Detecting Conficker in your Network

Adi Kriegisch (kriegisch@vrvis.at)

February 11, 2009

Abstract

Conficker[1] is a computer worm spreading on Windows operating system by mainly using a buffer overflow[2] or the Windows Autorun feature. The worm itself does not contain malware functions but contains a routine to load such code after infection. The purpose of this article is to sketch a way to detect such a worm in a small to medium business network as early as possible so that the effects of the worm can be minimized.

1 Introduction

In the past years malware has become more flexible by forming botnets that are remotely controllable. The first of those well known to public was the Storm worm botnet. The controlling servers, mostly bots by themselves, were hidden behind a DNS name with a technique named 'fast-flux'.

Conficker[3] takes 'fast-flux' to the next level by using dynamically generated domain names to find out about its control servers and load instructions from them. F-Secure managed to crack the algorithm built into Conficker to generate its daily list of domain names to check for control servers. Conficker uses Google, Yahoo and others to determine the current system date which then is used to generate 250 domain names for the current day.

There are numerous sources in the web dealing with the removal of the worm. Besides all the issues related to removal and embanking further spread of Conficker, early detection of the worm in a local network is required. Intrusion Detection and Prevention Systems like SNORT[4] will help substantially. Proxy Servers like Squid can be used to block access to a list of domains like the one created by F-Secure by reverse engineering Conficker's algorithm. Both methods require a certain infrastructure to be in place and capable of handling an even higher load.

This article deals with another way of intercepting Conficker's communication: deliver bogus DNS information to the worm.

2 DNS Server

Several security researchers like CERT.at[5] have published lists of all domain names[6] Conficker will use to get in touch with its masters and to get malware functions or updates. Using one of these lists makes it quite simple to add DNS trap functionality to BIND. First, create a zone file that refers to the local machine[7]:

```
$TTL 31536000
@ IN SOA ns1.conficker.at. \
  hostmaster.conficker.at. (
  2009020401 1W 2W 52W 1W)

  IN NS your.name.server

  A       127.0.0.1
* A       127.0.0.1
```

Second, register those domains within BIND's configuration file like this[8]:

```
zone "tkggvtqvj.org" {
```

```
  type master;
  file "master/conficker.db";
};
zone "yqdqyntx.com" {
  type master;
  file "master/conficker.db";
};
```

After doing this, Conficker cannot contact its control servers and runs headless. It still is able to infect other machines on the local subnet though.

OpenDNS[9] for example is starting to offer the very same thing to its customers[10].

There is one disadvantage to this brute force method: The complete list of domains contains 91.250 names per year. This demands quite alot of extra memory for BIND (roughly 300MB) and it takes quite long to (re)start the name service. BIND itself does not suffer performance loss after loading. To minimize those effects it would be wise to only load the current list with domains for the next few days. There are 250 domains per day which can be easily handled with much less RAM available.

As a side note: Assuming the name server used to serve Conficker's domains is used for other domains as well one needs to take care of zone update notifications. Notifications should be disabled for all Conficker zones by either disabling notifications globally and enabling only for zones that need notifications or disable notifications selectively for all Conficker domains. Notifications generate some traffic and use additional resources. To enable and disable them use:

```
notify [yes|no];
```

either in global options, in views or in zones section of BIND configuration.

3 Honeypot

Conficker is now isolated in the network and cannot download and execute malware. But there still is the danger of a spreading infection within the network. The easiest way to detect infections in the network would be to monitor for lookups of domains listed. This turned out to be harder than it first sounded because one may either log all queries to the name server or none but logging is no per zone feature. Changing BIND's source to be able to accomplish this is no good idea either.

As F-Secure[3] found out, the worm will try to connect to

```
http://%IPofDomainname%/search?q=%d
```

where %IPofDomainname% is the IP of one of the generated domain names and %d will be expanded to the filename of the malware to download. The filename itself is random with the extension '.tmp'. So even if the DNS lookup of the worm remains unseen it may still be detected by its follow-up behavior – the download in this case.

Our name server needs to send a valid IP address as answer. At this IP address a trap service is listening for incoming http connections and triggering alarms. Such a trap is called a Honeypot[11] and can expand to a very complex system. In this case a simple tool like 'Tiny Honeypot'[12] is more than sufficient.[1]

After downloading and installing 'Tiny Honeypot'[12] configuration is rather easy and can be done by integration into (x)inetd or by using netcat. Tiny Honeypot is written in Perl language and has a modular design. Every supported protocol is in its own file like those for 'http', 'ftp', 'ssh', 'smtp' and so on. There are more general plugins like 'catchall' and 'nullresp'. Taking a closer look at 'nullresp' shows how simple it is to write own modules with custom functionality:

```
sub nullresp {
  while (<STDIN>) {
    open(LOG, ">>$sesslog");
    select LOG;
    $|=1;
    print LOG $_;
```

[1] Even a simple shell script listening via inetd on the specified IP would do the job.

```
    close LOG;
  }
}
```

The source code above is a complete Tiny Honeypot plugin that does nothing else than just listen to what the remote part of the connection says and log everything. To use this plugin the following command line may be used.

```
thpot nullresp
```

Tiny Honeypot will always listen on stdin and write responses to stdout. Therefor it is necessary to use a wrapper like (x)inetd to make network communication possible.

