Information loss to public networks

Searching for threats and lost information

Authors:
Peter van Bolhuis
University of Amsterdam

Jan-Willem Selij
University of Amsterdam

Supervisors:
Steven Raspe
abn amro

Floris Ladan
abn amro

January 31, 2014

Abstract - The first phase of the research focused on identifying data-loss hotspots. This was done by extracting data from a proxy logger for a large (Dutch) company. It showed a lot of outgoing traffic related to e-mail and attachments. Other large factors for outgoing traffic were *office in the cloud* and online storage.

The second part of the research was trying to search the identified hotspots, and other known file-storage locations, for confidential information. Because most data required authentication, these end-points could barely be searched through. In the cases that it was possible to search the storage locations, interesting information could be found with Google in almost all cases.

Contents

1. Introduction

The number of solutions that allow users to create, modify and share files on the internet (or, the "cloud") has greatly increased the past few years. These solutions are often free and require no integration. This makes it easy to use, but can also lead to sharing files that a company would not want to be publicly accessible.

Beside company data being shared (accidentally or on purpose) by employees, hackers may publish stolen data. This data often comprises stolen credit card numbers or user name and (hashed) password combinations.

For companies it may be beneficial to pro-actively search for these documents, so that actions can be undertaken. In the case of an employee unknowingly sharing files, this means the documents can be taken offline, hopefully before anyone accessed them. In the case of leaked data like user names and passwords, the provided accounts can be blocked and the leak may be identified.

This research focuses on this lost data on the web. It looks for an efficient way to search for lost data and for data storage hotspots.

1.1. Research questions

The research will focus on finding confidential company data online. The research questions are as follows:

- How can confidential company data efficiently be detected on the most popular services, based on extracted company usage information?
- What online services pose the highest risk for loss of confidential data?

1.2. Related work

Previous research has been performed on how data leaves companies, and shows that it mostly happens (in 87% of the cases) due to people working at the company [3]. It furthermore focuses on prevention of data loss. This research goes into the prevention, but not into detection of data which might show up online.

Other research has been done on de-duplication in cloud storage. In this research the authors prove that it is possible to see if a specific file had been uploaded to Dropbox before [1] [2]. The last research proposed a fix to Dropbox, so whether or not the existence of files can still be proven has to be tested. This research could have impact on the actual detection of documents on data stores that use data de-duplication.

More research exists on watermarking information, which makes tampering with the data detectable, or gives proof of ownership [4].

2. Expected Results

Based on online statistics [5], the highest risk for loss of confidential data are expected to be (close to) the file sharing sites in Table 1. Beside the services named in the table, popular services such as *Google Drive* and *SkyDrive* are expected to show up.

Furthermore, data is expected to leak to text-sharing websites. Examples of these sites are *Pastebin.com, DPaste.com* and *Pastie.com*. Data that could leak to these sites includes: IP addresses, email addresses, credit card numbers and information about compromised accounts.

Social media are also a place that is expected to hold confidential information. People with open profiles could be sharing information that is confidential.

Other places information is expected to leak to are *office in the cloud* solutions. Examples of this are *Trello, Office 365, Google Docs, Evernote* and *Prezi*. Since complete documents are uploaded and edited there, these are expected to be a bigger risk. However, searching them is expected to be hard, as they require a login and don't set the document to public by default.

Lastly, documents are expected to be stored on personal storage connected to the internet (e.g. network-attached storage). These may (by accident) be open to the world.

Table 1: Top of most popular file sharing websites (January 2014)

Service	**Remarks**	**URL**
Dropbox	Requires account	https://www.dropbox.com/home
MediaFire	Requires account	http://www.mediafire.com/
4Shared	Has search	http://www.4shared.com/
ZippyShare	No registration required	http://www.zippyshare.com/
Uploaded	Requires account	http://uploaded.net/
Mega	-	https://mega.co.nz/
Depositfiles	Requires account, but allows Facebook, Google+ and Twitter login)	http://dfiles.eu/
RapidShare	Requires account	https://rapidshare.com/home
2Shared	No registration required	http://www.2shared.com/
FileFactory	Required account	http://www.filefactory.com/
ShareFile	Part of Citrix. Requires account. Trial available	http://www.sharefile.com/
SendSpace	-	http://www.sendspace.com/
FileServe	No public file sharing	http://www.fileserve.com/

3. Methods and Approach

The approach is split into two main parts: information gathering and developing a search method.

1. Determine most-used services
2. Determine keywords and identifiers for data
3. Develop a method for searching each used service type

3.1. Most-used services

There is some reference material required to build up a list of the most-used service types. Manually creating a list has the disadvantage of missing some things. Unless one is personally using all of them daily, using real-world data is paramount. Secondly, the internet environment is ever so moving that the list would be out of date in a year.

To determine the most-used websites, the logs can be filtered on most bytes sent. Users sharing confidential data will probably send this out in batch.

3.2. Keywords and identifiers

For directed searches, keywords specific to the company and its documents are required. Without specific identifiers for data, searches would be broad and will have a large amount of results. The keywords will be distilled from patterns in used documents and from filters in DLP, Data Loss Prevention, systems.

