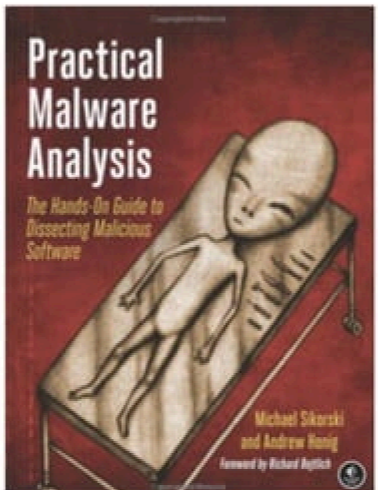
Practical Malware Analysis

Ch 14: Malware-Focused Network Signatures



Network Countermeasures

Common Network Countermeasures

- Filtering with firewalls and routers
  - By IP address, TCP and UDP ports
- DNS Servers
  - Resolve malicious domain names to an internal host (a *sinkhole*)
- Proxy servers
  - Can detect or prevent access to specific domains

Content-Based Countermeasures

- These devices can look at layer 7 data (deep packet inspection)
  - IDS (Intrusion Detection System)
  - IPS (Intrusion Prevention System)
  - Email proxy
  - Web proxy

Observing the Malware in Its Natural Habitat

- Before static or dynamic analysis
- Mine logs, alerts, and packet captures generated by malware in its original location

Advantages of Real Networks

- Live-captured data is the most accurate
  - Some malware detects lab environments
- Real traffic contains information about both ends, infected host and C&C server
- Passively monitoring traffic cannot be detected by the attacker
  - OPSEC (Operational Security)

Indications of Malicious Activity

Table 15-1. Sample Network Indicators of Malicious Activity

Information type	Indicator
Domain (with resolved IP address)	`www.badsite.com (123.123.123.10)`
IP address	`123.64.64.64`

GET request	`GET /index.htm HTTP 1.1`
	`Accept: */*`
	`User-Agent: Wefa7e`
	`Cache-Control: no`

OPSEC

- Preventing adversaries from obtaining sensitive information
- Running malware at home may alert attackers
  - Who expected it to be run in a company

Ways an Attacker Can Identify Investigative Activity

- Send spear-phishing email with a link to a specific individual
  - Watch for access attempts outside the expected geographic area
- Design an exploit that logs infections
  - In a blog comment, Twitter, Pastebin, etc.
- Embed an unused domain in malware
  - Watch for attempts to resolve the domain

Safely Investigate an Attacker Online

Indirection Tactics

- Proxy server, Tor, Web-based anonymizer
  - Not subtle—it's obvious that you are hiding
- Use a dedicated VM for research
  - Hide its location with a cellular or VPN connection
- Use an ephemeral cloud machine
  - Such as an Amazon E2C virtual machine

Search Engines

- Usually safe
- If the domain was previously unknown to the search engine, it may be crawled
- Clicking results still activates secondary links on the site
  - Even opening cached resources

