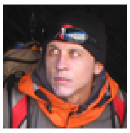
SECURELIST

The Flame: Questions and Answers

1.8



Aleks
Kaspersky Lab Expert
Posted May 28, 13:00 GMT
Tags: Targeted Attacks, Wiper, Cyber weapon, Cyber espionage, Flame

Duqu and Stuxnet raised the stakes in the cyber battles being fought in the Middle East – but now we've found what might be the most sophisticated cyber weapon yet unleashed. The 'Flame' cyber espionage worm came to the attention of our experts at Kaspersky Lab after the UN's International Telecommunication Union came to us for help in finding an unknown piece of malware which was deleting sensitive information across the Middle East. While searching for that code – nicknamed Wiper – we discovered a new malware codenamed Worm.Win32.Flame.

Flame shares many characteristics with notorious cyber weapons Duqu and Stuxnet: while its features are different, the geography and careful targeting of attacks coupled with the usage of specific software vulnerabilities seems to put it alongside those familiar 'super-weapons' currently deployed in the Middle East by unknown perpetrators. Flame can easily be described as one of the most complex threats ever discovered. It's big and incredibly sophisticated. It pretty much redefines the notion of cyberwar and cyberespionage.

For the full low-down on this advanced threat, read on…

General Questions

What exactly is Flame? A worm? A backdoor? What does it do?

Flame is a sophisticated attack toolkit, which is a lot more complex than Duqu. It is a backdoor, a Trojan, and it has worm-like features, allowing it to replicate in a local network and on removable media if it is commanded so by its master.

The initial point of entry of Flame is unknown - we suspect it is deployed through targeted attacks; however, we haven't seen the original vector of how it spreads. We have some suspicions about possible use of the MS10-033 vulnerability, but we cannot confirm this now.

Once a system is infected, Flame begins a complex set of operations, including sniffing the network traffic, taking screenshots, recording audio conversations, intercepting the keyboard, and so on. All this data is available to the operators through the link to Flame's command-and-control servers.

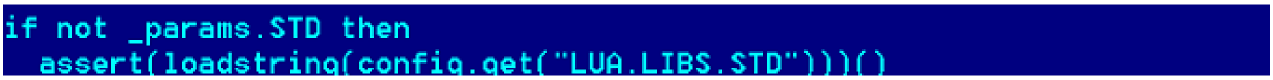
Later, the operators can choose to upload further modules, which expand Flame's functionality. There are about 20 modules in total and the purpose of most of them is still being investigated.

How sophisticated is Flame?

