Symantec™
Security Response

Have I Got Newsforyou: Analysis of Flamer C&C Server

Symantec Security Response

Contents

Overview

W32.Flamer is a sophisticated cyber espionage tool that targeted the Middle East. It is modular in design and contains some novel functionality, most notably its ability to spread across networks using a previously unknown man-in-the-middle attack against Windows Update. Symantec has performed a detailed forensic analysis of two of the command-and-control (C&C) servers used in the W32.Flamer attacks from earlier this year.

Based on our analysis, we were able to uncover details such as when the servers were operational, what entities were targeted, nicknames of those involved in the attack, and techniques used by the attackers to avoid discovery should the command-and-control server be compromised.

Analysis of these C&C servers was performed as a joint effort between Symantec, CERT-Bund/BSI, IMPACT, and Kaspersky. This paper focuses on the detailed forensic examination Symantec carried out on the C&C server images.

Background

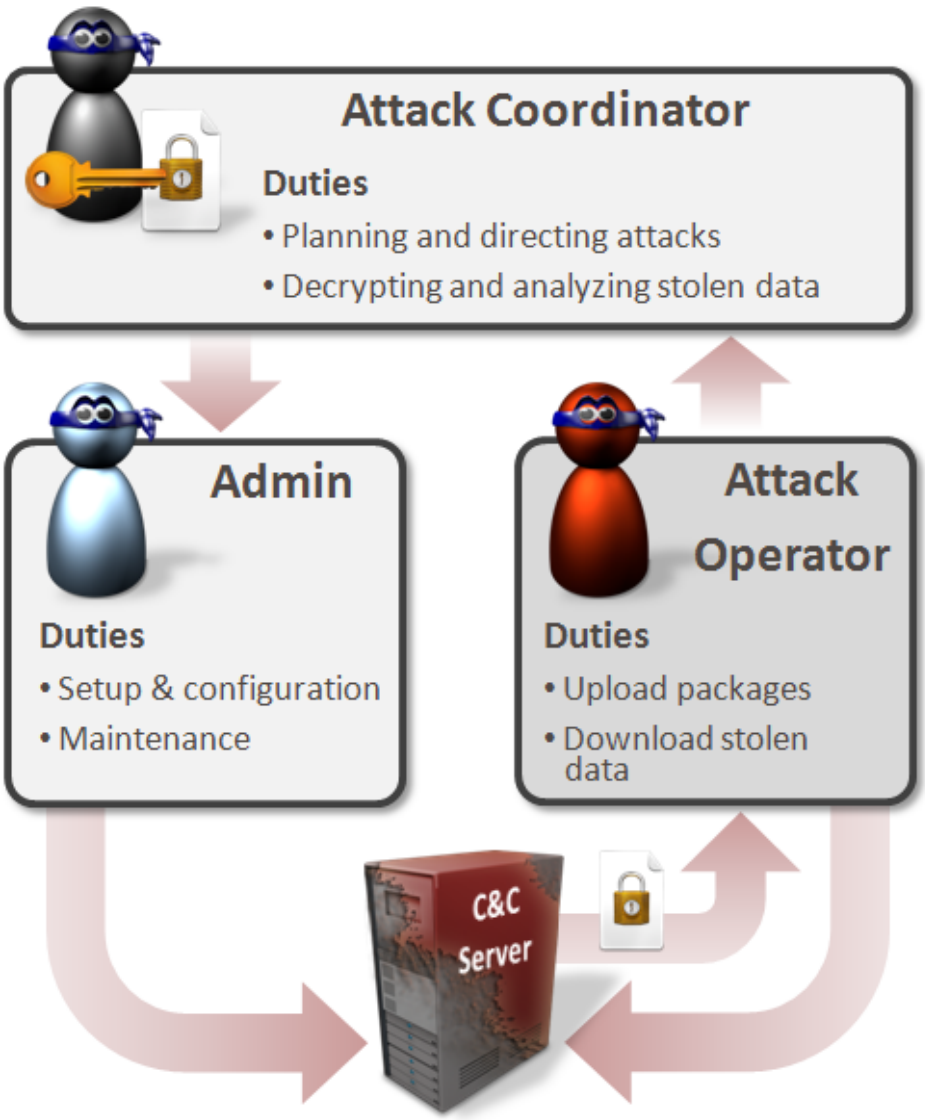
The first server was set up on May 18, 2012, and, just five hours after it was set up, it recorded the first interaction with a Flamer-compromised client. The server would go on to control at least a few hundred compromised clients over the next few weeks. The second server was set up on March 25, 2012, and controlled over a thousand clients in a period of just over one week.

The servers had been set up to record a minimal amount of information in case of discovery. The systems were configured to disable any unnecessary logging events and entries in the database were deleted at regular intervals. Existing log files were securely deleted from the servers on a regular basis. These steps were taken in order to hamper any investigation should the server fall into the hands of investigators or law enforcement.

However, the attackers were not thorough enough, as files revealing the entire history of the servers' setup were available and a limited set of encrypted records in the database revealed that compromised clients had been connecting from the Middle East. We were also able to recover the nicknames of four authors—D***, H*****, O******, and R***—who had worked on the code at various stages and on differing aspects of the project.

Figure 1

Data security compartmentalization used by W32.Flamer attackers



On both servers command-and-control activity happens through a Web application called Newsforyou. It processes the W32.Flamer client interactions and provides a simple control panel. The control panel allows the attackers to upload packages of code to deliver to compromised clients, and download packages containing stolen client data. However, in a technique not previously seen before the uploaded and downloaded packages are encrypted, so infiltrating the command-and-control server does not reveal the code or the stolen client data. The command-

and-control server simply serves as a proxy for the data and the data is encrypted and decrypted offline by the attackers using keys unique to each client. This application also contains functionality to communicate with clients compromised by malware other than Flamer. The Web application was designed to be a framework for supporting different malware campaigns.

In addition to avoiding the compromise of their operations, preventing both the uploading of rogue code and viewing of stolen data, the setup also maintains a clear distinction of roles. The roles include those responsible for setting up the server (admins), those responsible for uploading packages and downloading stolen data through the control panel (operators), and those holding the private key with the ability to decrypt the stolen data (attack coordinators). The operators themselves may actually be completely unaware of the contents in the stolen data. This is due to design of the process to use data security compartmentalization techniques, as shown in Figure 1.

Despite these techniques, we were still able to determine that one of the servers delivered a module instructing Flamer to commit suicide and wipe itself off computers in late May 2012, an action we also witnessed through compromised honeypots.

Finally, access to the control panel required a password which is stored as a hash. Despite brute force attempts at reversing the hash to plain text, we were unable to determine the plain text password.