Getting IP Address and Domain Information

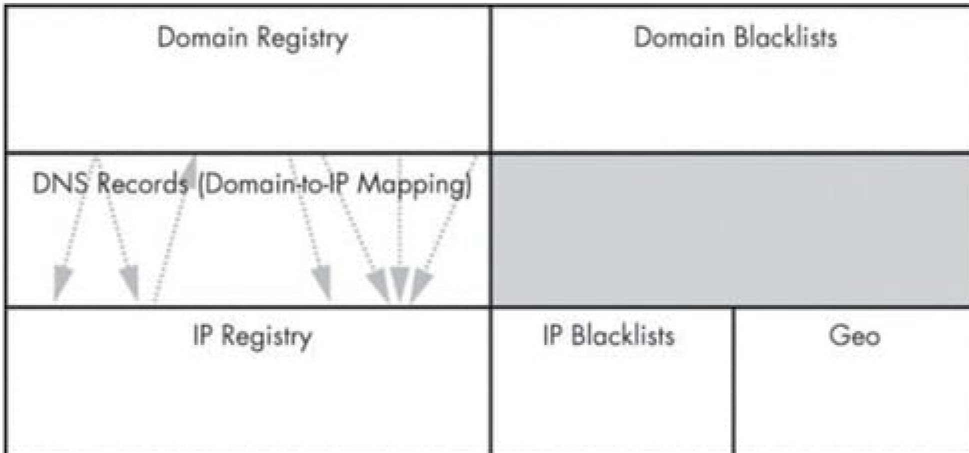


Figure 15-1. Types of information available about DNS domains and IP addresses

Command-Line v. Web-Based Lookups

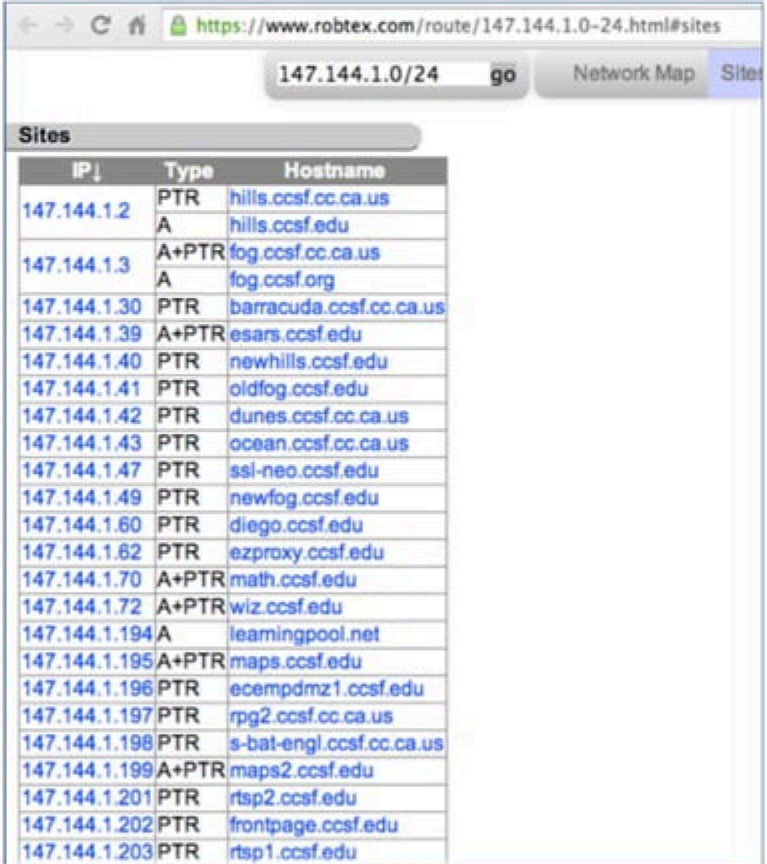
- **whois** and **dig** can be used, but they will expose your IP address
- Websites that do the query for you provide anonymity
  - May give more information

DomainTools

- Historical DNS records
- Reverse IP lookups
- Reverse whois (lookup based on contact information metadata)

RobTex

- Finds multiple domain names that point to a single IP address
- Checks blacklists

