmail account under our control. The scripts were implemented in Google Apps Script [6].

Our monitoring scripts periodically take snapshots of the honey spreadsheets, and send notification messages about those snapshots to a webmail account under our control. Information contained in the notifications include modified content, changes in spreadsheet structure, the date and time that the spreadsheet was last accessed, and date and time that the spreadsheet was last updated. We periodically retrieve and collate notifications from the webmail account, and parse them locally.

We set up an experiment involving spreadsheets whose URLs are posted on public paste sites. This experiment shows the usefulness of our system, and we believe that our approach will be helpful in future related work on the behavior of cybercriminals that compromise documents stored in the cloud. In Section 5 we explore additional experiments that could be performed to further understand the ecosystem around stolen online documents.

Honey links: One of our goals was to observe if cybercriminals would attempt to observe or tamper directly with any of the honey bank accounts listed in the payroll. To achieve this, we did the following: In each honey spreadsheet, we inserted 9 URLs, 3 of them pointing to a domain under our control, and the remaining 6 pointing to non-existent bank pages. Each URL was shortened and obfuscated using goo.gl [4], an online URL shortener. The idea is that all clicks on the embedded links would be logged by Google Analytics platform, which monitors URLs that were shortened using goo.gl [4], and clicks on the 3 links pointing to our web server would be logged by the web server. Figure 3 shows the honey URLs we embedded in the honey spreadsheets.

Transfer Accounts	Payment Link
Barclays	http://goo.gl/TmzFcm
HSBC	http://goo.gl/epbE0Q
Natwest	http://goo.gl/p9yti9
Lloyds	http://goo.gl/uYrZ6V
Halifax	http://goo.gl/AHGjKy
TSB	http://goo.gl/nIsJLQ
Paypal	http://goo.gl/PHr1wE
TransferWise	http://goo.gl/k0Gf35

Figure 3: Honey links embedded in leaked honey spreadsheet. The URLs were obfuscated using a URL shortener, and some of them point to a web server under our control.

3.2 Monitoring the spreadsheets

In addition to tracking changes in the honey spreadsheets using Google Apps Script, we incorporated honey links in each spreadsheet. As earlier stated, the idea is to track attempts by cybercriminals to stage further attacks on the honey accounts listed in the spreadsheets. Some of the embedded honey links point to a domain under our control, and clicks on those ones generate HTTP requests to a web server under our control.

We configured the web server to redirect all incoming requests to google.com [2], and log details such as the visitor's IP address, originating port number, HTTP header details, request paths, and timestamps of accesses. The purpose of the web server is to deflect the attention of visitors from our honeypot infrastructure, while logging accesses simultaneously.

We periodically run another script that connects to the webmail account that receives email notifications from the honey spreadsheets, to retrieve those notifications through the Internet Message Access Protocol (IMAP). This enables us to parse and analyse the notifications offline.

Links pointing to the honey spreadsheets were leaked on pastebin.com [7], to entice attackers to interact with the honey sheets. We set up two experiments: in the first one, we pretended to be the legitimate owner of the spreadsheet, sharing it with a colleague without realizing that this would make it accessible to the entire Internet. In the second experiment we presented to be a criminal who got illicit access to the spreadsheet and posted in on the paste sites to brag. Our experiments ran for 72 days.

3.3 Ethics

Our experiments involve deceiving potential cybercriminals into interacting with fake spreadsheets containing fake financial information. We made sure that all information in the spreadsheets were not related to any real financial account. To mitigate the possibility of misuse of the online account hosting the honey spreadsheets, we did not leak credentials of the account, thus ensuring we remained in full control of the account throughout the experiments. Finally, to ensure that our experiments were run in an ethical manner, we sought and obtained permission from our institution's ethics committee, under project number 6521/001.

4 Results

In this section, we discuss the activity observed in honey spreadsheets as part of our experiment.We also provide details of accesses to the modified web server under our control, and Google Analytics data we obtained from the shortened URLs we embedded in the spreadsheets.

4.1 Overview

Over the period of evaluation (72 days), we conducted two experiments with the honey sheets, with two main leak message themes. During the first experiment, we

pretended to be a hacker that discovered some corporate financial information. For instance, we leaked messages like "leaked corporate payments" and "st0len payrolls" along with unique links pointing to the spreadsheets. In the second experiment, we pretended to be a naive person sharing financial information with a colleague, and inadvertently leaking such information. In this case, the leak messages were of the form "Bob, here is the spreadsheet with payrolls for September." The idea was to identify the leak message theme that would generate more interest in the honey sheets.