What follows is a thorough analysis of one of the command-and-control servers, detailing the server setup, the Web application developed by at least four separate authors since 2006, the control panel used by the operators, and the database that helps drive the application. The report also includes key information about a second server, but in-depth details have been omitted to maintain the brevity of this report. The functionality and structure of both servers is identical. The differences are mainly in the amount of data and number of clients the servers were exposed to.

The server

File system

The following table details the important locations relevant to the investigation of the command-and-control servers.

Table 1

Location	Description
/root/.bash_history	Contains a history of root user commands
/var/spool/crontabs/root	Scripts set up to run at regular intervals
//home/[USERNAME]	Operator home directory, contains various scripts
/var/www/htdocs/newsforyou	Command-and-Control Web application
/var/lib/mysql	MySQL database

Server setup

The first sign of activity from the attackers on the first server was May 18, 2012. At 11:26 (UTC) the first scheduled job was run and a little over two hours later the server was fully operational. A malicious package was uploaded to the server at 13:53. The first recorded interaction with a W32.Flamer compromised client was at 16:15, when stolen data was uploaded to the server.

In comparison, the second server saw initial activity on March 25, 2012, and was fully operational on the same day.

During the server set up, the attackers installed various applications and tested connections to the MySQL database which supported their custom Web application. The end goal was to install a command-and-control application called Newsforyou which interacted with compromised W32.Flamer clients.

An important step taken by the attackers was to hide traces of their activity on the server. They did this by disabling logging and securely removing existing log files. The attackers, however, did not clear out the history of commands issued through the console by the administrator (root) account. This data was visible and was retrieved from the /root/.bash_history file, which revealed these activities. The following table lists and describes, in brief, a subset of commands that are most relevant to the server set up:

Table 2

Command Executed	Description
netstat –an	Checks open network connections
telnet localhost 3306	Tests MySQL server is accepting connections
nano apache2.conf	Configures Web server
adduser [USERNAME]	Adds user [USERNAME]
crontab /etc/cron.newsforyou	Cronjob executes scripts at regular intervals
openssl req -newkey rsa:1024 -nodes -x509 -days	Creates SSL key
python __main__.py 0 2 1 "127.0.0.1" 100 100	Simulator test to add entry (stolen data)
python __main__.py 2 1-2 1 "127.0.0.1" 100	Simulator test to retrieve news entry (payload)
nano /etc/apache2/ports.conf	Configures Web server to listen on TCP port 443
cp /home/[USERNAME]/LogWiper_fixed.txt /LogWiper.sh	Prepares log wiper file as BASH script
sh /LogWiper.sh	Wipes logs and disables logging services
mkdir /var/www/common/	Creates folders to be used in redirection
mkdir /var/www/wp-content/	Creates folders to be used in redirection
mkdir /var/www/pages/	Creates folders to be used in redirection
mkdir /var/www/services/	Creates folders to be used in redirection
nano /etc/apache2/sites-enabled/default-ssl	Sets up URL redirects
mkdir -p /var/www/cgi-bin	Creates folders to be used in redirection
mkdir -p /var/www/htdocs/newsforyou	Create Web appplication folder
php5 ../DB_creation_script.php	Creates database
/etc/init.d/apache2 restart	Restarts Web server
iptables -A INPUT -p tcp --dport 22 -j ACCEPT	Accepts connections on TCP port 22(SSH)
iptables -A INPUT -p tcp --dport 443 -j ACCEPT	Accepts connections on TCP port 443(HTTPS)

Following these commands, the Apache server was configured to listen on TCP ports 443 and 8080. The firewall was also reconfigured to allow connections over SSH and HTTPS. Logging activities were disabled and existing log files were securely deleted where necessary. The server is now ready to interact with compromised Flamer clients through the Newsforyou application.

The attackers also set up some URL redirections to disguise the true nature of the requests to an inexperienced eye. They would appear to look like legitimate requests for regular looking folder names. An example of one of the redirect rules is shown below:

/etc/apache2/sites-available/default-ssl

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/
<Directory "/var/www/cgi-bin">
      AllowOverride None
      Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
      Order allow,deny
      Allow from all
      RewriteEngine on
      RewriteRule ^counter\.cgi$ /newsforyou/index.php
</Directory>
```

Scheduled tasks

A cronjob was set up to periodically delete files from the file system and remove older entries from the database:

/etc/cron.newsforyou

Table 3

Repeat Interval	Command
2 minutes	/var/www/htdocs/newsforyou/UnloadChecker.php >> /var/log/newsforyou.log
6 hours	* python /home/[USERNAME]/pycleaner/Eraser.py
@Midnight	php /home/[USERNAME]/delete.php

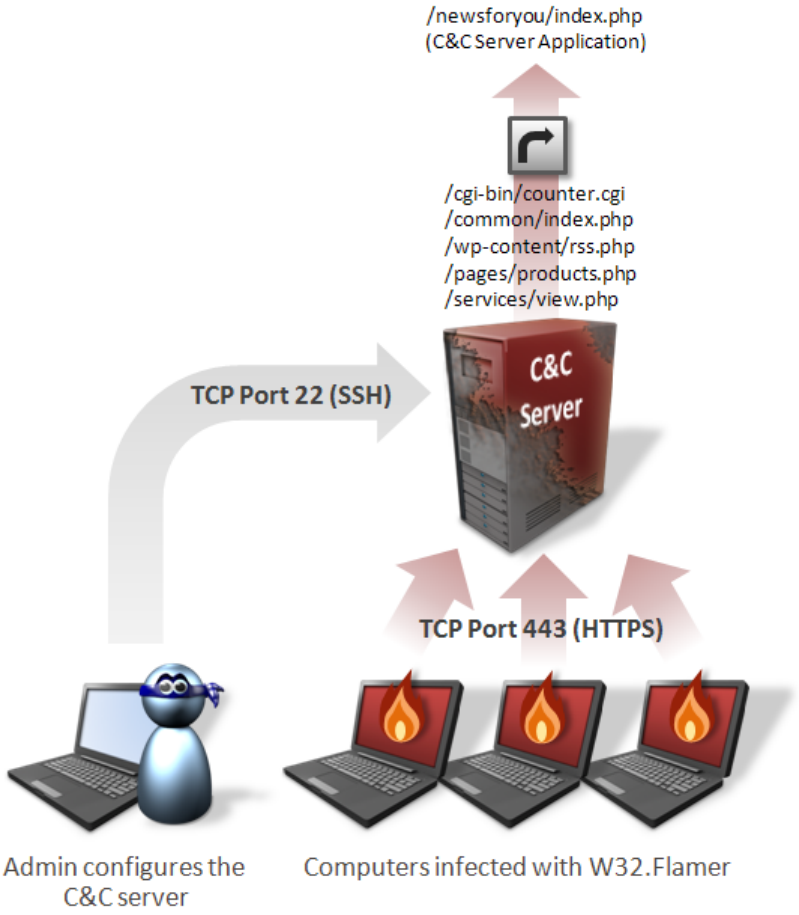
UnloadChecker.php, which is executed every two minutes, retrieves data uploaded from the compromised computer and places it in a tar archive. The archive could then be downloaded by the operators. Eraser.py is a script used to wipe certain files and content from folders. On one of the servers the operators made an error when setting up the scheduled task. They used the folder name pycleaner when creating the task, but this folder does not exist. The folder containing the script is pycleanscr. As a result, the Eraser.py script was never automatically executed.

Figure 2 illustrates the simple setup now in place for communicating with W32.Flamer compromised clients.

The next sections discuss the files and applications of most interest to the investigation, along with their locations. The analysis begins in the home directory of the unique added user, a folder that is created once the user is added during the initial set up. The usernames here were

Figure 2

Communications between C&C server and compromised computers



short three-letter names, unique to each server.

The home directory

Location: /home/[USERNAME]

The following table contains the location and a brief description of the files and folders present in the user's home directory:

Table 4

File/Directory	Description
./.bashrc	Standard .bashrc with no modifications
Simulator/	Python application used to test the Newsforyou application
pycleanscr/	Python application used to free up disk space if necessary
./LogWiper_fixed.txt	Script which disables logging and securely deletes specific log files
./RequestHandler.php	Used in the Newsforyou Web application
./Delete.php	PHP script which delete files and entries in the MySQL database

Stress testing the application

Simulator

This application consists of a set of Python scripts, which are used to stress test the Newsforyou application. This application was only present on one of the servers. The application connects to a chosen server and issues various queries that conform to the C&C server protocol. The tests performed during the set up were to add an entry (stolen data) and to retrieve a news entry (payload).

A point of interest here is the presence of dnslocation.info as part of the HTTP request header the script builds in a file named Connection.py. This is a known Flamer C&C server domain, indicating that the code is being shared and used to test across various C&C servers. The timestamp on this file supports this idea as it is stamped March 22, 2012.

