Detecting the ghost in the browser: Real time detection of drive-by infections

Thijs Kinkhorst Michael van Kleij

1 July 2009

Nine-Ball hacker attack rolls on

Web ad sales open door to viruses

Microsoft sounds alarm about PDF-attacks

Mass injection, Nine-ball infects more than 40,000 legitimate web sites

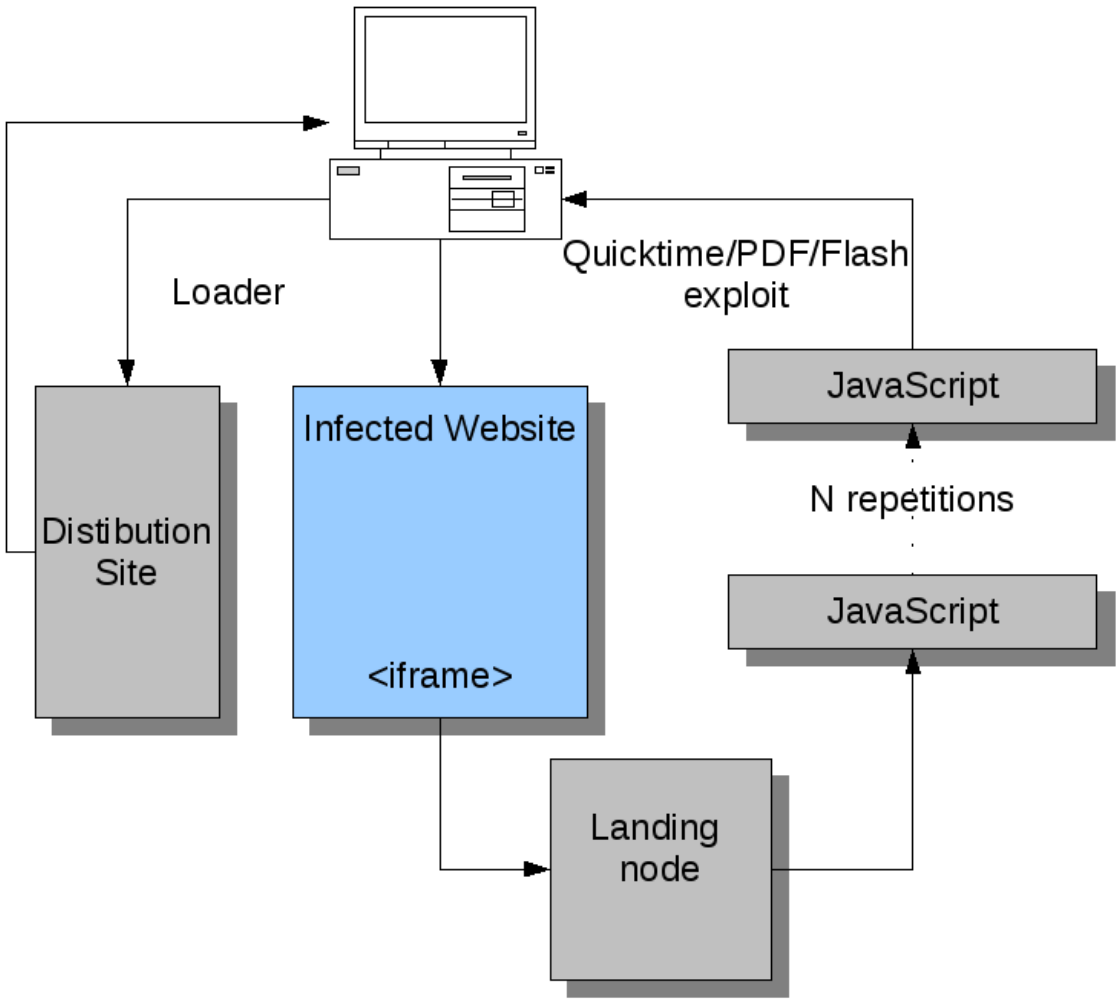
Canadian MSN site Sympatico compromised

What is a drive-by infection?

- Legitimate websites are compromised.
- An iFRAME is included which points to a browser exploit.