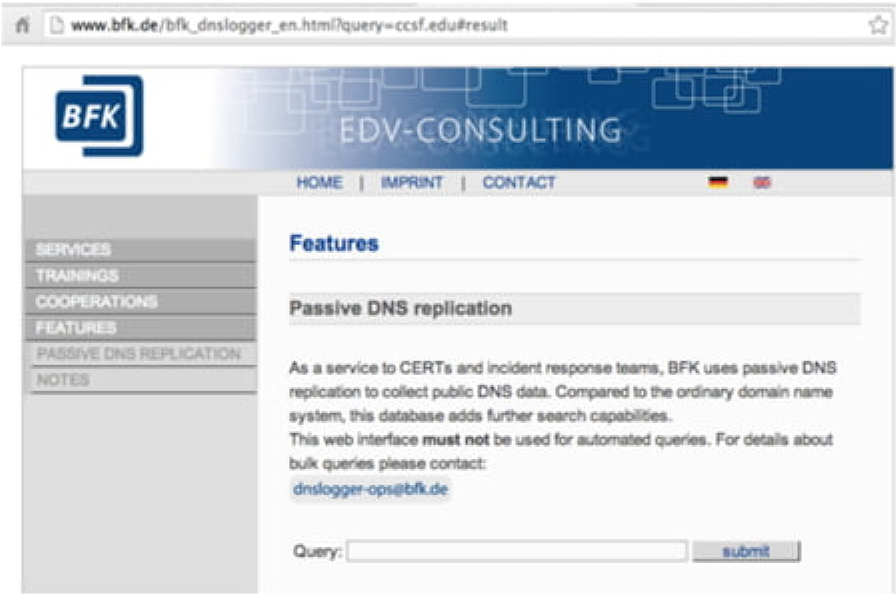


Sites

IP↓	Type	Hostname
147.144.1.2	PTR	hills.ccsf.cc.ca.us
	A	hills.ccsf.edu
147.144.1.3	A+PTR	fog.ccsf.cc.ca.us
	A	fog.ccsf.org
147.144.1.30	PTR	barracuda.ccsf.cc.ca.us
147.144.1.39	A+PTR	esars.ccsf.edu
147.144.1.40	PTR	newhills.ccsf.edu
147.144.1.41	PTR	oldfog.ccsf.edu
147.144.1.42	PTR	dunes.ccsf.cc.ca.us
147.144.1.43	PTR	ocean.ccsf.cc.ca.us
147.144.1.47	PTR	ssl-neo.ccsf.edu
147.144.1.49	PTR	newfog.ccsf.edu
147.144.1.60	PTR	diego.ccsf.edu
147.144.1.62	PTR	ezproxy.ccsf.edu
147.144.1.70	A+PTR	math.ccsf.edu
147.144.1.72	A+PTR	wiz.ccsf.edu
147.144.1.194	A	learningpool.net
147.144.1.195	A+PTR	maps.ccsf.edu
147.144.1.196	PTR	ecempdmz1.ccsf.edu
147.144.1.197	PTR	rpg2.ccsf.cc.ca.us
147.144.1.198	PTR	s-bat-engl.ccsf.cc.ca.us
147.144.1.199	A+PTR	maps2.ccsf.edu
147.144.1.201	PTR	rtsp2.ccsf.edu
147.144.1.202	PTR	frontpage.ccsf.edu
147.144.1.203	PTR	rtsp1.ccsf.edu

BFK DNS Logger

- Gathers data with passive DNS monitoring
- Stealthy



Content-Based Network Countermeasures

Intrusion Detection with Snort

- Rule-based detection, can use:
  - TCP or IP headers
  - Size of payload
  - Connection state (such as ESTABLISHED)
  - Layer 7 payload data

Snort Rule to Block HTTP Traffic by User-Agent

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"TROJAN Malicious User-Agent";
content:"|0d 0a|User-Agent\: Wefa7e"; classtype:trojan-activity; sid:2000001; rev:1;)
```

Table 15-2. Snort Rule Keyword Descriptions

Keyword	Description
`msg`	The message to print with an alert or log entry
`content`	Searches for specific content in the packet payload (see the discussion following the table)
`classtype`	General category to which rule belongs
`sid`	Unique identifier for rules
`rev`	With `sid`, uniquely identifies rule revisions

Taking a Deeper Look

- Running the malware several times shows these User-Agent strings
- Rules can be fine-tuned to capture the malware without false positives

We4b58	We7d7f	Wea4ee
We70d3	Wea508	We6853
We3d97	We8d3a	Web1a7
Wed0d1	We93d0	Wec697
We5186	We90d8	We9753
We3e18	We4e8f	We8f1a
Wead29	Wea76b	Wee716

Combining Dynamic and Static Analysis Techniques

Two Objectives of Deeper Analysis

- Full coverage of functionality
  - Provide new inputs to drive the malware down unused paths
  - Using iNetSim or custom scripts
- Understanding functionality, including inputs and outputs
  - Static analysis finds where and how content is generated
  - Dynamic analysis confirms the expected behavior

Danger of Overanalysis

Table 15-4. Malware Analysis Levels

Analysis level	Description
Surface analysis	An analysis of initial indicators, equivalent to sandbox output
Communication method coverage	An understanding of the code for each type of communication technique
Operational replication	The ability to create a tool that allows for full operation of the malware (a server-based controller, for example)
Code coverage	An understanding of every block of code

Hiding in Plain Sight

- Attackers mimic existing protocols
  - Often HTTP, HTTPS, and DNS
  - HTTP for beaconing (request for instructions)
  - HTTPS hides the nature and intent of communications
  - Information can be transmitted in DNS requests
    - For example, in long domain names

GETs

- Used to send a command prompt followed by a directory listing