```
def CreateHeaders(self):
    headers = {"Accept": "*/*", "Host": "dnslocation.info", "Connection":
    return headers

def CreateParams(self, data):
    dataLen = len(data)
    params = urllib.urlencode({'UNIQUE_NUMBER': self._uniqueNumber, 'PASS'
    return params
```

When a Python script is executed, a compiled Python file (.pyc) is generated. Examining these files, we confirmed this application was executed on May 18 which corroborates our finding about the system configuration date.

Cleaning up

pycleanscr

pycleanscr checks how much free disk space is available on the root partition. It deletes the files in /var/www/htdocs/newsforyou/tmp/ if the amount of the disk space in use is greater than 75 percent.

Delete.php

Deletes entries in the **news_entries** table of the MySQL database and securely deletes the files referenced by that entry on disk that are older than 30 days.

Disabling logging

LogWiper_fixed.txt

This .txt file is renamed to a .sh BASH script and run to disable logging services, to securely delete any prior log files created, and to disable logging in two particular applications required by the attackers: Apache and SSH.

Table 5

Log files deleted	Services disabled	Services with logging disabled
/var/log/wtmp	Rsyslog	Apache
/var/log/lastlog	Sysklogd	SSH
/var/run/utmp	Msyslog	
/var/log/mail.*	syslog-ng	
/var/log/syslog*		
/var/log/messages		
hidden files in /root and /home		
/var/log/auth/*		
/var/log/apache2/*		

Files are securely deleted using Shred, a tool that repeatedly overwrites files to prevent their recovery, even forensically.

The next section will discuss the Web application processing the compromised W32.Flamer client requests.

The Web application

***Location:* /var/www/htdocs/newsforyou/**

The Newsforyou application is written in PHP and contains the primary command-and-control functionality split into two parts: the main module and the control panel. The control panel is a basic user interface which allows packages to be uploaded and installed on chosen W32.Flamer clients. It also allows for the retrieval of stolen data that had been uploaded from these clients.

The table below describes the layout and offers a brief description of the components of the Newsforyou application:

Table 6

Directories	Description
/newsforyou	Main command and control application
/newsforyou/news	Encrypted packages distributed to all infected clients*
/newsforyou/ads	Encrypted packages distributed to chosen infected clients**
/newsforyou/entries	Encrypted data uploaded from infected clients**
/newsforyou/files	Temporary location for files when creating unloads
/newsforyou/tmp	Temporary files (database exports, tar archives)
/newsforyou/bak	Archives generated when unloading
/newsforyou/CP	Control panel

*Symmetric encryption with known key
**Asymmetric encryption with known public key, private key unknown

The application is designed to resemble a simple news/blog application. This approach may serve to disguise the true nature of the application from any automation or casual inspection. Although the code was running on a Linux server, it is likely some of the command-and-control servers were running Windows, or at least that the code was developed and tested on Windows computers.

The following comment was present in one of the PHP files:

```
-- This function was added by D***, returns a true/false value
depending on if this is a Windows box
```

Authors

The PHP source code references four authors, identified in the table below, and also attributes to them particular functionality within the code:

Table 7

@author	Edited Files	Dates	Control Panel	Protocols	Database	Cleanup	Encryption
D***	33	12/4/2006 01/23/2007	X	X	X	X	X
H*****	10	09/02/2007	X	X	X		X
O******	4	12/3/2006			X		
R***	1	2011				X	

It is clear that D*** and H***** had the most input into the project, having edited the most files in the application. O****** and R*** were tasked with database and cleanup operations and could easily have had little or no understanding of the inner workings of the application.

However, D*** and H***** have direct involvement in handling interactions with clients, as they worked on the protocols and also worked on the control panel that the operators used. It is likely D*** and O****** knew each other, as they both worked on the same files and during a similar time period in December 2006.