The first experiment, in which we pretended to be a hacker, was conducted from 23rd January 2016 till 8th March 2016 (46 days), during which links to the honey sheets were leaked twice daily. The second experiment, in which we pretended to naively share links with a colleague, was carried out from 9th March 2016 till 3rd April 2016 (26 days). Similarly, we shared unique links pointing to the honey sheets twice daily.

4.2 Activity on honey spreadsheets

For each leaked honey spreadsheet, we recorded file open events and file modification events. File open events are triggered when a honey spreadsheet is opened by visiting the unique URL pointing to it, while modification events are triggered when the structure of the spreadsheet is changed by the person accessing it. Modification events are also triggered when the contents of the spreadsheet are changed. We recorded snapshots of each spreadsheet every two hours, and compared those snapshots over time, to monitor the values that were modified in the spreadsheets.

In the first experiment, during which we pretended to be a hacker leaking financial information, we observed 112 file open events and 17 modification events over a period of 46 days. In the second experiment, in which we posed as a naive user, we observed 53 file open events and 11 modification events over 26 days. We could not identify significant statistical differences between the two experiments with regards to file open and modification events. The dataset containing the details of the open and modification events is available at http://dx.doi.org/10.14324/000.ds.1502241.

We acknowledge the possibility that some visitors may make copies of the spreadsheets for offline viewing and modification. This is one of the limiting factors in the experiments.

Interesting modification events. We observed some interesting modification events among the 28 recorded during our experiments. In one of the honey spreadsheets, one of the bank account numbers was deleted, and the spreadsheet itself was rendered unusable. The attacker changed font sizes and colors, along with background colors in the spreadsheet, and replaced all the embedded honey links with the short URL https://goo.gl/ufniSo. The short URL redirects to a Google search page for "minions." In addition, the word '\MINIONSXDDDD' was written on the spreadsheet, while an insult was entered in the spreadsheet. Interestingly, it appears the attacker did not click on any of the embedded honey links. This case study shows that our honey spreadsheet infrastructure could be used to study trolling, cyber bullying, and hate speech behavior on the web.

In another case, we observed a number of spreadsheet modifications that brought about no change in the content of the affected honey spreadsheets. A closer look revealed that the viewers expanded some columns, especially the ones containing fake payment links for funds transfer, presumably to have a better view of the URLs there.

4.3 Activity on embedded honey links

We observed 39 unique IP addresses that visited the 3 links pointing to our web server, and a total of 44 visits to those 3 links. We recorded accesses from 35 different countries, and a total of 174 clicks on all the 9 embedded honey links. Those clicks originated from the following browsers: Chrome, Firefox, Safari, and Samsung browsers, on devices running Windows, Linux, Macintosh, and Android operating systems.

Location of accesses. In order to have an idea of the locations of visitors accessing the honey spreadsheets, we extracted country information from Google Analytics dashboard, and also performed geolocation on IP addresses recorded by our web server. We plotted the locations on a world map, as shown in Figure 4.

5 Discussion

In this section, we mention the limitations that we encountered during the experiments, and discuss some additional experiments that could be performed using our infrastructure to better understand the modus operandi of attackers and cybercriminals illicitly accessing cloud documents.

Limitations. We were able to leak the spreadsheets only on a paste site (and not, for example, on an underground forum), implying that visitors accessing the spreadsheets are likely unsophisticated cybercriminals or just curious users. This is further emphasized by the fact that the visitors did not appear to make much effort to cover up their accesses.

Attackers could make copies of the honey spreadsheets and explore them offline. We would be unable to monitor what they do with the spreadsheets offline, unless they click on the embedded honey links.

Figure 4: Location of visits to the embedded honey links.

In Google spreadsheets, there are limitations on the functionality of Google Apps Script for users that are not logged in. Since visitors are not required to login before interacting with the spreadsheets, there are limits on the amount of information we can collect on the visitors, and also how much activity we can monitor in the spreadsheets. This is an intrinsic limitation of our monitoring infrastructure.