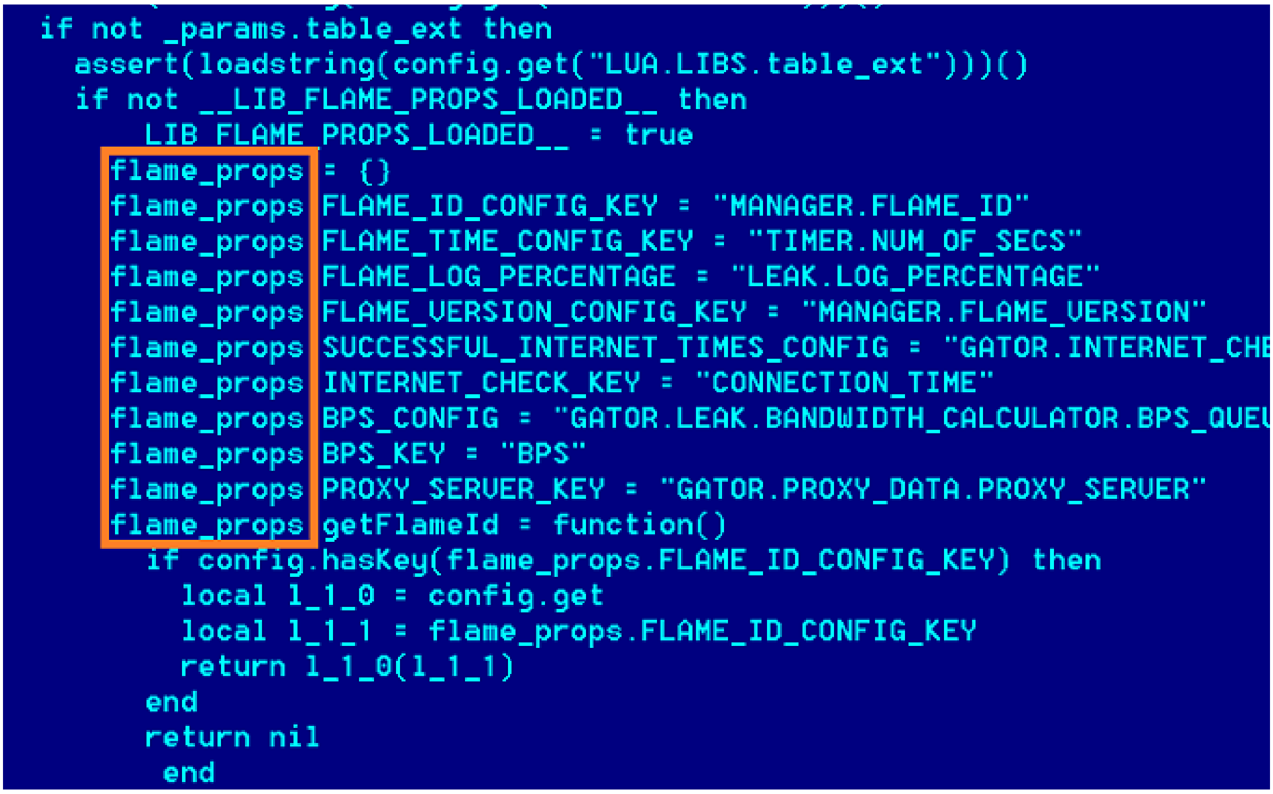
First of all, Flame is a huge package of modules comprising almost 20 MB in size when fully deployed. Because of this, it is an extremely difficult piece of malware to analyze. The reason why Flame is so big is because it includes many different libraries, such as for compression (zlib, libbz2, ppmd) and database manipulation (sqlite3), together with a Lua virtual machine.

Lua is a scripting (programming) language, which can very easily be extended and interfaced with C code. Many parts of Flame have high order logic written in Lua - with effective attack subroutines and libraries compiled from C++.

The effective Lua code part is rather small compared to the overall code. Our estimation of development 'cost' in Lua is over 3000 lines of code, which for an average developer should take about a month to create and debug.



```
if not _params.STD then
  assert(loadstring(config.get("LUA.LIBS.STD")))()
```



```
if not _params.table_ext then
  assert(loadstring(config.get("LUA.LIBS.table_ext")))()
  if not __LIB_FLAME_PROPS_LOADED__ then
      LIB_FLAME_PROPS_LOADED__ = true
    flame_props = {}
    flame_props FLAME_ID_CONFIG_KEY = "MANAGER.FLAME_ID"
    flame_props FLAME_TIME_CONFIG_KEY = "TIMER.NUM_OF_SECS"
    flame_props FLAME_LOG_PERCENTAGE = "LEAK.LOG_PERCENTAGE"
    flame_props FLAME_VERSION_CONFIG_KEY = "MANAGER.FLAME_VERSION"
    flame_props SUCCESSFUL_INTERNET_TIMES_CONFIG = "GATOR.INTERNET_CHE
    flame_props INTERNET_CHECK_KEY = "CONNECTION_TIME"
    flame_props BPS_CONFIG = "GATOR.LEAK.BANDWIDTH_CALCULATOR.BPS_QUEU
    flame_props BPS_KEY = "BPS"
    flame_props PROXY_SERVER_KEY = "GATOR.PROXY_DATA.PROXY_SERVER"
    flame_props getFlameId = function()
      if config.hasKey(flame_props.FLAME_ID_CONFIG_KEY) then
        local l_1_0 = config.get
        local l_1_1 = flame_props.FLAME_ID_CONFIG_KEY
        return l_1_0(l_1_1)
      end
      return nil
       end
```

Also, there are internally used local databases with nested SQL queries, multiple methods of encryption, various compression algorithms, usage of Windows Management Instrumentation scripting, batch scripting and more.