Hooks for directed searches may include the following:

- Watermarks
- File names
- Recurring content patterns
- Default disclaimers

3.3. Search method

The endpoints will be researched more in depth after it has been deducted which ones are used most. Specific to these services, a method for searching them will be developed. This method will be tailored to the keywords defined in the previous step.

4. Experiments and Data Gathering

This section describes the methods used to gather information. It starts with the method used for extracting data from the proxy, after which the search methods for different sources are described.

4.1. Proxy logs

The proxy logs contain information on the sources and endpoints of the network traffic, as well as traffic type and amount of bytes. The query as shown in Listing 1 was used to filter on outgoing traffic larger than 50KiB. This prevented normal HTTP requests from showing up as uploads, and contributing to the total traffic.

Listing 1: Query for filtering on uploads (table and column names are obfuscated)

```
SELECT destinationHostName,SUM(bytesOut) AS TotalBytesOut,
   SUM(bytesIn) AS TotalBytesIn, trafficType
FROM proxyLog
WHERE bytesOut > 50000
GROUP BY destinationHostName ORDER BY TotalBytesOut DESC
```

The query was performed on the database of the ABM AMRO bank. The proxy logged the traffic for about 23,500 employees[1]. The query was repeated daily for one week. This gives a global idea of what employees of large organizations visit on the internet.

The results were aggregated with a python script (see Appendix A) and filtered by domains.

Table 2 shows the domains to which the most traffic is sent.
The top 100 of outgoing data per domain can be found in Appendix C.

Table 2: Top sites for outgoing traffic, aggregated (one week)

Nr	Domain	MiB out
1	wetransfer-eu1.s3.amazonaws.com	14,076
2	*.mail.google.com	6,383
3	*.gateway.messenger.live.com	3,872
4	*.dropbox.com	2,636
5	*.docs.google.com	2,392
6	*.channel.facebook.com	1,388

4.2. Identifying keywords

Much of the confidential data consists of credit card numbers and bank account numbers. These patterns can be matched using regular expressions.

[1] Roughly 80% of the employees are Dutch. However, traffic from offices from around the world flows through this proxy.

Words and phrases the DLP system filters on are:

- Proprietary
- Trade secret
- Internal use only
- Not for distribution

Other words that companies often use, include "secret", "confidential" and "classified". All of which are used internationally for identifying confidential information. Most of these words can be found on the title page of a document, or in its disclaimer.

Specific words and phrases that are unique to a type of document also exist. An example of this are mortgages and their Dutch equal: "hypotheken" which contain words like *mortgage, mortgageoffer, offerte* and *hypotheek*. Different departments have different words that occur in their files.

4.3. Dropbox

As indicated by previous research, Dropbox uses de-duplication to save storage space [1] [2]. By uploading documents, network traffic would indicate whether or not the document was already on the Dropbox servers.

During the research, this experiment was repeated. A file of 50MiB with random contents was created and uploaded on one account's Dropbox folder. The same file was then uploaded on another account. The network traffic showed no indication of just a hash being uploaded, but rather the entire file.

The official Dropbox forum shows people asking about the de-duplication functionality, but it seems to have been disabled. Later posts indicate that the de-duplication has only been enabled per account [18].

4.4. Google hacking

Google is one of the biggest search engines online at this moment. Due to its massive index and constant crawling of the internet, it is a useful tool to search for lost information.

Google offers multiple operators that can be used to fine-tune searches. A complete list can be found at *GoogleGuide.com* [13]. The most important operators for searching documents are listed in Table 3.

The operators can be combined, negated (with a dash) and support logical AND (&) and OR (|) operators. This leads to queries as the one in Listing 2, which searches for all documents (pdf, doc and docx) that contain *classified* and one or both of the words *abn* and *amro*. To prevent all promotional and informational material from showing up, everything from the domain of ABN AMRO is ignored.

Table 3: Google operators

Operator	Function	Example
filetype:	Search for specified filetype	filetype:pdf
inurl:	Only show results with keyword in the URL	inurl:os3.nl
intext:	Search through text for a keyword	intext:password

Table 4: Bing operators

Operator	Function	Example
filetype:	Search for pages created in a filetype	filetype:pdf
ext:	Search for files/pages with a specific extension	ext:doc
instreamset:	Search in stream (can be combined with url)	instreamset:url:"rp.delaat.net" secret

Listing 2: Targeted Google query for PDF documents

```
abn amro "classified" -inurl:abnamro (filetype:pdf | filetype:
    doc | filetype:docx)
```

Besides Google, some other search engines support comparable operators. Bing, for instance, has the operators shown in Table 4. The full set of operators can be found at MSDN [14].

More specific Google searches and their results can be found in Appendix B.

4.5. Other file sharing

Not all sites have a built-in search capability. There are a few indexing sites, but none of them support all the popular ones. This results in having to use multiple to search them all for confidential files. See Table 5 for the search engines for popular file sharing sites (see Table 1 for the popular sites).

Table 5: File sharing search websites

Service	Supported websites	URL
FileTram	Rapidshare, Mediafire, 4Shared, 2Shared, Zippyshare and about "60" others	http://filetram.com/
FileCrop	RapidShare, Mega, Mediafire, Depositfiles, 4Shared	http://www.filecrop.com/
Sharedigger	Megashare, Rapidshare, zShare, Depositfiles and 12 others	http://sharedigger.com/
FileDiva	Combines search engines such as Google, FileTram and others	http://www.filediva.com/

4.6. Text sharing

Text-sharing websites are used to share code or other plain text data. These websites are interesting because other than their intended purpose, they are also used to make confidential data public [9].