Future work. In the future, we plan to continue studying the modus operandi of cybercriminals interacting with compromised online documents, and the underground economy surrounding them. We intend to instrument more documents, and leak documents containing honey credentials linked to honey web services. The idea is that cybercriminals would attempt to log in to those honey web services using the honey credentials, and this hopefully would be a good way to distinguish between merely curious visitors and cybercriminals. As a result, leaking credentials in honey documents would give us broader visibility into their modus operandi. We also plan to leak documents to more sites (for instance, through malware and underground forums), and study differences in accesses across different leak sites. We could modify the existing monitoring infrastructure we have developed to other online document storage platforms. We also plan to devise a taxonomy of attackers targeting online cloud documents. Finally, we intend to study online trolling and hate speech further, having discovered that our honey documents attract insults and destructive action (defacing).

6 Related Work

Honeypots have proved to be useful in cyber security studies, for instance, in the analysis of malware behaviour in controlled environments, tracking of insider threats in organizations, and understanding cybercriminal schemes. In general, resources created with the sole purpose of being misused by cybercriminals to enable researchers to learn their modus operandi are referred to as *honeytokens*. In this section, we compare our work with previous studies.

Applications of honeytokens to security. Liu et al. [15] studied the problem of privacy in Peer-to-Peer (P2P) networks, using honey files. Similarly, Salem et al. [18] and Spitzner [20] proposed the use of decoy documents to track attackers, mitigate insider threats, and detect intruders [27]. Unlike those studies, we consider documents stored in the cloud instead of documents stored locally. In 2012, Sobesto et al. [19] studied the impact of computer configuration on computer-focused attacks, by deploying honeypot machines and allowing brute-force attackers to "break" into them via SSH. Unlike our work, they do not study compromised online documents. Nikiforakis et al. [16] used honey files to expose privacy leaks and abuse on file hosting services. Kapravelos et al. [13] used honey web pages to study privacy invasive and mali-

cious browser extensions on Google Chrome. De Cristofaro et al. [11] created a Facebook page for a fake company on Facebook, with the goal of studying the fake likes that it would receive. Nikiforakis et al. [17] displayed fake advertisements on ad-based URL shortening services, with the goal of exposing their privacy and security issues. A handful of papers [14, 24] deployed honey profiles on online social networks to identify accounts operated by spammers. Stringhini et al. [23] identified various actors in the email spam ecosystem, and their interrelationships, by leaking honey email addresses and monitoring spam emails received by them. In this paper we looked at honey Google Documents, with the goal of understanding the activity of malicious users on such documents.

Studies on compromised accounts and credentials.
Stone-Gross et al. [22] investigated the *Pushdo/Cutwail* botnet, and discovered that cybercriminals were selling online account information in underground forums. We used a similar approach to leak links pointing to honey spreadsheets on public paste sites. Bursztein et al. [10] studied account activity in Gmail accounts that were compromised mostly through phishing pages. In contrast to Bursztein et al. [10], this paper focuses on activity in Google spreadsheets, rather than on Gmail accounts. Egele et al. [12] studied the problem of compromised online social network accounts, developing behavioral models to detect and block these compromises. We believe that the activity of hijackers on honey documents will be significantly different than the one from their legitimate owners. For these reasons, similar techniques to detect compromises could be applied to the context of honey documents as well.

7   Conclusion

In this paper, we presented a honey spreadsheet system able to monitor the activity of cybercriminals interacting with compromised Google spreadsheets. We set up 5 honey spreadsheets, populated with fake financial data, and leaked them on a paste site. Prior to leaking them, we embed 9 honey links in each spreadsheet, to track potential attempts to mount attacks against the fake financial accounts in the spreadsheets. We observed accesses to the spreadsheets, and attempts by visitors to access some honey links embedded in the spreadsheets, for a period of 72 days. We present preliminary results of the experiments and highlight how our method may be useful in studying malicious activity in online documents, in general.

8   Acknowledgments

References

[1] Dropbox. https://www.dropbox.com/.

[2] Google. https://www.google.com/.

[3] Google apps for work. https://apps.google.co.uk/intx/en_au/products/.

[4] Google url shortener. https://goo.gl/.

[5] Microsoft One Drive. https://onedrive.live.com/.

[6] Overview of Google Apps Script. https://developers.google.com/apps-script/overview.

[7] Pastebin. pastebin.com.