Running and debugging the malware is also not trivial as it’s not a conventional executable application, but several DLL files that are loaded on system boot.

Overall, we can say Flame is one of the most complex threats ever discovered.

How is this different to or more sophisticated than any other backdoor Trojan? Does it do specific things that are new?

First of all, usage of Lua in malware is uncommon. The same goes for the rather large size of this attack toolkit. Generally, modern malware is small and written in really compact programming languages, which make it easy to hide. The practice of concealment through large amounts of code is one of the specific new features in Flame.

The recording of audio data from the internal microphone is also rather new. Of course, other malware exists which can record audio, but key here is Flame’s completeness - the ability to steal data in so many different ways.

Another curious feature of Flame is its use of Bluetooth devices. When Bluetooth is available and the corresponding option is turned on in the configuration block, it collects information about discoverable devices near the infected machine. Depending on the configuration, it can also turn the infected machine into a beacon, and make it discoverable via Bluetooth and provide general information about the malware status encoded in the device information.

What are the notable info-stealing features of Flame?

Although we are still analyzing the different modules, Flame appears to be able to record audio via the microphone, if one is present. It stores recorded audio in compressed format, which it does through the use of a public-source library.

Recorded data is sent to the C&C through a covert SSL channel, on a regular schedule. We are still analyzing this; more information will be available on our website soon.

The malware has the ability to regularly take screenshots; what’s more, it takes screenshots when certain “interesting” applications are run, for instance, IM’s. Screenshots are stored in compressed format and are regularly sent to the C&C server - just like the audio recordings.

We are still analyzing this component and will post more information when it becomes available.

When was Flame created?

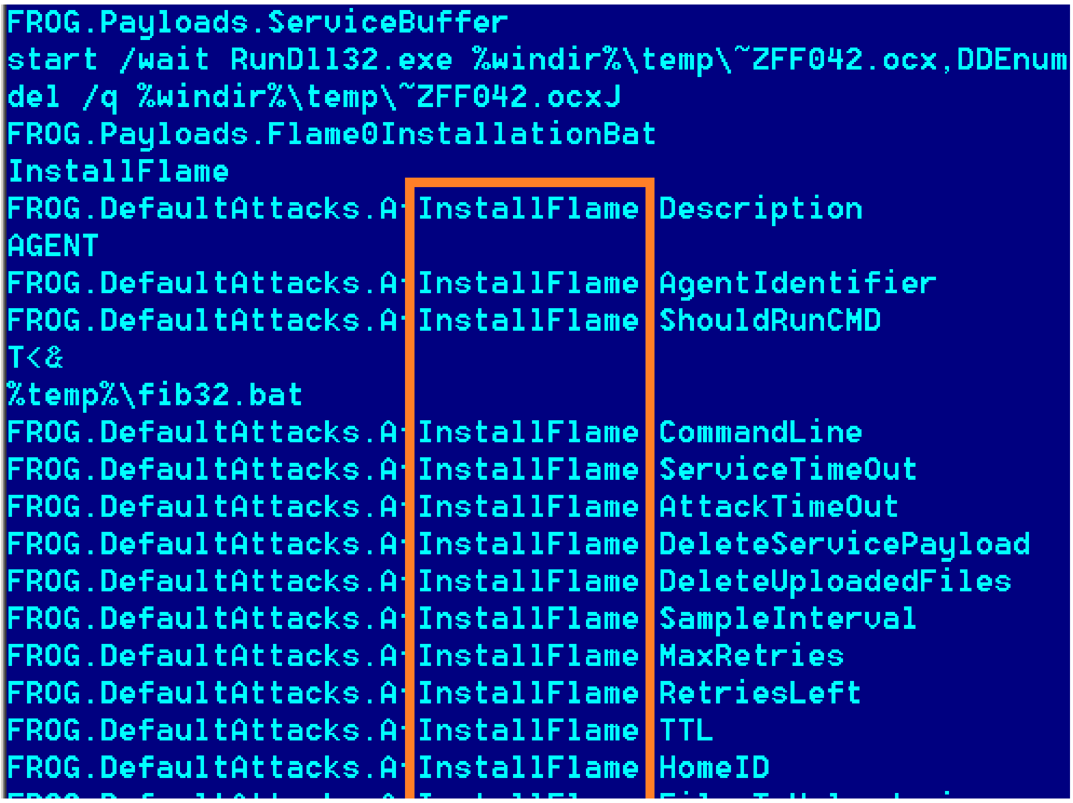
The creators of Flame specially changed the dates of creation of the files in order that any investigators couldn’t establish the truth re time of creation. The files are dated 1992, 1994, 1995 and so on, but it’s clear that these are false dates.