Most websites offer a search. There are also sites that search multiple text-sharing sites at once. This offers a way to search for generic keywords but not for variable information such as login accounts, bank account numbers and e-mail addresses.

Because of the variability of interesting information on text-sharing sites, searches can not always be executed with static keywords. Instead, these searches often require regular expressions. However, neither Google nor the text-sharing websites support searching with regular expressions. There are some search websites which search through multiple text-sharing websites, such as *CanaryPW* [16], but these do not support this either. Other sites searching through text-sharing sites are scarce.

There are a few Twitter bots out there that crawl these sites based on regular expressions, and post potentially interesting pastes. Examples of such bots are *Dump Monitor* [10], *LeakedIn* [12] and *Pastebin Dorks* [15].

4.7. Social media

Information posted on social media is often shielded from people that are not in the contact list. This makes searching for posts with confidential content very difficult. However, links to confidential files might be shared via social media by posting them or sharing them via chat.

To test if these links and files are crawled/indexed (and therefore searchable by, for instance, Google), files and web pages were created on a private server. The hyperlinks to these files were shared on *Gmail, Google+ Hangouts* and *Facebook Chat*.

Afterwards, the log files on the server were monitored for accesses on the shared link. Only when the link was sent via Facebook, the link would be visited by a crawler. Facebook sent an HTTP 206 request, which is a request for partial data [7]. The request was answered with the complete document.

The files and links submitted could afterwards not be found via Facebook or Google.

4.8. Cloud offices

Searching for public documents can be done with Google hacking (see previous subsection). Searches on Prezi can be performed by specifying *inurl:prezi.com*. Prezi also offers its own search, but doesn't search "smart" like Google does. Google also tries the search with words that are similar (e.g. Netherlands, NL).

4.9. Network-attached storage

Searching specifically for Network-attached storage (NAS) systems can be done with the search engine ShodanHQ [19]. Using a query like the one in Figure 3, Dutch NAS devices connected to the internet can be found. Devices that require authentication

(HTTP 401 [7] and FTP 530 [8]) are removed from the results. However, searching for keywords is more difficult, as Shodan only indexes header information and responses instead of the content of files.

If the NAS is an open system, the folder names are exposed. These can be searched for. The catch is that it only works if employees placed documents in folders named after the company.

Searching through all files is a lot of manual labour. It also involves going through files that may not be connected to the company in question in any way.

Listing 3: ShodanHQ Query for NAS systems in the Netherlands

```
country:NL nas OR nasftpd -530 OR iomega -401
```

The same works for systems that work with FTP. These can be searched in a similar manner, as shown in Listing 4.

Listing 4: ShodanHQ Query for FTP systems in the Netherlands

```
country:NL ftp backup
```

5. Analysis

In this chapter, the results of the experiments are analysed. The chapter will give an indication in what methods for searching online are feasible and what methods are not. It will also give insight in the most used online services, which in turn may pose a threat to the loss of information.

5.1. Analysing top sites

WeTransfer, a service for sending attachments that are too big to exchange over e-mail, is on top of the outgoing list. This indicates there is a certain need for such a service. It is unknown what the nature of these files is. Offering such a service in control of the company, enables it to keep an eye on the outgoing information.

Other mail services also account for a large part of the outgoing traffic. Almost half the traffic to Google is for Gmail, as shown in Figure 2.

In addition to this, online storage services such as SkyDrive and Dropbox are popular. Due to in-place encryption, the contents of files being uploaded cannot be seen.

For all visited domains, only sent data is taken into account. More specifically, only the requests that are larger than 50KiB are taken into account. This is done to filter out all simple HTTP requests.