```
GET /world.html HTTP/1.1
User-Agent: %^&NQvtmw3eVhTfEBnzVw/aniIqQB6qQgTvmxJzVhjqJMjcHtEhI97n9+yy+duq+h3
b0RFzThrfE9AkK9OYIt6bIM7JUQJdViJaTx+q+h3dm8jJ8qfG+ezm/C3tnQgvVx/eECBZT87NTR/fU
QkxmgcGLq
Cache-Control: no-cache

GET /world.html HTTP/1.1
User-Agent: %^&EBTaVDPYTM7zVs7umwvhTM79ECrrmd7ZVd7XSQFvV8jJ8s7QVhcgVQOqOhPdUQB
XEAkgVQFvms7zmd6bJtSfHNSdJNEJ8qfGEA/zmwPtnC3d0M7aTs79KvcAVhJgVQPZnDIqSQkuEBJvn
D/zVwneRAyJ8qfGIN6aIt6aIt6cI86qI9mlIe+q+OfqE86qLA/FOtjqE86qE86qE86qHqfGIN6aIt6
aIt6cI86qI9mlIe+q+OfqE86qLA/FOtjqE86qE86qE86qHsjJ8tAbHeEbHeEbIN6qE96jKt6kEABJE
86qE9cAMPE4E86qE86qE86qEA/vmhYfVi6J8t6dHe6cHeEbI9uqE96jKtEkEABJE86qE9cAMPE4E86
qE86qE86qEATrnw3dUR/vmbfGIN6aINAaIt6cI86qI9ulJNmq+OfqE86qLA/FOtjqE86qE86qE86qN
Ruq/C3tnQgvVx/e9+ybIM2eIM2dI96kE86cINygK87+NM6qE862/AvMLs6qE86qE86qE87NnCBdn87
JTQkg9+yqE86qE86qE86qE86qE86bEATzVCOymduqE86qE86qE86qE86qE96qSxvfTRIJ8s6qE86qE
86qE86qE86qE9Sq/CvdGDIzE86qK8bgIeEXItObH9SdJ87s0R/vmd7wmwPv9+yJ8uIlRA/aSiPYTQk
fmd7rVw+qOhPfnCvZTiJmMtj
Cache-Control: no-cache
```

User Agents

- Early malware used strange User-Agent strings
- This made it easy to block
- Valid user agent:

```
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727;
.NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET4.0C; .NET4.0E)
```

3 Possible User Agents

- Malware alternates between these to defeat detection

```
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; .NET CLR 1.1.4322)
```

Attackers Use Existing Infrastructure

- Botnet commands concealed in source code of a Web page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>  Roaring Capital | Seed Stage Venture Capital Fund in Chicago</title>
<meta property="og:title" content="  Roaring Capital | Seed Stage Venture
Capital Fund in Chicago"/>
<meta property="og:site_name" content="Roaring Capital"/>
<!--  -->
<!-- adsrv?bG9uZ3NsZWVw -->
<!--<script type="text/javascript" src="/js/dotastic.custom.js"></script>-->
<!-- OH -->
```