```
/**
* @desc This class is in charge of handling the SQL for the site. <BR>
*       It basically connects between the database itself and the storage class.
*
* @author O      , D
*/
class LowLevelStorage extends Breakable
{
    /**
    * @var $db_object This varible will hold the DBWrapper class object.
    */
    var $db_object;


    /**
    * @desc The LowLevelStorage constructor.<BR>
    *       Creates the DBWrapper object in $db_object.
    */
    function LowLevelStorage()
    {
        // Initializes successful_init to FALSE.
        $this->initialized(FALSE);

        // Creates the DBWrapper object.
        $this->db_object = new DBWrapper(LL_STORE_DB_SERVER, LL_STORE_DB_USERNAME,
        // Checks if the created object is broken.
        if ($this->db_object->is_broken()) {
            return;
        }
        // Sets successful_init to TRUE.
        $this->initialized(TRUE);
    }
```

H***** is responsible for the SignupProtocol, while D*** is involved in the OldProtocol (Flamer), both of which will be discussed in more detail in the next section.

Protocols

The main module communicating with the compromised clients is index.php. It deciphers the protocol then logs, decodes, and processes requests. Four protocols have been identified, of which three are in use. The Red protocol has not been implemented yet.

The existence of three supported protocols, along with one protocol under development, confirms the C&C server's requirement to communicate with multiple evolutions (variants) of W32.Flamer or additional cyber-espionage malware families currently unknown to the public.

```
switch ($protocol_num) {
    case PROTOCOL_OLD:
        return array(RC_SUCCESS, new OldProtocol);

    case PROTOCOL_OLD_E:
        return array(RC_SUCCESS, new OldProtocolE);

    case PROTOCOL_SIGNUP:
        return array(RC_SUCCESS, new SignupProtocol);
    // Currently, RED isn't supported yet
    //case PROTOCOL_RED:
    //  return array(RC_SUCCESS, new RedProtocol);

    default:
        Log::log_report(LOG_SEVERITY_ERROR, "Bad protocol number
        return array(RC_PROTOCOL_UNKNOWN_PROTOCOL);
}
```

These protocols are identified in the table below:

Table 8

Protocol Identifier	Protocol	Request
PROTOCOL_OLD*	HTTPS	UNIQUE_NUMBER=[DIGITS]&PASSWORD=LifeStyle2
PROTOCOL_SIGNUP	HTTPS	uid=[DIGITS]&action=[DIGITS]
PROTOCOL_OLD_E**	HTTP	NOT(uid=[DIGITS]&action=[DIGITS])
PROTOCOL_RED	N/A	N/A

*Used by W32.Flamer
**PROTOCOL_OLD with custom encryption over HTTP

Specific requests handled by the application are:

Table 9

Request	Functionality
GetNewsHandler	Responsible for sending news to compromised clients
AddEntryHandler	Responsible for storing entries (stolen data) from compromised clients
GetAdHandler	Responsible for sending ads to compromised clients

The possibility that multiple Trojans, or at least evolutions of W32.Flamer, are at work here is backed up by the fact that four different IDs are used internally to identify them:

Table 10

Client	Internal ID	Protocol	Threat
CLIENT_TYPE_SP	1	PROTOCOL_OLD	Unknown
CLIENT_TYPE_SPE	2	PROTOCOL_OLD_E	Unknown
CLIENT_TYPE_FL	3	PROTOCOL_OLD	W32.Flamer
CLIENT_TYPE_IP	6	PROTOCOL_SIGNUP	Unknown
N/A	N/A	PROTOCOL_RED	Unknown

Here is a snippet of code that identified connecting clients:

```
function determine_client_type($raw_field_str)
{
    // Check to see if it's an SP client
    if (preg_match('/TOOL_B=SP/', $raw_field_str)) {
        // It's an SP client
        return array(RC_SUCCESS, CLIENT_TYPE_SP);
    }

    // Check to see if it's an FL client, those either don't send out a filename or use just digits
    if (preg_match('/FILE_NAME=\d*&/', $raw_field_str)) {
        // It's an FL client
        return array(RC_SUCCESS, CLIENT_TYPE_FL);
    }

    // This could still be an SP client, check for a filename that doesn't contain digits
    if (preg_match('/FILE_NAME=[^&\d]+/', $raw_field_str)) {
        return array(RC_SUCCESS, CLIENT_TYPE_SP);
    }

    // Hmmmmm, must be some really old client
    return array(RC_SUCCESS, CLIENT_TYPE_UNKNOWN);
}
```

It is likely here that CLIENT_TYPE_SP, CLIENT_TYPE_SPE, and CLIENT_TYPE_FL implemented by D*** are evolutions of the same threat. However, CLIENT_TYPE_IP which is implemented by H***** appears to come after the in-use Flamer protocol, which suggests that either new variants exist that we are unaware of, or there is a separate Trojan at work. They also included an unrecognized fallback for clients.

Stolen Data

Compromised clients upload stolen data to the entries directory. Files stored in this directory are encrypted with a public key stored in the database. These files cannot be decrypted without the corresponding private key. There was a file of 157,548 bytes still left on the server that the operators did not have the opportunity to download.

The stolen data is encrypted with this public key on the server, thus requiring the corresponding unknown private key to decrypt:

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtZslxFiR9KJE05Nhh7Xk
+lVVpD9F6AQnvZeknDiwL3SBjZB/dB/LLXtwiet8LUS6JYCXnaIq4NxW1PymwGFZ
zuc/B3p+ZAFPt06veOHOfaMAI0KDMb+laNPINvn/jJ8TfvCaUMUuMEY4sayh0xwD
MwSAazMYI8rvaaS/BqhI/6vPN6D02UIpwT1TSBVeRRoPBHuYE7A93b8vJw9sBGIp
KXZ90sgP1CjdAmCbhYelelninKdeTKCGvd5YXt86grWgEVf5WXzxXi3ZK1T4w0Yt
mNhUEAwS7zCdtZ+Ak8b0M83wAirASvPZiBl6qF8hqCT5pKkwgBG//kk8JicboLsM
VQIDAQAB
-----END PUBLIC KEY-----
```

Based on timestamps seen on the first server in the /newsforyou/bak directory we determined that the operators downloaded the stolen data on four separate dates in May.