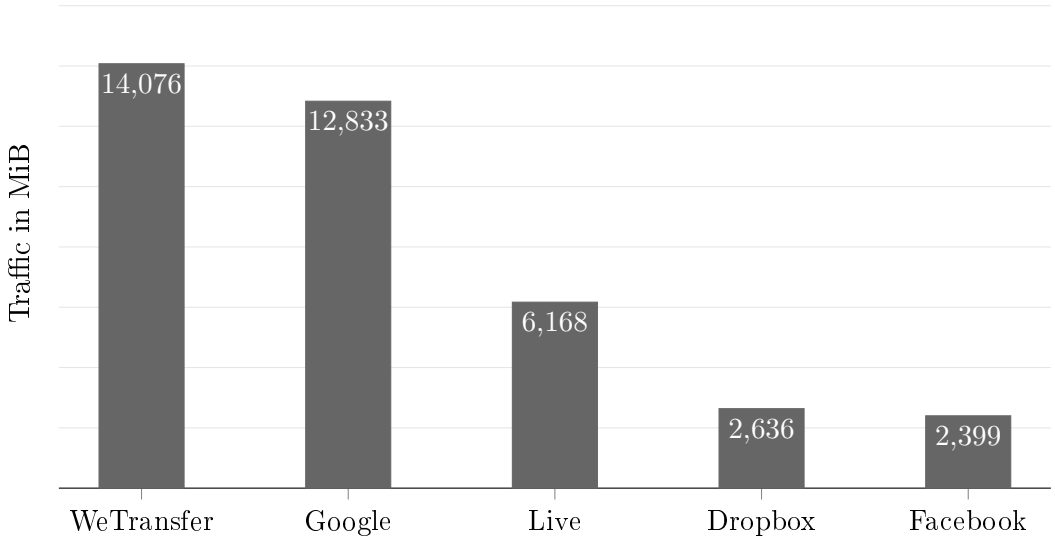

Figure 1: Outgoing network traffic aggregated by domain

The biggest part of *Live* consists of data going to **.users.storage.live.com* (known as SkyDrive) and **.gateway.messenger.live.com*, where the latter is actually data from Skype despite it's naming. Google's subservices are shown in Figure 2. The *Other* slice incorporates a few less used domains. It includes *plus.google.com* (4.43%), *www.google.com*

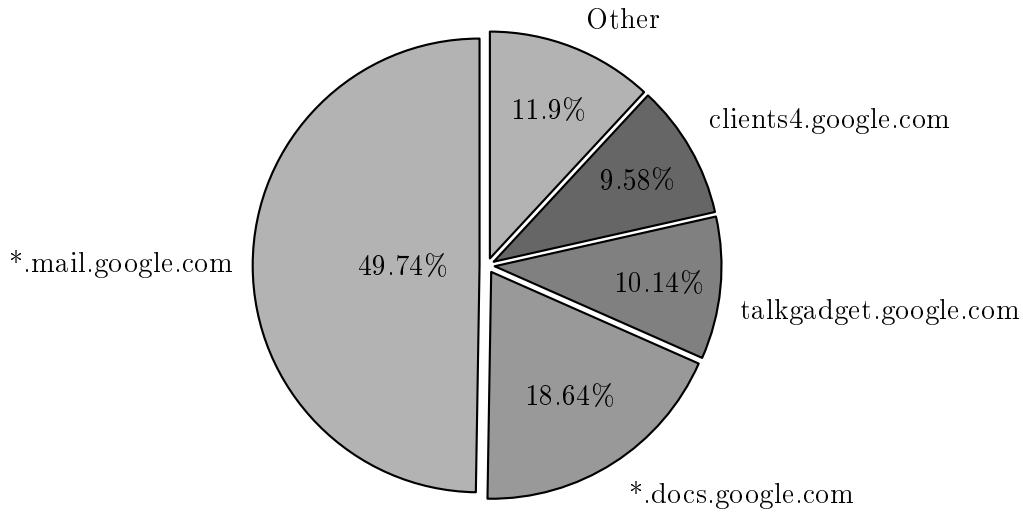

Figure 2: Google sub-services

(1.89%), **.drive.google.com* (1.44%) and *mts0.google.com* (1.30%). They do not always have to accord for all the traffic the name indicates. Since the proxy only logged HTTP traffic, Google Drive usage might lie a bit higher if a standalone client is used. It's possible this does not communicate directly over HTTP but over a regular TCP connection.

5.2. Analysing keywords and identifiers

Many of the used keywords are generic words that mark a documents classification. These can be used in combination with the name of the company to get matching results.

Other identifiers, such as bank account numbers and credit card numbers, can be matched with regular expressions. However, search engines don't usually support regular expressions. This means a special crawler would have to be coded.

The more specific identifiers are, the better the results will be. Broad identifiers, or identifiers that are used often, are less likely to return good results.

5.3. Dropbox

The side channel attack on Dropbox that used de-duplication to check the existence of files that are not in the users own space is not possible any more. Because Dropbox requires logging in for all other non-shared files, these cannot be searched for.

Searching for publicly shared files is still possible, but requires a regular search. Downloading these files automatically and checking if they match a confidential document would require a lot of bandwidth. Checking them automatically is possible, but remains infeasible.

5.4. Google hacking

Google hacking offers the best possibilities in terms of searching through (Google's) indexed data. It also offers a few capabilities of searching through websites that do not offer a search functionality themselves. Public links to shared files can be found with a bit of tweaking. The downside is the lack of regular expressions which would make searching for a range of possible keywords easier.

Certain queries resulted in a few phishing sites for banking. Some of these were not marked as malicious sites.

5.5. Other file sharing sites

It is not possible to search for documents other than by its name. Also, it is not possible to search for meta data or within a file. DLP systems use hashes that match (partial) file content, but no feature of the built-in search comes close. It's not viable to download every file and compare the hash or search through it.

5.6. Text sharing

It's difficult to find some interesting data. The search sites do not support regular expressions, which only leaves searching for static keywords. Although the bots use regular expressions, searches on the sites are restricted to keywords. As an alternative to these bots, it is possible to develop or run an indexer in the organization itself, but there are a few things to consider such as legal and ethical issues. Getting every paste also fills up storage space quickly, let alone polling every paste which might get one banned. A couple examples of open source indexers are *Dump Monitor* [11] and *Pystemon* [17].

5.7. Social media

Sharing a hyperlink via Gmail or Google Hangout does not cause it to be indexed and available for search on first look. Facebook is on the top of used social media websites, followed by Google+ and Twitter. While it is possible to upload files to some of them, currently there is no other indication that they are being used for other things than their intended purpose. They are presumably just used for sharing statuses and uploading/viewing pictures.