We consider that in the main the Flame project was created no earlier than in 2010, but is still undergoing active development to date. Its creators are constantly introducing changes into different modules, while continuing to use the same architecture and file names. A number of modules were either created of changed in 2011 and 2012.

According to our own data, we see use of Flame in August 2010. What’s more, based on collateral data, we can be sure that Flame was out in the wild as early as in February to March 2010. It’s possible that before then there existed earlier version, but we don’t have data to confirm this; however, the likelihood is extremely high.

Why is it called Flame? What is the origin of its name?

The Flame malware is a large attack toolkit made up of multiple modules. One of the main modules was named Flame - it’s the module responsible for attacking and infecting additional machines.



```
FROG.Payloads.ServiceBuffer
start /wait RunDll32.exe %windir%\temp\~ZFF042.ocx,DDEnum
del /q %windir%\temp\~ZFF042.ocxJ
FROG.Payloads.Flame0InstallationBat
InstallFlame
FROG.DefaultAttacks.A InstallFlame Description
AGENT
FROG.DefaultAttacks.A InstallFlame AgentIdentifier
FROG.DefaultAttacks.A InstallFlame ShouldRunCMD
T<&
%temp%\fib32.bat
FROG.DefaultAttacks.A InstallFlame CommandLine
FROG.DefaultAttacks.A InstallFlame ServiceTimeOut
FROG.DefaultAttacks.A InstallFlame AttackTimeOut
FROG.DefaultAttacks.A InstallFlame DeleteServicePayload
FROG.DefaultAttacks.A InstallFlame DeleteUploadedFiles
FROG.DefaultAttacks.A InstallFlame SampleInterval
FROG.DefaultAttacks.A InstallFlame MaxRetries
FROG.DefaultAttacks.A InstallFlame RetriesLeft
FROG.DefaultAttacks.A InstallFlame TTL
FROG.DefaultAttacks.A InstallFlame HomeID
```

```
FROG.DefaultAttacks.A InstallFlame FilesToUpload.size
```

Is this a nation-state sponsored attack or is it being carried out by another group such as cyber criminals or hacktivisits?