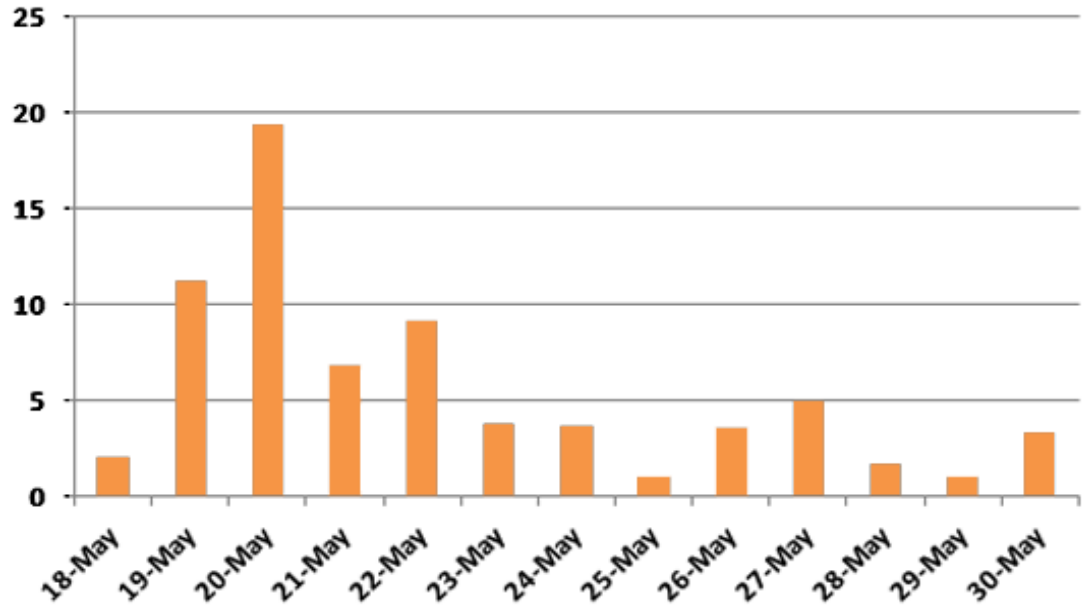
Table 11

Date	Stolen data files
2012-05-22	112
2012-05-23	12
2012-05-24	11
2012-05-30	43

A total of around 75 MB of stolen data was found in the backups on one of the servers (server #1). Although the timing data is limited here, the timestamps of these backups suggest that the operators could be in the EMEA region. There was a four-day delay from the initial setup to when the operators began to retrieve stolen data. This is a possible indicator that the information about the new server had to take time to filter down to them.

Figure 3.1

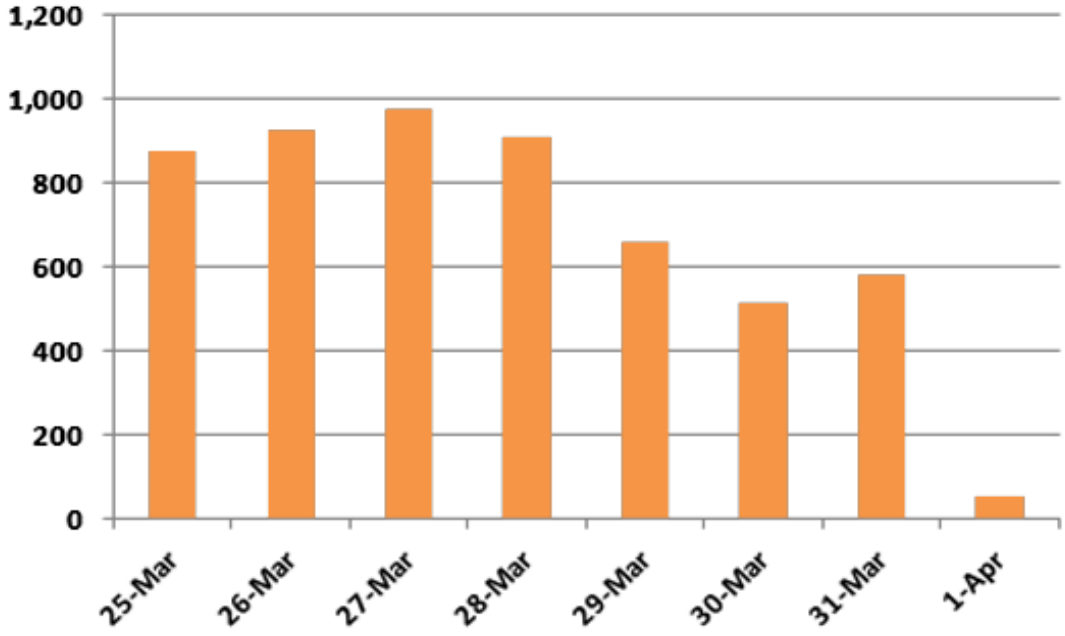
Megabytes of stolen data by date, server #1



In comparison, the other server (server #2) saw a massive 5.7 GB of stolen data:

Figure 3.2

Megabytes of stolen data by date, server #2



Activity

By examining the files in the bak directories, specifically the number of files contained within the backup archives created each day, it is possible to obtain an indication of the activity of the C&C server, which can be seen in the chart below:

Figure 4.1

Number of backup archives created each day, server #1

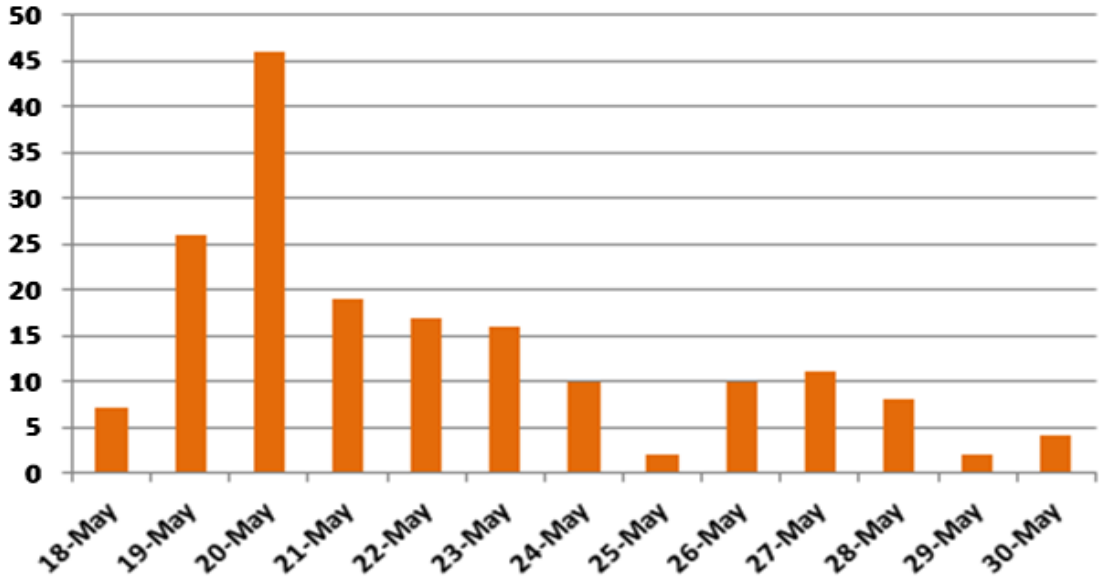
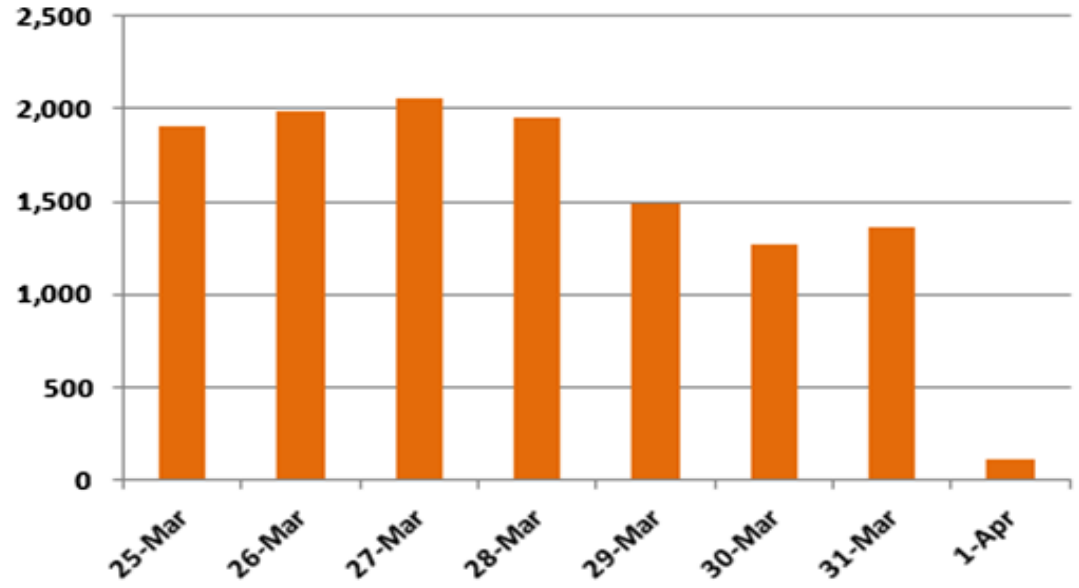