Example

```
<iframe src="http://globalnameshop.cn:8080/index.php"
width=153 height=102 style="visibility:
hidden"></iframe>
```


Loader
Quicktime/PDF/Flash
exploit
Distibution
Site
Infected Website
<iframe>
JavaScript
N repetitions
JavaScript
Landing
node

Research question

Can drive-by infections be discerned from legitimate sessions purely by measuring changes in HTTP traffic patterns and meta data?

- Enables detection by monitoring the local network.
- Low chance on false positives or false negatives.

Scope

- Detection via network traffic, not on the client machine.
- Not HTTP content inspection, no signature matching.
- Only infections that require no user interaction.
- Just the infection itself, not subsequent behaviour of the malware.

Lab setup and dataset composition

- Infected sites in our dataset are found using `www.malwaredomainlist.com` and similar.
- Each site was tested in a clean virtual machine using a test protocol.
- Test protocol consists of:
  - Start capture.
  - Visit site, wait 2 minutes.
  - Close browser, wait 2 minutes.
  - Shut down machine.
  - Restore machine to clean state, rotate IP address.
- Capturing both clean sessions and infected sessions.

Analysis

- **TCP port numbers.**
- Geographical locations.
- Hostnames.
- **User agents.**
- Invalid POST requests.
- Request URIs.
- Content types.
- **Redirection.**

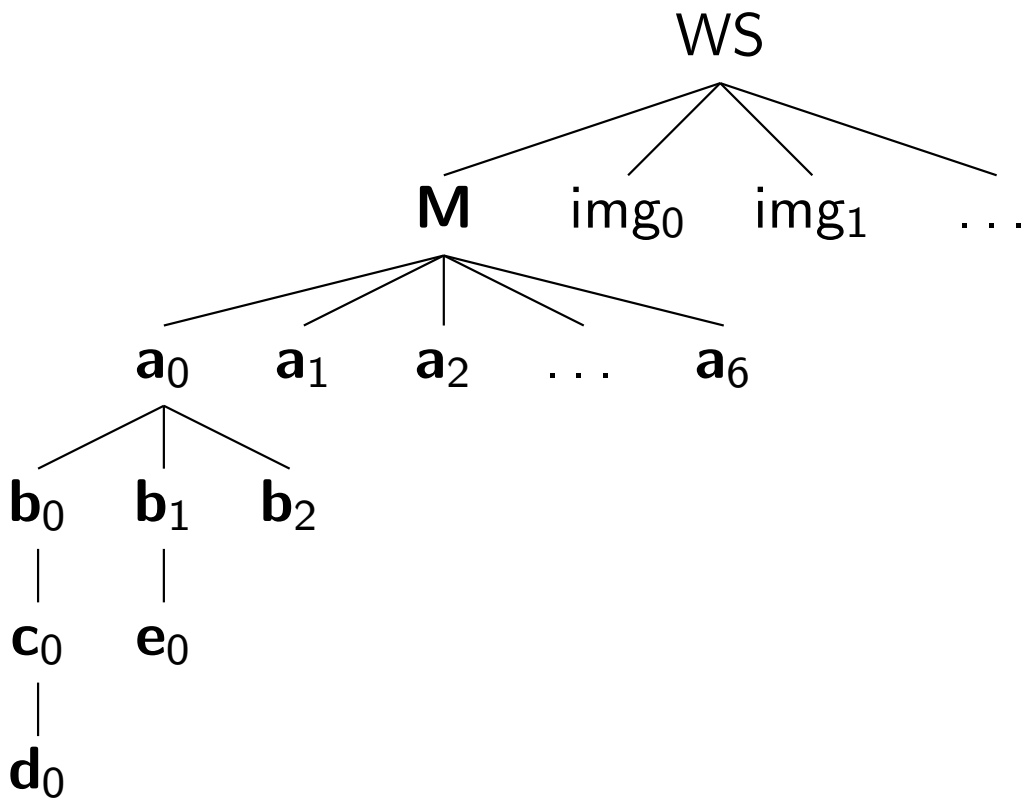
TCP port numbers

	Clean		Infected	
	# of sess.	% of sess.	# of sess.	% of sess.
port 80	39	100%	25	100%
port 8080	0	0%	10	40%
port 443	3	8%	2	8%