Leveraging Client-initiated Beaconing

- Hosts behind NATs or proxy servers have a concealed IP address
- Makes it difficult for attackers to know which bot is phoning home
- Beacon identifies host with an unique identifier
  - Such as an encoded string with basic information about the host

Understanding Surrounding Code

- Malware beacon

```
GET /1011961917758115116101584810210210256565356 HTTP/1.1
Accept: * / *
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)
Host: www.badsite.com
Connection: Keep-Alive
Cache-Control: no-cache
```

- URIs

```
/1011961917758115116101584810210210256565356 (actual traffic)
/14586205865810997108584848485355525551
/7911554172581099710858484848535654100102
/2332511561845810997108584848485357985255
```

Table 15-5. Windows Networking APIs

WinSock API	WinINet API	COM interface
WSAStartup	InternetOpen	URLDownloadToFile
getaddrinfo	InternetConnect	CoInitialize
socket	InternetOpenURL	CoCreateInstance
connect	InternetReadFile	Navigate
send	InternetWriteFile	
recv	HTTPOpenRequest	
WSAGetLastError	HTTPQueryInfo	
	HTTPSendRequest	

Example Malware

- Uses `InternetOpen` and `HTTPOpenRequest`
- URI is generated from calls to
  - `GetTickCount`, `Random`, `gethostbyname`

Sources of Network Content

- Random data
- Data from networking libraries
  - Such as the GET created from a call to HTTPSendRequest
- Hard-coded data
- Data about the host and its configuration
  - Hostname, current time, CPU speed
- Data received from other sources
  - Remote server, file system, keystrokes

Hard-Coded vs. Ephemeral Data

- Malware using lower-level networking APIs such as Winsock
  - Requires more manually-generated content to mimic common traffic
  - More hard-coded data
  - Likely the author makes a mistake that leaves a signature in the network traffic
  - May misspell a word like Mozilla

How URI is Generated

```
<4 random bytes>:<first three bytes of hostname>:<time from
GetTickCount as a hexadecimal number>
```

Identifying and Leveraging the Encoding Steps

Table 15-6. Regular Expression Decomposition from Source Content

<4 random bytes>	:	<first 3 bytes of hostname>	:	<time from GetTickCount>
0x91, 0x56, 0xCD, 0x56	:	"m", "a", "l"	:	00057473
0x91, 0x56, 0xCD, 0x56	0x3A	0x6D, 0x61, 0x6C	0x3A	0x30, 0x30, 0x30, 0x35, 0x37, 0x34, 0x37, 0x33
1458620586	58	10997108	58	4848485355525551
(([1-9]|1[0-9]|2[0-5]){0,1}[0-9]){4}	58	[0-9]{6,9}	58	(4[89]|5[0-7]|9[789]|10[012]){8}

Creating a Signature

- Avoid excessive complexity
  - Slows down the IDS
- Include enough detail to eliminate false positives

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"TROJAN Malicious Beacon ";
content:"User-Agent: Mozilla/4.0 (compatible\; MSIE 7.0\; Windows NT 5.1)";
content:"Accept: * / *"; uricontent:"58"; content:!"|0d0a|referer:"; nocase;
pcre:"/GET \/([12]{0,1}[0-9]{1,2}){4}58[0-9]{6,9}58(4[89]|5[0-7]|9[789]|10[012]){8}
HTTP/";
classtype:trojan-activity; sid:2000002; rev:1;)
```

Analyzing the Parsing Routines

- Malware strings and the Web page comments both include the string **adsrv?**