[8] Internet and cloud services - statistics on the use by individuals. http://ec.europa.eu/eurostat/statistics-explained/index.php/Internet_and_cloud_services_-_statistics_on_the_use_by_individuals, 2014.

[9] Man in the cloud (mitc) attacks. https://www.imperva.com/docs/HII_Man_In_The_Cloud_Attacks.pdf, 2015.

[10] BURSZTEIN, E., BENKO, B., MARGOLIS, D., PIETRASZEK, T., ARCHER, A., AQUINO, A., PITSILLIDIS, A., AND SAVAGE, S. Handcrafted fraud and extortion: Manual account hijacking in the wild. In *ACM SIGCOMM Conference on Internet Measurement* (2014).

[11] DE CRISTOFARO, E., FRIEDMAN, A., JOURJON, G., KAAFAR, M. A., AND SHAFIQ, M. Z. Paying for likes?: Understanding facebook like fraud using honeypots. In *Internet Measurement Conference (IMC)* (2014).

[12] EGELE, M., STRINGHINI, G., KRUEGEL, C., AND VIGNA, G. Compa: Detecting compromised accounts on social networks. In *Symposium on Network and Distributed System Security (NDSS)* (2013).

[13] KAPRAVELOS, A., GRIER, C., CHACHRA, N., KRUEGEL, C., VIGNA, G., AND PAXSON, V. Hulk: Eliciting malicious behavior in browser extensions. In *USENIX Security Symposium* (2014).

[14] LEE, K., CAVERLEE, J., AND WEBB, S. Uncovering social spammers: social honeypots+ machine learning. In *Iternational ACM SIGIR conference on Research and development in information retrieval* (2010).

[15] LIU, B., LIU, Z., ZHANG, J., WEI, T., AND ZOU, W. How many eyes are spying on your shared folders? In *ACM workshop on Privacy in the electronic society* (2012).

[16] NIKIFORAKIS, N., BALDUZZI, M., VAN ACKER, S., JOOSEN, W., AND BALZAROTTI, D. Exposing the lack of privacy in file hosting services. In *LEET* (2011).

[17] NIKIFORAKIS, N., MAGGI, F., STRINGHINI, G., RAFIQUE, M. Z., JOOSEN, W., KRUEGEL, C., PIESSENS, F., VIGNA, G., AND ZANERO, S. Stranger danger: exploring the ecosystem of ad-based url shortening services. In *World Wide Web Conference (WWW)* (2014).

[18] SALEM, M. B., AND STOLFO, S. J. Decoy document deployment for effective masquerade attack detection. In *Detection of Intrusions and Malware, and Vulnerability Assessment.* 2011.

[19] SOBESTO, B., CUKIER, M., AND MAIMON, D. Are computer focused crimes impacted by system configurations? an empirical study. In *Software Reliability Engineering (ISSRE), 2012 IEEE 23rd International Symposium on* (2012), IEEE, pp. 191–200.

[20] SPITZNER, L. Honeypots: Catching the insider threat. In *Annual Computer Security Applications Conference (ACSAC)* (2003).

[21] STOLFO, S. J., SALEM, M. B., AND KEROMYTIS, A. D. Fog computing: Mitigating insider data theft attacks in the cloud. In *IEEE Security and Privacy Workshops (SPW)* (2012).

[22] STONE-GROSS, B., HOLZ, T., STRINGHINI, G., AND VIGNA, G. 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).

[23] STRINGHINI, G., HOHLFELD, O., KRUEGEL, C., AND VIGNA, G. 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 (ASIACCS)* (2014).

[24] STRINGHINI, G., KRUEGEL, C., AND VIGNA, G. Detecting Spammers on Social Networks. In *Annual Computer Security Applications Conference (ACSAC)* (2010).

[25] STRINGHINI, G., AND THONNARD, O. That ain't you: Blocking spearphishing through behavioral modelling. In *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)* (2015).

[26] WALKER, D. Study: Leaked credentials on pastebin, similar sites, linked to 47 gov't agencies. http://www.scmagazine.com/analysis-of-17-paste-sits-uncovers-login-\credentials-from-47-govt-agencies/article/422921/, 2015.

[27] YUILL, J., ZAPPE, M., DENNING, D., AND FEER, F. Honeyfiles: deceptive files for intrusion detection. In *IEEE Information Assurance Workshop* (2004).