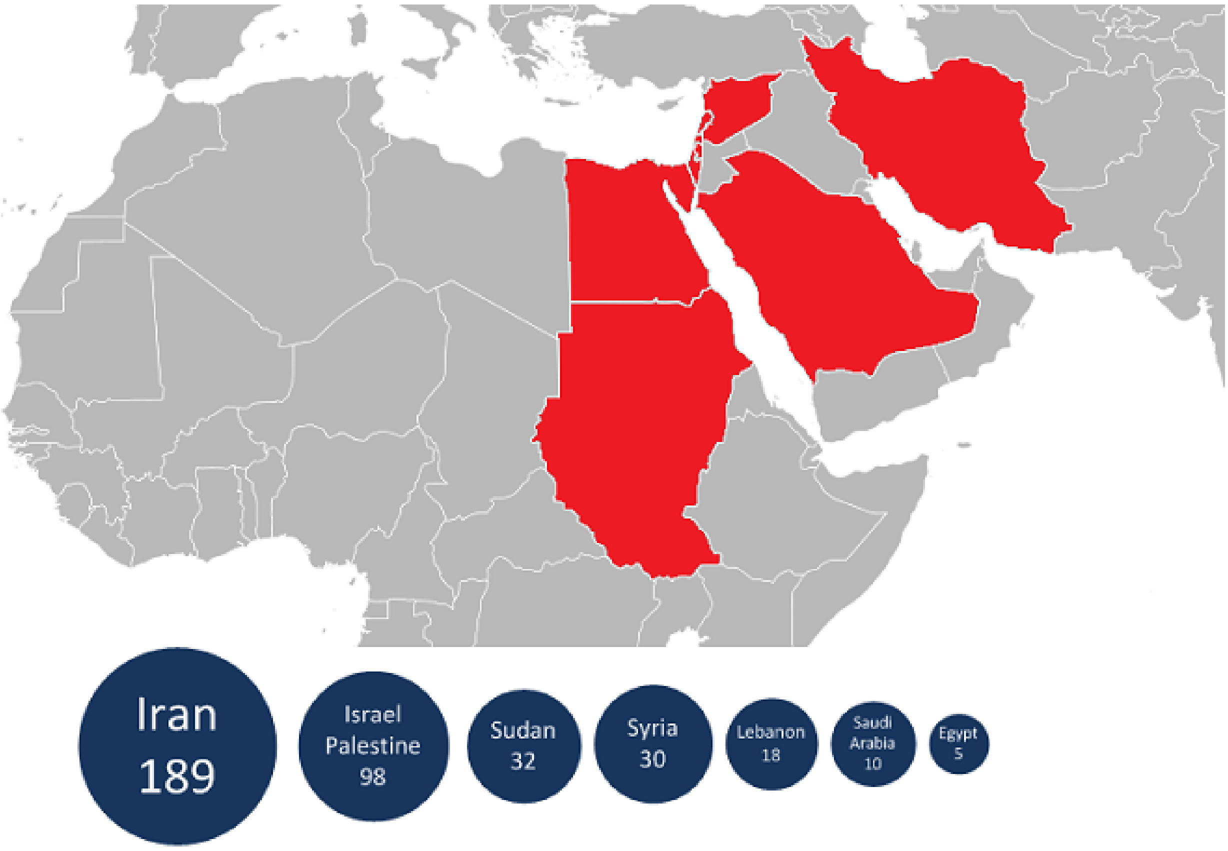
Currently there are three known classes of players who develop malware and spyware: hacktivists, cybercriminals and nation states. Flame is not designed to steal money from bank accounts. It is also different from rather simple hack tools and malware used by the hacktivists. So by excluding cybercriminals and hacktivists, we come to conclusion that it most likely belongs to the third group. In addition, the geography of the targets (certain states are in the Middle East) and also the complexity of the threat leaves no doubt about it being a nation state that sponsored the research that went into it.

Who is responsible?

There is no information in the code or otherwise that can tie Flame to any specific nation state. So, just like with Stuxnet and Duqu, its authors remain unknown.

Why are they doing it?

To systematically collect information on the operations of certain nation states in the Middle East, including Iran, Lebanon, Syria, Israel and so on. Here's a map of the top 7 affected countries:



Is Flame targeted at specific organizations, with the goal of collecting specific information that could be used for future attacks? What type of data and information are the attackers looking for?

From the initial analysis, it looks like the creators of Flame are simply looking for any kind of intelligence - e-mails, documents, messages, discussions inside sensitive locations, pretty much everything. We have not seen any specific signs indicating a particular target such as the energy industry – making us believe it's a seen any specific signs indicating a particular target such as the energy industry - making us believe it's a complete attack toolkit designed for general cyber-espionage purposes.

Of course, like we have seen in the past, such highly flexible malware can be used to deploy specific attack modules, which can target SCADA devices, ICS, critical infrastructure and so on.

What industries or organizations is Flame targeting? Are they industrial control facilities/PLC/SCADA? Who are the targets and how many?