5.8. Cloud offices

Applications such as Google Docs are on top of most used in the cloud, as shown by the proxy data. A survey could provide insight on the popularity of similar services. Keeping the data within the company would require to have a similar product provided by the company. Presumably, the main reason to use these specific services is their ease of use and quality, usually not matched by products offered by off the shelf products.

5.9. Network-attached storage

Finding private network-attached storage servers on the internet appears not too difficult. The question here lies with the legal aspect of it. It is debatable if it's a breach of privacy when looking at these open directories or files. It would also require going through a lot of files, scripted or by hand, because the contents are not indexed.

6. Conclusion

Based on the gathered information, the following conclusions can be drawn.

How can confidential company data efficiently be detected on the most popular services, based on extracted company usage information?
Detecting the existence of information on the most used online services is very difficult. Nearly all of these services require the user to log in. Because of this, the information is invisible for normal searches. The documents can also no longer be detected with a de-duplication check.

Finding documents and other information online can most effectively be done with Google hacking. Google is the most versatile search engine that can be used in combination with logical operators. This way, the best queries can be created. There is however one drawback to Google, and many others, there is no option to use regular expressions. In the case of a bank, these regular expressions can be important for finding account numbers.

There are many smaller sites where files can be stored. While some search engines specifically search these, searching all of them is a difficult challenge. The same goes for text-sharing sites. There are online bots that specifically search these sites for dumped information.

What online services pose the highest risk for loss of confidential data?
The traffic from a large bank (ABN AMRO) has been analysed to identify destinations of larger uploads. The results show that uploading of data consistently happens to private mail addresses on Google and Live mail. It also showed that employees perform work on documents in the cloud (e.g. Google docs, Prezi).

One service that stands out is a site that offers emailing of larger attachments, WeTransfer. Unless the paid version is used, attachments are unprotected, although the link to the file consists of two hashes.

A second risk are the services offered by Google. Many employees use these services for convenience, most notably Gmail. The same goes for Live mail from Microsoft, even though less traffic is sent there.

It can be concluded that most data leaves the company because of employees using convenient methods to work on- and with data.

A possible solution would be to create more awareness about data loss among employees. Implementing technical barriers will only lead to users trying to circumvent these [6]. As a side risk, the methods that users will choose may be less secure and increase the risk overall.

A second option would be to create an in house solution like WeTransfer. If employees can be convinced to use this instead of the online service, the data will stay in the hands of the company. It will also give insight in who downloads the files.

7. Suggestions for future work

The results in this document do not cover all of the used services, merely the most popular ones. While this covers the most common services that can be set for example, neglected ones may be able to rise in popularity in the coming years. One can hypothesize that these can be anticipated on to provide a proactive solution.

This research is also based on proxy data by a large Dutch organisation. Similar research may be conducted in other countries, which may show difference in popularity of sites.

References

[1] D. Harnik et. al., *Side channels in cloud services, the case of deduplication in cloud storage*, 2010
http://www.pinkas.net/PAPERS/hps.pdf

[2] M. Mulazzani et. al., *Dark Clouds on the Horizon: Using Cloud Storage as Attack Vector and Online Slack Space*, 2011
https://www.usenix.org/legacy/event/sec11/tech/full_papers/Mulazzani6-24-11.pdf

[3] K. Leung, *Information Leakage & Data Loss Prevention*, 2009
http://uwcisa.uwaterloo.ca/Biblio2/Topic/KarenKarYanLeung.pdf

[4] V. Shobana and M. Shanmugasundaram, *Data leakage detection using cloud computing*, 2013
http://www.ijetae.com/files/Conference%20ICISC-2013/IJETAE_ICISC_0113_17.pdf

[5] Ebizma, *Top 15 Most Popular File Sharing Websites | January 2014*, Accessed 2014-01-11
http://www.ebizmba.com/articles/file-sharing-websites

[6] A. Adams and M.A. Sasse, *Users are not the enemy*, 1999
http://discovery.ucl.ac.uk/20247/2/CACM%20FINAL.pdf

[7] R. Fielding et. al., *Hypertext Transfer Protocol – HTTP/1.1*, 1999
https://tools.ietf.org/html/rfc2616

[8] J. Postel and J. Reynolds, *FILE TRANSFER PROTOCOL (FTP)*, 1985
https://tools.ietf.org/html/rfc959

[9] BBC: L. Kelion, *Pastebin: Running the site where hackers publicise their attacks*, 2012 - Accessed 2014-01-23
http://www.bbc.co.uk/news/technology-17524822

[10] Dump Monitor, *Hi there! I'm a bot which monitors multiple paste sites for password dumps and other sensitive information.*, Accessed 2014-01-13
https://twitter.com/dumpmon

[11] Information Dump Monitor, *Twitter-bot which monitors paste sites for interesting content*, Accessed 2014-01-16
https://github.com/jordan-wright/dumpmon
http://raidersec.blogspot.nl/2013/03/introducing-dumpmon-twitter-bot-that.html
http://www.reddit.com/r/netsec/comments/1i2926/canary_a_search_engine_for_data_mined_pastebin/cb0fig3

[12] LeakedIn, *Stories About Data Leaks and Related Stuff*, Accessed 2014-01-14
http://www.leakedin.com/
https://twitter.com/leakedin