Figure 4.2

Number of backup archives created each day, server #2



On the first server, there is a clear increase in activity around May 20, which then drops off. This is to be expected as computers are cleared up.

On the second server the Flamer operation is in full flow and huge volumes of files are being created on the system.

Payload

The news directory contains a compressed and encrypted file for distribution to the compromised clients connecting to the server. This file is encrypted with a known symmetric key retrieved from the database.

Table 12

MD5	Filename	File size
0xfccd97e2513089e74ba707bbf4c1b0e5	59c5452fb2ba21091513ccdc1e0ec7fd	204,024 bytes

This is the browse32.ocx Flamer module, the sole purpose of which is to remove all known traces of the Flamer malware from the compromised client.

The payload is encrypted with this symmetric key:

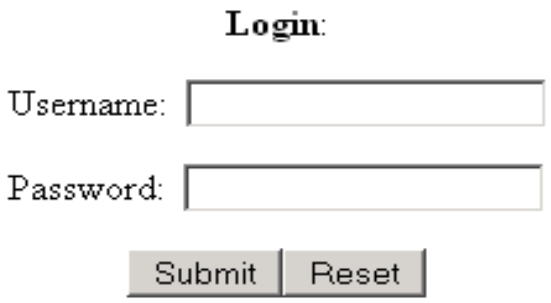
```
R&^T&IOHIO%^&^E%@#$UIO*()
```

The control panel

Access to the Flamer control panel requires authentication:

Figure 5

Control panel login page



The following credentials are used to gain access to the control panel (only the MD5 password hash is available; attempts to crack the password were unsuccessful):

Username: username
Password Hash: 27934e96d90d06818674b98bec7230fa

The control panel for interacting with Flamer-compromised computers is very basic. The simple interface allows the operators to upload packages and download encrypted data stolen from the compromised clients. The operator is required to upload specially crafted tar.gz files. The archived filename needs to follow a specific format, which is parsed and then stored in the ads or news directory, depending on the file extension used.

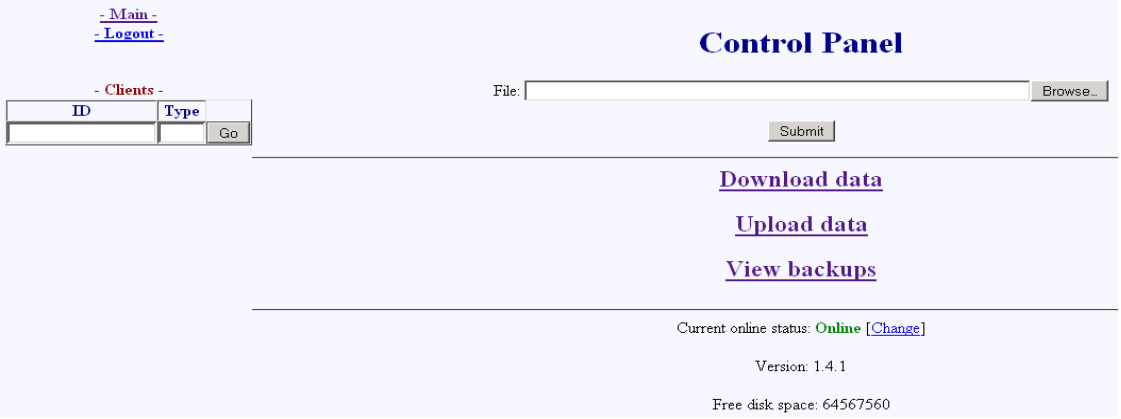
The filename format for these packages is as follows:

```
[USER _ TYPE] _ [USER _ ID] _ [PRIORITY] _ [TEMP _ FLAG][OPTIONAL _ AND _ IGNORED].news
[USER _ TYPE] _ [USER _ ID] _ [EXPIRY _ TIME] _ [AD _ NAME].ad
```

The following is a screen shot of the control panel that allows the operator to upload data to the server:

Figure 6

Control panel data upload screen



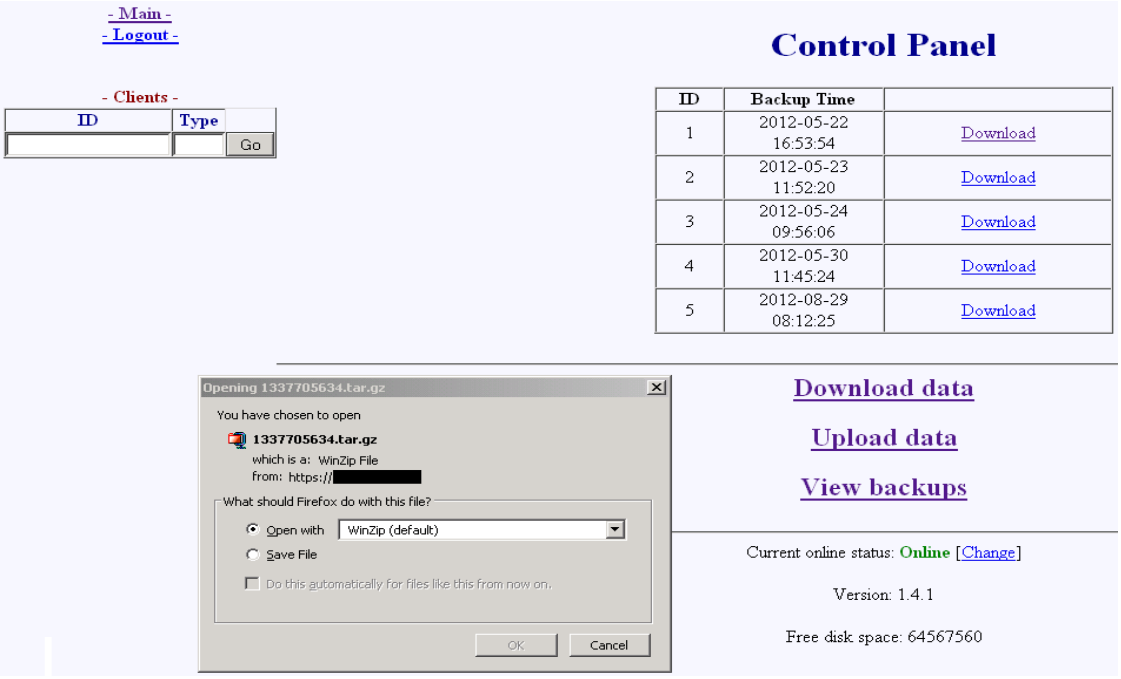
This approach to uploading packages and downloading data fits the profile of military and intelligence operations. A typical control panel is easy to use and self-explanatory. The operator has full control over what to do with the compromised clients and has the ability to retrieve and inspect the stolen information. The Flamer control panel provides limited capabilities to the operator, preventing them from viewing and interpreting the information being exchanged.

This approach would help prevent operators from knowing what is contained within the packages or the significance of the data being exchanged between clients and the attackers. Such a design would also serve as a defensive measure as it would prevent arbitrary packages from being uploaded and prevent downloaded data being from being inspected by unknown parties that may have gained access to the control panel.

The following screen on the control panel allows the user to view and download backed up data from the server:

Figure 7

Control panel screen for downloading stolen data



The tar.gz contains all the relevant stolen data and details of where the information was stolen from. The private key is required here to inspect the encrypted data.

The database

Location: /var/lib/mysql

The database is used to store the relevant data about connecting clients, packages to send to the clients, some logging and settings required for encryption, and authentication to access the control panel.

Table 13

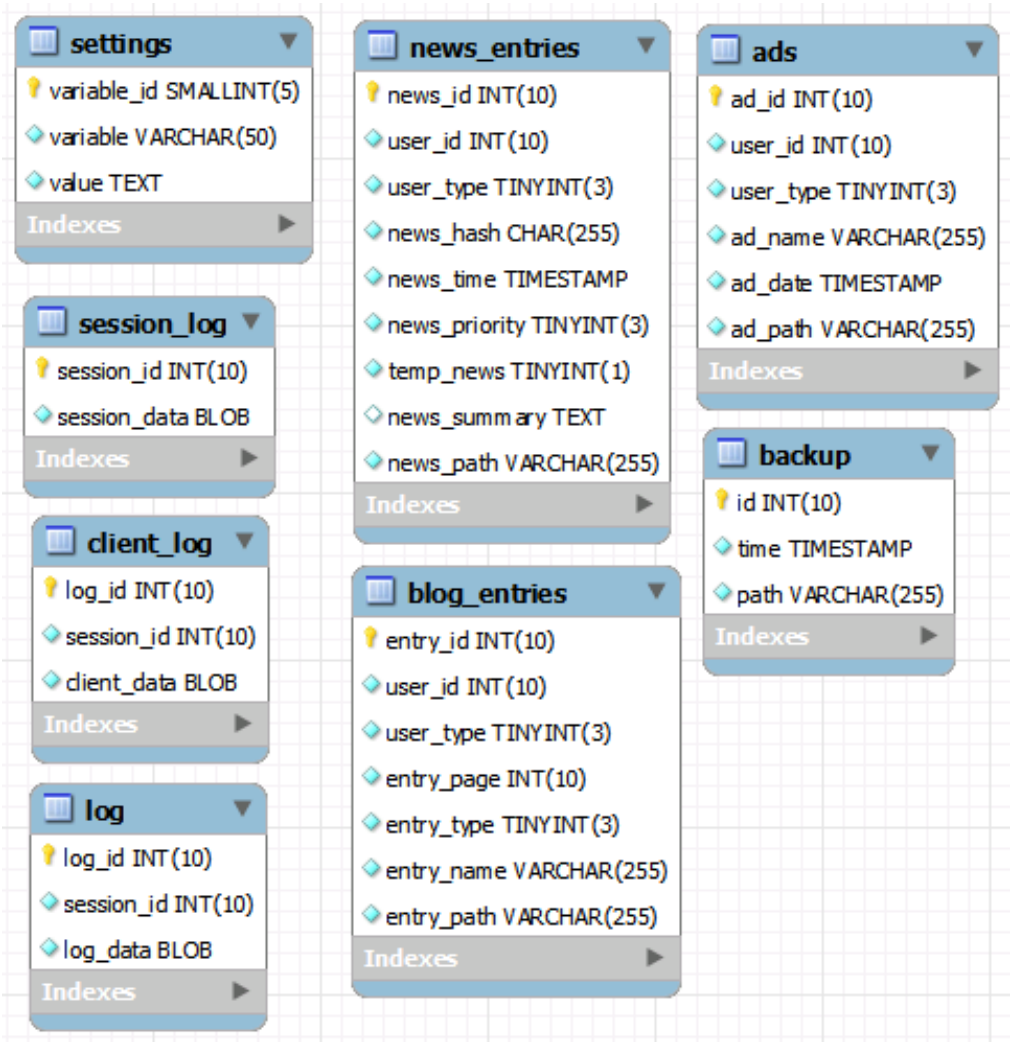
Database	User	Password
MySQL	news_user	news_pass

Schema

The database is used to store the relevant data about connecting clients, packages to send to the clients, some logging and settings required for encryption, and authentication to access the control panel.