There doesn't seem to be any visible pattern re the kind of organizations targeted by Flame. Victims range from individuals to certain state-related organizations or educational institutions. Of course, collecting information on the victims is difficult because of strict personal data collecting policies designed to protect the identity of our users.

Based on your analysis, is this just one variation of Flame and there are others?

Based on the intelligence received from the Kaspersky Security Network, we are seeing multiple versions of the malware being in the wild - with different sizes and content. Of course, assuming the malware has been in development for a couple of years, it is expected that many different versions will be seen in the wild.

Additionally, Flame consists of many different plug-ins – up to 20 – which have different specific roles. A specific infection with Flame might have a set of seven plugins, while another infection might have 15. It all depends on the kind of information that is sought from the victim, and how long the system was infected with Flame.

Is the main C&C server still active? Is there more than one primary C&C server? What happens when an infected machine contacts the C&C server?

Several C&C servers exist, scattered around the world. We have counted about a dozen different C&C domains, run on several different servers. There could also be other related domains, which could possibly bring the total to around 80 different domains being used by the malware to contact the C&C. Because of this, it is really difficult to track usage of deployment of C&C servers.

Was this made by the Duqu/Stuxnet group? Does it share similar source code or have other things in common?

In size, Flame is about 20 times larger than Stuxnet, comprising many different attack and cyber-espionage features. Flame has no major similarities with Stuxnet/Duqu.

For instance, when Duqu was discovered, it was evident to any competent researcher that it was created by the same people who created Stuxnet on the "Tilded" platform.

Flame appears to be a project that ran in parallel with Stuxnet/Duqu, not using the Tilded platform. There are however some links which could indicate that the creators of Flame had access to technology used in the Stuxnet project - such as use of the "autorun.inf" infection method, together with exploitation of the same print spooler vulnerability used by Stuxnet, indicating that perhaps the authors of Flame had access to the same exploits as Stuxnet's authors.

On the other hand, we can't exclude that the current variants of Flame were developed after the discovery of Stuxnet. It's possible that the authors of Flame used public information about the distribution methods of Stuxnet and put it to work in Flame.

In summary, Flame and Stuxnet/Duqu were probably developed by two separate groups. We would position Flame as a project running parallel to Stuxnet and Duqu.

You say this was active since March 2010. That is close to the time when Stuxnet was discovered. Was this being used in tandem with Stuxnet? It is interesting they both exploit the printer-spooler vulnerability.

One of the best pieces of advice in any kind of operation is not to put all your eggs in one basket. Knowing that sooner or later Stuxnet and Duqu would be discovered, it would make sense to produce other similar projects - but based on a completely different philosophy. This way, if one of the research projects is discovered, the other one can continue unhindered.

Hence, we believe Flame to be a parallel project, created as a fallback in case some other project is discovered.

In your analysis of Duqu you mentioned “cousins” of Duqu, or other forms of malware that could exist. Is this one of them?

Definitely not. The “cousins” of Duqu were based on the Tilded platform, also used for Stuxnet. Flame does not use the Tilded platform.

This sounds like an info-stealing tool, similar to Duqu. Do you see this as part of an intelligence-gathering operation to make a bigger cyber-sabotage weapon, similar to Stuxnet?

The intelligence gathering operation behind Duqu was rather small-scale and focused. We believe there were less than 50 targets worldwide for Duqu - all of them, super-high profile.

Flame appears to be much, much more widespread than Duqu, with probably thousands of victims worldwide.

The targets are also of a much wider scope, including academia, private companies, specific individuals and so on.

According to our observations, the operators of Flame artificially support the quantity of infected systems on a certain constant level. This can be compared with a sequential processing of fields – they infect several dozen, then conduct analysis of the data of the victim, uninstall Flame from the systems that aren’t interesting, leaving the most important ones in place. After which they start a new series of infections.

What is Wiper and does it have any relation to Flame? How is it destructive and was it located in the same countries?