[13] GoogleGuide Search Operators, *Making searching even easier*, Accessed 2014-01-15
http://www.googleguide.com/advanced_operators_reference.html

[14] Microsoft Developer Network, *Advanced Operator Reference*, Accessed 2014-01-23
http://msdn.microsoft.com/en-us/library/ff795620.aspx

[15] Pastebin Dorks, *I'm a bot in alpha, posting links to potentially interesting pastebin posts*, Accessed 2014-01-15
https://twitter.com/Pastebindorks
http://www.exploitaration.net/projects/pastebin/

[16] CanaryPW, *A search engine designed to look through text that has been publicly posted on services like Pastebin.*, Accessed 2014-01-15
https://canary.pw/
http://www.reddit.com/r/netsec/comments/1i2926/canary_a_search_engine_for_data_mined_pastebin

[17] pystemon, *Monitoring tool for PasteBin-alike sites written in Python*, Accessed 2014-01-16
https://github.com/cvandeplas/pystemon

[18] Dropbox, *Deduplication - Reports on privacy issues*, Accessed 2014-01-21
https://forums.dropbox.com/topic.php?id=36365

[19] ShodanHQ, *Expose Online Devices*, Accessed 2014-01-23
http://shodanhq.com

Appendices

A. Python script for aggregating proxy data

```
# Logger CSV export domain aggregator
#
# Jan-Willem Selij
# Initial idea by Peter van Bolhuis written in bash

import csv
import re


class AggregateResults:

  def __init__(self):
    # Format: [domain, bytes_out, bytes_in]
    self.results = []

    # For aggregating domains into one
    # Format: [match regex, replace string|[1, 2, ...]]
    # let replace string be a list with strings to dupe output
    # I used [+] to indicate this is a domain that has
       subdomains
    # aggregated (and therefore can't be counted as extra data)
    self.domain_conversions = [
      [r"^$", "(unknown)"],  # Blank domain
      [r".*\.dropbox\.com", "*.dropbox.com"],
      # Skydrive, Skype
      [r".*\.gateway\.messenger\.live\.com", "*.gateway.
         messenger.live.com"],
      [r".*\.users\.storage\.live\.com", "*.users.storage.live.
         com"],

      # Google mail, docs, drive subdomains (like b.mail.google
         .com)
      [r"(.*\.)?mail\.google\.com", "*.mail.google.com"],
      [r"(.*\.)?docs\.google\.com", "*.docs.google.com"],
      [r"(.*\.)?drive\.google\.com", "*.drive.google.com"],


      [r".*\.channel\.facebook\.com", "*.channel.facebook.com"
         ],
      #[r".*\.facebook\.com", "[+] facebook.com"],

      # Aggregate everything from Google's domain
      # Also output the original domain instead of sucking it
         up
      # with \g<0>. Downside is that it isn't catched anymore
         with
```

```
    # the subdomain regexes.
    # Alternative would be adding a second domain to each of
      the Google
    # subdomains, but then we would have to know each and
      every one of
    # them.

    #[r".*\.google\.com", [r"\g<0>", "[+] google.com"]],
    #[r".*\.youtube\.com", "[+] youtube.com"],

    #[r".*\.live\.com", "[+] live.com"],

    [r".*\.drip\.trouter\.io", "*.drip.trouter.io"],
    [r".*\.webex\.com", "*.webex.com"],

    #[r"(.*\.)?good\.com", "*.good.com"]


  ]

  self.post_aggregates = [
    [r".*\.live\.com", "[+] live.com"],
    [r".*\.google\.com", "[+] google.com"],
    [r".*\.youtube\.com", "[+] youtube.com"],
    [r".*\.facebook\.com", "[+] facebook.com"]


  ]

  # Category stats
  self.total_lines = 0
  self.categories = []
  self.category_stats = {}

# maybe a little slow checking each and every domain
# Returns a list of converted domains
def aggregate_domain(self, domain):
  domains = []
  for domain_conversion in self.domain_conversions:
    # Does it match a conversion?
    conversion_match_regex = domain_conversion[0]
    conversions = domain_conversion[1]

    if re.match(conversion_match_regex, domain):
      #print "domain matched ->", domain
      if not isinstance(conversions, list):
        conversions = [conversions]
      for conversion_item in conversions:
        # Replace into an aggregate version
```

```
            domains.append(re.sub(conversion_match_regex,
                conversion_item, domain))

    if domains:
      return domains

    # Return the domain if not matched to anything
    return [domain]

  # Get post aggregate from post_aggregates
  # False if non-existing
  # only matches one in the list (from top to bottom)
  def get_post_aggregate(self, domain):
    for domain_conversion in self.post_aggregates:
      # Does it match a conversion?
      conversion_match_regex = domain_conversion[0]
      conversion = domain_conversion[1]

      #print "Matching " , domain,  "with",
          conversion_match_regex
      if re.match(conversion_match_regex, domain):
        return re.sub(conversion_match_regex, conversion,
            domain)

    return False

  # False: no index, new one
  def find_domain_index(self, domain):
    for position, domain_item in enumerate(self.results):
      # 0 = domain name
      if domain_item[0] == domain:
        return position

    return False

  def add_domain_stats(self, domain, bytes_out, bytes_in):
    """Directly adds domain stats, no aggregating here."""

    domain_index = self.find_domain_index(domain)

    if not domain_index:
      self.results.append([domain, int(bytes_out), int(bytes_in
          )])
    else:
      #print "Added bytes to existing domain ->", domain
      former_domain_stats = self.results[domain_index]
      former_domain_stats[1] += int(bytes_out)
      former_domain_stats[2] += int(bytes_out)
```

```
def sort_results(self, sort_type, descending=True):
  if sort_type not in ["domain_name", "bytes_out", "bytes_in"
      ]:
    print "Can't sort by '{0}'.".format(sort_type)

  if sort_type == "bytes_out":
    sort_key = lambda x: int(x[1])
  elif sort_type == "bytes_in":
    sort_key = lambda x: int(x[2])
  elif sort_type == "domain_name":
    sort_key = lambda x: x[0]

  return sorted(self.results, key = sort_key, reverse=
      descending)

# Should actually be a generic function that parses the CSV,
    other
# functions can then operate on the data
def xadd_csv(self, csv_file):
  with open(csv_file, "rU") as csvfile:
    statsreader = csv.reader(csvfile)
    for row in statsreader:
      # Doesn't work if it does not have the exact amount of
          values
      #domain, bytes_out, bytes_in, category, x, y, z = row
      domain = row[0]
      bytes_out = row[1]
      bytes_in = row[2]
      category = row[3]
      device = row[4]

      # Negate lines that aren't log lines
      if device != "Blue Coat":
        continue

      self.total_lines += 1

      if category not in self.categories:
        self.categories.append(category)
        #self.category_stats[category] = {"name": category, "
            count": 1}

        # [category, amount, percent of total]
        self.results.append([category, 1])
      else:
        # abusing it to find the category, picks first
            element which is the category
        category_index = self.find_domain_index(category)
```

```
            self.results[category_index][1] += 1
            amount = self.results[category_index][1]

            # Percent, not working
            #self.results[category_index][2] = round((amount /
                self.total_lines) * 100, 2)
            #print self.results[category_index][2]
            #self.category_stats[category]["count"] += 1

    def lines_stat(self):
      print "Total:", self.total_lines
      # abusing the sort method, sorts second item which is the
          amount
      print self.sort_results("bytes_out")
      #print self.category_stats

    # Expects domain, bytes out, bytes in, category, device
    def add_csv(self, csv_file):
      with open(csv_file, "rU") as csvfile:
        statsreader = csv.reader(csvfile)
        for row in statsreader:
          # Doesn't work if it does not have the exact amount of
              values
          #domain, bytes_out, bytes_in, category, x, y, z = row
          domain = row[0]
          bytes_out = row[1]
          bytes_in = row[2]
          category = row[3]
          device = row[4]

          # Negate lines that aren't log lines
          if device != "Blue Coat":
            continue

          #print domain, bytes_out

          # Convert into aggregate domain(s)
          domains = self.aggregate_domain(domain)

          for domain in domains:
            self.add_domain_stats(domain, bytes_out, bytes_in)



    def show(self):
      print self.results

    def export(self, file_name):
      #post_aggregates = []
```

```
    for domain in self.results:
      post_aggregate = self.get_post_aggregate(domain[0])
      if post_aggregate:
        #post_aggregates.append(post_aggregate)
        self.add_domain_stats(post_aggregate, domain[1], domain
            [2])

    #print "Post aggregates ->", post_aggregates



    results = self.sort_results("bytes_out")

    with open(file_name, "wb") as csvfile:
      csvwriter  = csv.writer(csvfile)

      csvwriter.writerow(["Domain", "Bytes Out", "Bytes In", "
          MB Out", "MB In"])
      for stat in results:
        stat = stat + [stat[1]/1024/1024, stat[2]/1024/1024]
        csvwriter.writerow(stat)

  def export_stats(self, file_name):
    results = self.sort_results("bytes_out")

    with open(file_name, "wb") as csvfile:
      csvwriter  = csv.writer(csvfile)

      csvwriter.writerow(["Category", "Lines"])
      for stat in results:
        #stat = stat + [stat[1]/1024/1024, stat[2]/1024/1024]
        csvwriter.writerow(stat)


stats = ["Get all uploads (bytesOut) - 50 kB and filter on File
    StorageSharing_01-18-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-19-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-20-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-21-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-22-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-23-2014-08-00-00.csv",
"Get all uploads (bytesOut) - 50 kB and filter on File
   StorageSharing_01-24-2014-08-00-00.csv"
]
```

```
aggregated = AggregateResults()

for stat in stats:
  aggregated.add_csv(stat)
  #aggregated.xadd_csv(stat)

#aggregated.lines_stat()
#aggregated.export_stats("lines_stat.csv")


aggregated.export("results.csv")
print aggregated.sort_results("bytes_out")[:25]   # 25 top
    results
```

B. Google hacking results

Note that with the **Page/Results** below, this was true at the time of writing for the author. However, the location of the results may differ per person trying this. Google orders results different per user based on previous searches