```
<!-- adsrv?bG9uZ3NsZWVw -->
```

- Parser looks for 3 elements
- <!–
- text
- -->

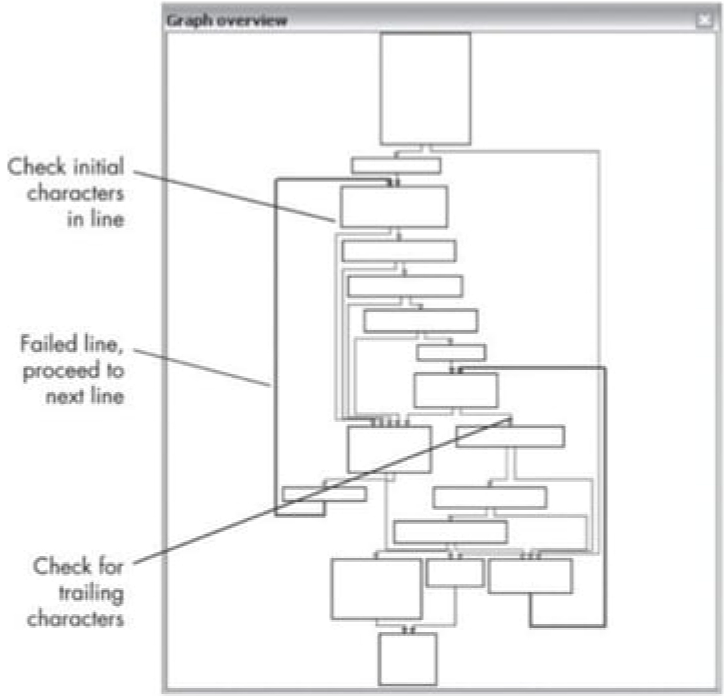


Figure 15-3. An IDA Pro graph of a sample parsing function

Table 15-7. Sample Malware Commands

Command example	Base64 translation	Operation
`longsleep`	`bG9uZ3NsZWVw`	Sleep for 1 hour
`superlongsleep`	`c3VwZXJsb25nc2xlZXA=`	Sleep for 24 hours
`shortsleep`	`c2hvcnRzbGVlcA==`	Sleep for 1 minute
`run:www.example.com/fast.exe`	`cnVuOnd3dy5leGFtcGxlLmNvbS9mYXN0LmV4ZQ==`	Download and execute a binary on the local system
`connect:www.example.com:80`	`Y29ubmVjdDp3d3cuZXhhbXBsZS5jb206ODA=`	Use a custom protocol to establish a reverse shell

Possible Signatures

- The five possible commands
- These will work, but any change in the malware will evade them

```
<!-- adsrv?bG9uZ3NsZWVw -->
<!-- adsrv?c3VwZXJsb25nc2xlZXA= -->
<!-- adsrv?c2hvcnRzbGVlcA== -->
<!-- adsrv?cnVu
<!-- adsrv?Y29ubmVj
```

Targeting Multiple Elements

- These are more general
- The first one accepts any Base64 in a comment with the adsrv prefix

```
pcre:"/<!-- adsrv\?([a-zA-Z0-9+\/=]{4})+ -->/"
content:"<!-- "; content:"bG9uZ3NsZWVw -->"; within:100;
content:"<!-- "; content:"c3VwZXJsb25nc2xlZXA= -->"; within:100;
content:"<!-- "; content:"c2hvcnRzbGVlcA== -->"; within:100;
content:"<!-- "; content:"cnVu";within:100;content: "-->"; within:100;
content:"<!-- "; content:"Y29ubmVj"; within:100; content:"-->"; within:100;
```

Making General Signatures

Target 1: User-Agent string, `Accept` string, no referrer
Target 2: Specific URI, no referrer

- Demo: capture GET in Wireshark
- User-Agent and Accept always appear together for normal browser traffic

Understanding the Attacker's Perspective

Rules of Thumb

- Focus on elements of the protocol that are part of both end points
  - Look for elements that use code on both the client and server
  - It will be hard for the attacker to change them both

Rules of Thumb

- Focus on elements of the protocol known to be part of a key
  - Such as a User-Agent that identifies bot traffic
  - Again, it would require updating both ends to change
- Identify elements of the protocol that are not immediately apparent in traffic
  - This will be less likely to be used by other, sloppy, defenders who leak info to the attacker