Figure 8

Newsforyou application database schema



The **session_log, log**, and **client_log** are all encrypted using the keys stored in the **settings** table shown below and were successfully decrypted:

Table 14

Variable	Value
is_online	True
cp_user	username
cp_hash 8	27934e96d90d06818674b98bec7230fa
max_backup_size	52428800
version	1.4.11.4.1
minimum_unload_size	1572864
unload_flag	0
general_key	acK3xKMoJzsa9AVvtg59+OT4RM/x5MQ3bO2p0j+5Jd0=
session_key	13eZ+Now4Pt5ATpPv3WUv1E8UrvJReVMD0pO0MTQqol=

The database is regularly emptied by the delete.php script; there is limited data in it. The following table gives a brief description of the tables found in the database:

Table 15

Table	Records	Record Id	Encryption	Description
settings	11	11	N	Configuration values listed in the table below
session_log	8	1144	Y	Encrypted session log
client_log	4	1071	Y	Encrypted client log
log	213	21259	Y	Encrypted logs
news_entries	1	1	N	Entries in newsforyou/news (payloads)
blog_entries	1	189	N	Entries in newsforyou/entries (stolen data)
ads	0	n/a	N	Ad files in newsforyou/ads/
backup	4	4	N	Backup files in newsforyou/bak/

The **session_log** table contains details of all connections to the server while **client_log** only contains connections with a recognized protocol (e.g. valid compromised computers).

Examining the data in the table indicates that 1071 valid requests were recorded from compromised clients. A decrypted client log entry contains the following information:

```
CLIENT _ ID:[UUID]
CLIENT _ TYPE:3
CLIENT _ VERSION:0
REQUEST _ TYPE:0
RAW _ REQUEST:UNIQUE _ NUMBER=[UUID]&PASSWORD=LifeStyle2&ACTION=1&FILE _
NAME=&FILE _ SIZE=0
PROTOCOL:1
```

The four encrypted requests in the database on the first server are W32.Flamer client requests, which use the old protocol:

- CLIENT = 3 = CLIENT_TYPE_FL
- PROTOCOL = 1 = OLD_PROTOCOL

The additional sessions logged are likely attributed to researchers who had discovered the whereabouts of the command-and-control servers.

Table 16

IP Address	Organization	Country	client_log
77.42.[REMOVED]	LIBANTELECOM	Lebanon	Y
37.8. [REMOVED]	Israel Haifa Hadara Technologies Private Shareholding Company	Israel	Y
37.75. [REMOVED]	Orange Palestine Group Co. for Technological Inves	Palestine	Y
79.212. [REMOVED]	Deutsche Telekom AG	Germany	N
95.211. [REMOVED]	LeaseWeb B.V.	Netherlands	N

The **session_log** data contained three separate HTTP_HOST values, indicating multiple Flamer C&C server URLs were used to access this server.

Although only one encrypted file existed in the newsforyou/entries folder (the stolen data uploaded from the compromised clients), the database reveals that 189 records had been created. The other 188 files had already been removed from the server. The entry left on the server had been successfully stolen from the computer located in Israel.

A final point to note here is in relation to the entry in the **news_entries** table. Only one record ever existed: the malicious payload to clean the computers up, uploaded on May 18, 2012, at 13.43:45. This server only served up one package and it was uploaded as soon as the server had been initially configured. The last time a compromised client connected to the server was Friday, June 1, 2012, at 11:42:47, and the last log recorded in the database was Friday, June 1, 2012, at 11:46:01.

Conclusion

Examining the W32.Flamer servers has provided additional insight into the architecture of not only the threat, but also into the command structure of the entities behind it. The server code was written and updated by at least four separate individuals, indicating a continuing development effort to support W32.Flamer and, potentially, new or additional threats of a similar nature. The command-and-control Web application has been in active development for many years, possibly as early as 2006, which is well before Flamer's earliest seen compilation date in 2010.

The operators of the C&C servers may be a group of less senior individuals, working on a need-to-know basis, as the operator is not required (nor has the permission) to interpret the value or purpose of the incoming data. Only the attackers have the permission to access and interpret this data. This separation of operational and attacker visibility and roles indicates that this is the work of a highly organized and sophisticated group. The likelihood of a large and well-funded entity's involvement in Flamer is corroborated by the use of the unique certificate weakness used to hijack the Windows Update feature to spread across networks.

They were also careful to unload and archive data where necessary, remove duplicate files, and delete unnecessary files to prevent the server from running out of disk space. This was an ongoing development up to 2011. R***'s edits in the source code suggest the concern was less about updating protocols and more about ensuring there was ample room on the servers for the stolen data to be uploaded to, begging the question of how much data was actually being stolen.

This investigation simply provides a snapshot in time of the Flamer attack campaign. Considering that logging was disabled and data was wiped clean in such a thorough manner, the remaining clues make it virtually impossible to determine the entity behind the campaign. There is little doubt that the larger project involving cyber-espionage tools, such as Flamer, will continue to evolve and retrieve information from the designated targets.

Resources

Flamer: Highly Sophisticated and Discreet Threat Targets the Middle East
http://symantec.com/connect/blogs/flamer-highly-sophisticated-and-discreet-threat-targets-middle-east

Painting a Picture of W32.Flamer
http://symantec.com/connect/blogs/painting-picture-w32flamer

Flamer: A Recipe for Bluetoothache
http://symantec.com/connect/blogs/flamer-recipe-bluetoothache

W32.Flamer: Spreading Mechanism Tricks and Exploits
http://symantec.com/connect/blogs/w32flamer-spreading-mechanism-tricks-and-exploits

W32.Flamer: Leveraging Microsoft Digital Certificates
http://symantec.com/connect/blogs/w32flamer-leveraging-microsoft-digital-certificates

W32.Flamer: Microsoft Windows Update Man-in-the-Middle
http://symantec.com/connect/blogs/w32flamer-microsoft-windows-update-man-middle

W32.Flamer: Enormous Data Collection
http://symantec.com/connect/blogs/w32flamer-enormous-data-collection

Flamer: Urgent Suicide
http://symantec.com/connect/blogs/flamer-urgent-suicide

Symantec
Security Response

About Symantec

Symantec is a global leader in providing security, storage and systems management solutions to help businesses and consumers secure and manage their information. Headquartered in Mountain View, Calif., Symantec has operations in more than 40 countries. More information is available at www.symantec.com.

For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 (800) 745 6054.

Symantec Corporation
World Headquarters
350 Ellis Street
Mountain View, CA 94043 USA
+1 (650) 527-8000
www.symantec.com