User agents

	Clean sess.	Infected sess.
Total number of different User-Agent headers found	2	9
Average number of unique User-Agent headers per session	1.0	2.5
Average number of requests with non-original User-Agent per session	0.2	6.1

Redirection



Detection method

- Scoring like SpamAssassin.
- Combine inconclusive information into high-confidence verdict.
- Flexible and expandable

$score \leftarrow 0$

$firstrequest \leftarrow$ front($capture$)
for all $rule \in ruleset$ **do**
  $score \leftarrow score +$ min ($rule(capture, firstrequest)$, 4.0)
end for

return $score \geq 5.0$

TCP port numbers

Rule 1 Detecting 'bad' ports

function Rule (*capture*, *firstreq*) : *s*
for all *request* $\in$ *capture* **do**
  if *request.port* $\notin \{80,443\} \wedge$ *request.port* $\neq$ *firstreq.port* **then**
    return 2.0
  end if
end for
return 0
end function

User Agents

Rule 6 Detecting 'bad' user agents

function Rule (*capture*, *firstreq*) : *s*
$s \leftarrow 0$
for all *request* $\in$ *capture* **do**
  if *request.useragent* $\notin \alpha \wedge$
  request.useragent $\neq$ *firstreq.useragent* **then**
    $s \leftarrow s + 0.4$
  end if
end for
return *s*
end function

α: whitelist of special user agents, like Adobe Updater.

Redirection Trees

Rule 10 Analysing redirection trees

function Rule (*capture*, *firstreq*) : s
$T \leftarrow$ BuildRedirectionTree (*capture*)
return min (max (0, height(T) $-$ 2), 2.0)
end function

height(T) $\leq 2 \Rightarrow 0.0$
height(T) $= 3 \Rightarrow 1.0$
height(T) $\geq 4 \Rightarrow 2.0$

Validation

- Collected second, separate dataset for testing usefulness.
- Consists of 20 legitimate and 15 infected captures.
- Apply our ruleset to this new data.

- True-negative rate: 14 out of 15 (93%)
- False-positive rate: 0 out of 20 (0%)

Validation: Infected Sessions

rule	1	2	3	4	5	6	7	8	9	10	$\sum$
session 1	2.0	2.0		1.0		0.4	2.0	2.0	0.4		9.8
session 2	2.0	2.0	1.5	1.0	2.0	0.8	2.0	3.0	0.4		14.7
session 3			1.5	1.0		4.0		1.0	1.2		8.7
session 4			1.5		2.0	0.4			1.6	2.0	7.5
session 5			1.5		2.0	0.8			1.8	2.0	8.1
session 6			1.5		2.0	0.8			0.4	2.0	6.7
session 7	2.0	2.0		1.0				1.0	0.4		6.4
session 8			1.5		2.0	0.4			2.2	2.0	8.1
session 9	2.0	2.0		1.0			2.0	2.0	0.4		9.4
session 10	2.0	2.0		1.0			2.0	3.0	0.4		10.4
session 11			1.5		2.0	0.8			1.0	2.0	7.3
session 12			1.5		2.0	0.8			2.0	2.0	8.3
session 13		2.0	1.5	1.0	2.0	1.2		1.0	0.2		8.9
session 14			1.5						0.2		**1.7**
session 15		2.0	1.5		2.0	1.6		1.0		1.0	9.1

Validation: Clean Sessions

rule	1	2	3	4	5	6	7	8	9	10	$\sum$
session 1			1.5								1.5
session 2											0.0
session 3			1.5								1.5
session 4											0.0
session 5											0.0
session 6											0.0
session 7											0.0
session 8			1.5								1.5
session 9											0.0
session 10											0.0
session 11											0.0
session 12											0.0
session 13											0.0
session 14					2.0						2.0
session 15											0.0
session 16											0.0
session 17											0.0
session 18										1.0	1.0
session 19											0.0
session 20			1.5								1.5

Conclusion

Can drive-by infections be discerned from legitimate sessions purely by measuring changes in HTTP traffic patterns and meta data?

Yes, it is possible.

Focus points:

- Scoring and rules need more real-life improvement.
- Identification of a session may be problematic.

Thank You

Questions?