The Wiper malware, which was reported on by several media outlets, remains unknown. While Flame was discovered during the investigation of a number of Wiper attacks, there is no information currently that ties Flame to the Wiper attacks. Of course, given the complexity of Flame, a data wiping plugin could easily be deployed at any time; however, we haven’t seen any evidence of this so far.

Additionally, systems which have been affected by the Wiper malware are completely unrecoverable - the extent of damage is so high that absolutely nothing remains that can be used to trace the attack.

There is information about Wiper incidents only in Iran. Flame was found by us in different countries of the region, not only Iran.

Functionality/Feature Questions about the Flame Malware

What are the ways it infects computers? USB Sticks? Was it exploiting vulnerabilities other than the print-spooler to bypass detection? Any 0-Days?

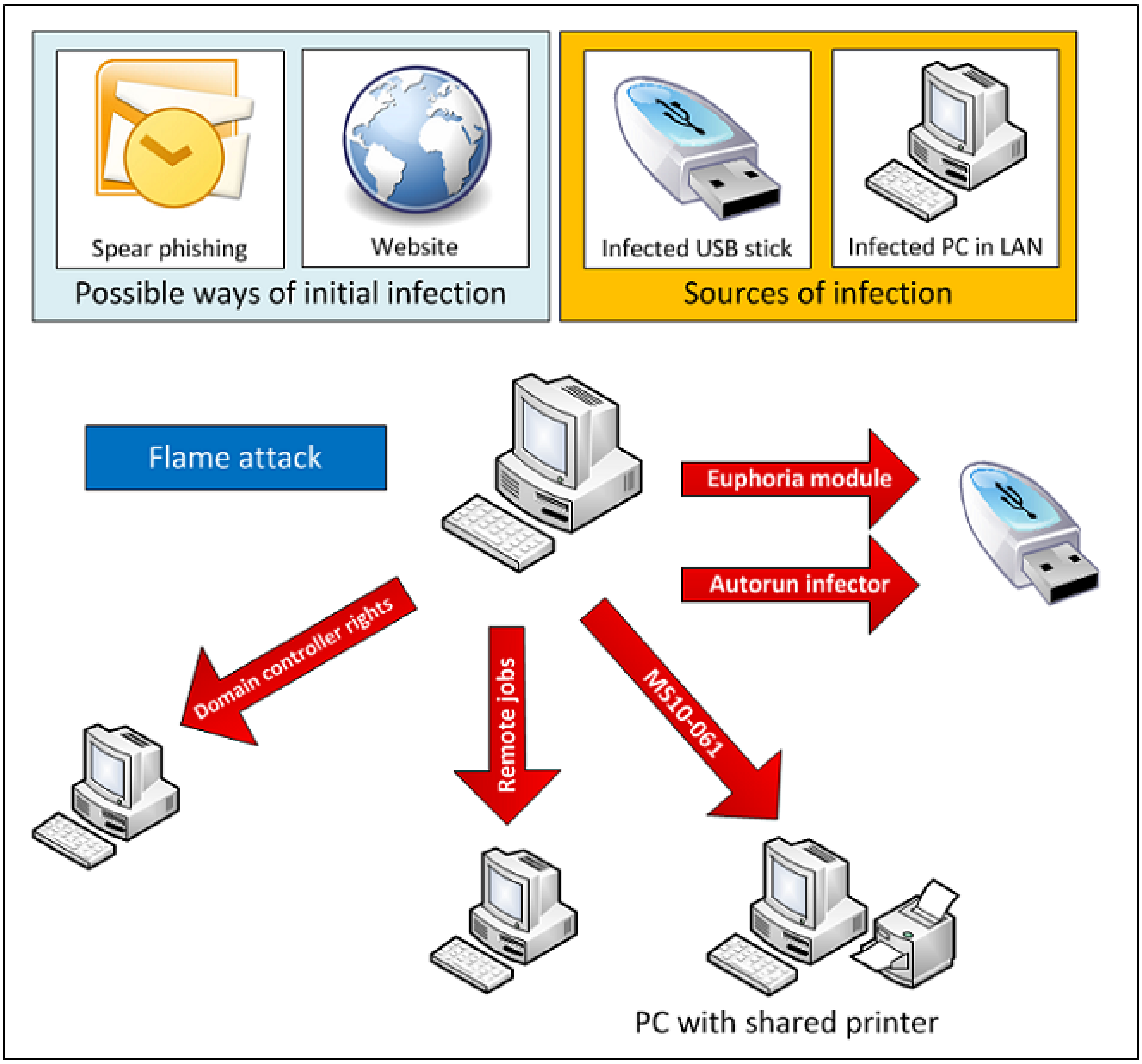
Flame appears to have two modules designed for infecting USB sticks, called “Autorun Infector” and “Euphoria”. We haven’t seen them in action yet, maybe due to the fact that Flame appears to be disabled in the configuration data. Nevertheless, the ability to infect USB sticks exists in the code, and it’s using two methods:

1. Autorun Infector: the “Autorun.inf” method from early Stuxnet, using the “shell32.dll” “trick”. What’s key here is that the specific method was used only in Stuxnet and was not found in any other malware since.
2. Euphoria: spread on media using a “junction point” directory that contains malware modules and an LNK file that trigger the infection when this directory is opened. Our samples contained the names of the files but did not contain the LNK itself.

In addition to these, Flame has the ability to replicate through local networks. It does so using the following:

1. The printer vulnerability MS10-061 exploited by Stuxnet - using a special MOF file, executed on the attacked system using WMI.
2. Remote jobs tasks.

3. When Flame is executed by a user who has administrative rights to the domain controller, it is also able to attack other machines in the network: it creates backdoor user accounts with a pre-defined password that is then used to copy itself to these machines.



At the moment, we haven’t seen use of any 0-days; however, the worm is known to have infected fully-patched Windows 7 systems through the network, which might indicate the presence of a high risk 0-day.

Can it self-replicate like Stuxnet, or is it done in a more controlled form of spreading, similar to Duqu?

The replication part appears to be operator commanded, like Duqu, and also controlled with the bot configuration file. Most infection routines have counters of executed attacks and are limited to a specific number of allowed attacks.

Why is the program several MBs of code? What functionality does it have that could make it so much larger than Stuxnet? How come it wasn’t detected if it was that big?

The large size of the malware is precisely why it wasn’t discovered for so long. In general, today’s malware is small and focused. It’s easier to hide a small file than a larger module. Additionally, over unreliable networks, downloading 100K has a much higher chance of being successful than downloading 6MB.

Flame’s modules together account for over 20MB. Much of these are libraries designed to handle SSL traffic, SSH connections, sniffing, attack, interception of communications and so on. Consider this: it took us several months to analyze the 500K code of Stuxnet. It will probably take year to fully understand the 20MB of code of Flame.

Flame.

Does Flame have a built-in Time-of-Death like Duqu or Stuxnet ?

There are many different timers built-in into Flame. They monitor the success of connections to the C&C, the frequency of certain data stealing operations, the number of successful attacks and so on. Although there is no suicide timer in the malware, the controllers have the ability to send a specific malware removal module (named “browse32”), which completely uninstalls the malware from a system, removing every single trace of its presence.

What about JPEGs or screen-shots? Is it stealing those too?

The malware has the ability to regularly take screenshots. What’s more, it takes screenshots when certain “interesting” applications are run, for instance, IM’s. Screenshots are stored in compressed format and are regularly sent to the C&C server, just like the audio recordings.

We are still analyzing this component and will post more information when it becomes available.

We will share a full list of the files and traces for technical people in a series of blog posts on Securelist during the next weeks.

What should I do if I find an infection and am willing to contribute to your research by providing malware samples?

We would greatly appreciate it if you could contact us by e-mail at the previously created mailbox for Stuxnet/Duqu research: stopduqu@kaspersky.com.

Update 1 (28-May-2012):

According to our analysis, the Flame malware is the same as “SkyWiper”, described by the CrySyS Lab and by Iran Maher CERT group where it is called “Flamer”.