To get real-time notification on activity one has to either write a simple protocol extension to 'Tiny Honeypot' that sends mail on connect[2] or use a log file monitor like logfmon[13] or find a different way to get to know about activity in the trap. Keeping in mind that doing all of this to keep the impact of one infection as small as possible makes it obvious that checking the trap once a day will not suffice.

4 Conclusion

The suggested way to detect the Conficker worm brings certain benefits only to those already having certain parts of the infrastructure used in this article up and running. The critical part of the setup is the DNS server. It needs to be capable to load 100000 domains or one has to create mechanisms to only load current zones. On the other hand, software like BIND is built for high performance[3] so it is a good way to handle the load.

OpenDNS' aforementioned service offers notification to system administrators as well. The only drawback here is that most business networks use one IP address for internet traffic with masquerading for their internal subnet. So the advance in information is to know that one machine is infected. It is the lack of details that make this service not so well suited for businesses.

Seeing Conficker and Storm worm in a line, probably subscribable DNS services will arise like it is now with SNORT signatures. Having such a service running via DNS will probably reduce the pressure to deploy SNORT in a network. Real IDS/IPS systems are quite heavy to operate in times of increasing bandwidths. By using DNS with fake answers to malware and probably even some phishing sites' requests one may get good extra security.

Tiny Honeypot is slim and easy to use. No special effort is needed. Such a setup will almost certainly be able to deal with all the copycat worms to follow at least as long as botnet creators won't start using new means of communication.

5 Acknowledgements

My thanks go to Aaron Kaplan from CERT Austria for the interesting discussion on this topic and for providing a ready to use zone file for BIND. My thanks also go to Stephan Mantler for being online when I needed him and for coming up with the Honeypot idea to solve the log issue.

References

[1] Wikipedia. Conficker — Wikipedia, the free encyclopedia [online]. 2009 [cited 10-February-2009]. Available from World Wide Web: `http://en.wikipedia.org/w/index.php?title=Conficker&oldid=269779729`.

[2] A limitation of messages per time frame should be in place because Conficker queries and connects to 250 domain names per day which may cause a flood of messages.

[3] There are some rare messages about what loads root name servers can handle on the internet.

[2] Mitre Corporation. Cve-2008-4250 [online]. 2008 [cited 05-February-2009]. Available from World Wide Web: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4250.

[3] F-Secure. Malware information pages: Worm:w32/downadup.al [online]. 2009 [cited 05-February-2009]. Available from World Wide Web: http://www.f-secure.com/v-descs/worm_w32_downadup_al.shtml.

[4] Jack Pepper (Autoshun). DNS conficker rules for SNORT autoshun plugin [online]. 2009 [cited 06-February-2009]. Available from World Wide Web: http://www.autoshun.com/downloads/conficker.rules.

[5] CERT.at. national CERT Austria [online]. 2009 [cited 10-February-2009]. Available from World Wide Web: http://www.cert.at/.

[6] CERT.at. complete list of all conficker domains (plain text) [online]. 2009 [cited 10-February-2009]. Available from World Wide Web: http://www.cert.at/static/conficker/all_domains.txt.

[7] CERT.at. Bind zone file containing all conficker domains [online]. 2009 [cited 10-February-2009]. Available from World Wide Web: http://www.cert.at/static/conficker/conficker.db.

[8] CERT.at. named.conf example [online]. 2009 [cited 10-February-2009]. Available from World Wide Web: http://www.cert.at/static/conficker/named.conf.conficker.

[9] OpenDNS. Opendns — providing a safer and faster internet [online]. 2009 [cited 09-February-2009]. Available from World Wide Web: http://www.opendns.com.

[10] The Register. Opendns rolls out conficker tracking, blocking [online]. 2009 [cited 09-February-2009]. Available from World Wide Web: http://www.theregister.co.uk/2009/02/07/opendns_conficker_protection.

[11] Wikipedia. Honeypot (computing) — wikipedia, the free encyclopedia [online]. 2009 [cited 05-February-2009]. Available from World Wide Web: http://en.wikipedia.org/w/index.php?title=Honeypot_(computing)&oldid=266313197.

[12] George Bakos. Tiny Honeypot [online]. 2009 [cited 06-February-2009]. Available from World Wide Web: http://www.alpinista.org/thp/.

[13] Nicholas Marriott. logfmon - log file monitoring daemon [online]. 2004 [cited 05-February-2009]. Available from World Wide Web: http://logfmon.sourceforge.net/.