Table 6: Google results

Documents	Pension/Salary information
Search Pattern	filetype:doc | filetype:docx | filetype:pdf AND ("abn amro" OR "abnamro") AND (-inurl:abn OR -inurl:abnamro) "overgemaakt op rekeningnummer *" "Sofinummer"
Page/Result	1/all
Documents	Confidential documents
Search Pattern	filetype:doc | filetype:txt | filetype:pdf AND ("abn amro" OR "abnamro") AND (-inurl:abn OR -inurl:abnamro) vertrouwelijk
Page/Result	1/1, 1/2, 1/3, 1/4, 1/10
Documents	Web folders which may include forged login pages
Search Pattern	intitle:index.of abn amro -inurl:"payments/abnamro" -jpg
Page/Result	1/9, 4/2

C. Outgoing data per domain results

Aggregated domains are indicated by the [+] symbol. Several aggregated subdomains are indicated by the wildcard character (*).
Multiple services such as *good.com* and *webex.com* are left out in the findings, because they are for internal usage. These services still show up in these logs because they are hosted by an external company.

Table 7: Outgoing data per domain, top 100

Domain	MiB Out	MiB In
www.abnamro.nl	26,963	42,085
xml30.good.com	21,899	19,418
(unknown)	21,406	21,805
*.webex.com	20,846	20,619
gti01.good.com	17,172	14,160
wetransfer-eu1.s3.amazonaws.com	14,076	10,735
[+] google.com	12,833	14,048
upl01.good.com	12,365	0
upl01.good.com	7,463	0
*.mail.google.com	6,383	7,598
[+] live.com	6,168	6,082
demo.abnamro.nl	5,186	5,724
abnamro.demo.fx.com	4,312	3,918
*.gateway.messenger.live.com	3,872	3,854
upl01.good.com	3,740	0
ch1hub.cognizant.com	3,030	3,030
treasurykoersen.abnamro.nl	2,876	5,168
collab.thomsonreuters.com	2,815	2,464
*.dropbox.com	2,636	2,294
[+] facebook.com	2,399	2,873
*.docs.google.com	2,392	2,019
*.drip.trouter.io	2,000	1,996
matrix.ms.com	1,631	1290
[+] youtube.com	1,428	1,425
login.youforce.biz	1,422	1,154
upload.youtube.com	1,394	1,390
*.channel.facebook.com	1,388	1,381
talkgadget.google.com	1,302	1,157
www.intermediair360.nl	1,273	1,267
trello.com	1235	1,062
clients4.google.com	1,229	1,267
ftpsite.vmware.com	1,116	535
abnnl35.uat.fx.com	1,098	981
iebs.icap.com	1,068	860
wgeu.marketaxess.com	1,066	988
www-et2.abnamro.nl	965	2,064
www.uitgesprokentalent.nl	963	2,091
sdts.ema.kpmg.com	953	0
pro3.live.fxinside.net	900	1,113
www-et1.abnamro.nl	890	2,080
dpsw.info	807	803

Domain	MiB Out	MiB In
www.accessonline-et.abnamro.com	776	962
sip.reuters.net	690	715
support.oracle.com	687	457
1511493172.cloud.vimeo.com	653	0
1511493180.cloud.vimeo.com	653	0
www.facebook.com	624	1,098
login.binck.nl	602	533
public-ftp.mosaicfinance.fr	592	472
plus.google.com	568	663
clearing.flowtraders.nl	532	416
outlookweb.eur.nl	521	523
ftp.tower-research.com	503	396
www.accessonline.abnamro.com	473	751
stream22prod.citivelocity.com	472	472
www2.saxowebtrader.com	466	420
s3.amazonaws.com	421	362
www.csia.in	397	397
stex.360t.com	377	411
www.google.nl	374	775
ftp-abnamroeur.jumptrading.com	353	276
mail.cognizant.com	340	287
access.rbsm.com	333	420
quadia.webtvframework.com	330	0
apothema.imc.nl	328	257
apothiki.imc.nl	327	257
www.pwmconnect.abnamro.com	315	310
sc2.omniture.com	312	382
tfs.rhea.infosupport.net	307	522
bankieren.mijn.ing.nl	300	290
arena.abnamro.org	281	481
www.aab-hypotheken-business.nl	280	209
www.randstad.nl	275	322
hm.highfive.nl	252	284
vpn2.4sight.com	244	453
secure.autobahnfo-uk.db.com	243	206
www.google.com	241	520
www.mijnpensioenoverzicht.nl	237	237
www.asrcockpit.nl	233	265
ftp.rgmadvisors.com	224	175
fv-zprod-tc-0.farmville.com	220	200
prod4.rest-notify.msg.yahoo.com	216	169
abnamro.stockloan.net	212	316
abnacc.acadooacademy.com	210	54
www.debtdomain.com	200	154

Domain	MiB Out	MiB In
xnet.infosys.com	199	326
www.neuflizeobcinvestissements.fr	188	0
*.drive.google.com	185	232
www.boursorama.com	182	172
login1.youforce.biz	179	228
1511493168.cloud.vimeo.com	176	0
dropbox.qrm.com	175	0
www-et4.abnamro.nl	175	551
fwd106.livemeeting.com	173	172
eccgw01.boulder.ibm.com	172	164
mts0.google.com	167	520
realtime.icapfusion.com	165	164
www.ergv.euroclearfrance.com	165	192
www.citivelocity.com	162	174