Hide’n’Seek
An Adaptive Peer-to-Peer Botnet

Adrian Șendroiu
Vladimir Diaconescu
Context

- IoT Botnets increasing in impact and diversity
- Tried and tested models (Mirai)
  - Central C2 Server
  - (Different) Infecting machine
  - (Different) Reporting machine
- Dictionary and CVE extensions
- Main objective: DDoS
Analysis - Overview

- A new idea: Peer-to-Peer botnet
  - Also seen in Hajime
- Custom protocol
- Modular
- Authenticated
- Different goals
Functionality

- Two main components
  - Scanner
Functionality

- Two main components
  - Scanner
  - P2P protocol
Scanning for victims

- Pick a random IP and a port
  - 23, 2323 (telnet) - try default credentials
  - 80, 8080 (http) - try known IoT exploits
  - 5555 - ADB
  - Others (2480, 5984)
Analysis - Infection process
Analysis - Infection process: Scan

TCP SYN :23

ACK

SCAN
Analysis - Infection process: Connect

Get login prompt

“login:”

SCAN  
CONN  
...
Analysis - Infection process: Dictionary
Analysis - Infection process: Sysinfo

cat /proc/cpuinfo

ARM

SCAN ➔ CONN ➔ DICT ➔ INFO
Analysis - Infection process: Probing

wget
curl
base64

wget: not found
curl: not found
Analysis - Infection process: Dropping

- “echo -e \x7fELF…’ > abc”
- chmod +x abc
Analysis - Infection process: Dropping

- ./abc a1.2.3.4:5678 k23 l4444 e5.4.3.2:80
Analysis - Infection process: Dropping

- ./abc a1.2.3.4:5678 k23 l4444 e5.4.3.2:80
  - Initial starting peers
Analysis - Infection process: Dropping

- ./abc a1.2.3.4:5678 k23 l4444 e5.4.3.2:80
  - Initial starting peers
  - Kill port
Analysis - Infection process: Dropping

- ./abc a1.2.3.4:5678 k23 l4444 e5.4.3.2:80
  - Initial starting peers
  - Kill port
  - P2P listening port (UDP)
Analysis - Infection process: Dropping

- ./abc a1.2.3.4:5678 k23 l4444 e5.4.3.2:80
  - Initial starting peers
  - Kill port
  - P2P listening port (UDP)
  - Additional scan targets
P2P protocol

● Custom UDP protocol
  ○ Data structures
  ○ Messages
## Data structures - Peer table

<table>
<thead>
<tr>
<th>IP</th>
<th>port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.3.4</td>
<td>20123</td>
</tr>
<tr>
<td>5.6.7.8</td>
<td>30456</td>
</tr>
<tr>
<td>4.3.2.1</td>
<td>40789</td>
</tr>
</tbody>
</table>
Data structures - Caches

payload id    ->  hash
0x15            -> 1af3…
0x13            -> 3f14...

hash    ->  data
3f14… -> \x7fELF...
1af3… -> \x7fELF…

config cache

data cache
$ cat /proc/cpuinfo
$ cat /proc/cpuinfo

model name : ARMv7 Processor rev 1

• What to download?
model name: ARMv7 Processor rev 1

payload id -> hash
0x15 -> 1af3...
0x13 -> 3f14...

hash -> data
3f14... -> \x7fELF...
1af3... -> \x7fELF...

config cache
data cache
The model name of the processor is ARMv7 Processor rev 1.

The payload ids and their corresponding hashes are:
- **Payload ID**: 0x15  ->  Hash: 1af3...
- **Payload ID**: 0x13  ->  Hash: 3f14...

The hashes and their corresponding data are:
- **Hash**: 3f14...  ->  Data: \x7fELF...
- **Hash**: 1af3...  ->  Data: \x7fELF...

To view the CPU information, use the command:

```
$ cat /proc/cpuinfo
```
model name    : ARMv7 Processor rev 1

payload id    -> hash
0x15            -> 1af3…
0x13            -> 3f14…

hash    -> data
3f14… -> \x7fELF…
1af3… -> \x7fELF…
model name: ARMv7 Processor rev 1

payload id -> hash
0x15 -> 1af3...
0x13 -> 3f14...

hash -> data
3f14... -> \x7fELF...
1af3... -> \x7fELF...

cat /proc/cpuinfo
model name : ARMv7 Processor rev 1

payload id -> hash
0x15 -> 1af3...
0x13 -> 3f14...

hash -> data
3f14... -> \x7fELF...
1af3... -> \x7fELF...

cfg config cache
data cache

$ cat /proc/cpuinfo
$ echo -e '\x7fELF…' > abc; ./abc

payload id    ->  hash
0x15           ->  1af3…
0x13           ->  3f14…

hash    ->  data
3f14…   ->  \x7fELF…
1af3…   ->  \x7fELF…

config cache

data cache
P2P protocol messages

- Config cache update
- Peer management
- Target reporting

![Diagram showing 1 byte id and message data]
Config cache update

payload id -> hash
## Config cache update

<table>
<thead>
<tr>
<th>payload id</th>
<th>hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x15</td>
<td>1af3...</td>
</tr>
<tr>
<td>0x13</td>
<td>3f14...</td>
</tr>
</tbody>
</table>
Config cache update

h - config cache version report
Config cache update

H - config cache version reply
Config cache update

y - chunk request
Y - chunk reply

Y 0  data

check signature
$ cat /proc/cpuinfo

model name : ARMv7 Processor rev 1

payload id -> hash

0x15 -> 1af3...
0x13 -> 3f14...

hash -> data

3f14... -> \x7fELF...
1af3... -> \x7fELF...

config cache
data cache
Config cache update

<table>
<thead>
<tr>
<th>y</th>
<th>0</th>
</tr>
</thead>
</table>

Y 0 data

check signature

y - chunk request
Y - chunk reply
Data cache update

hash -> data

3f14… -> \x7fELF...
1af3… -> \x7fELF…
Peer update

~ - peer request
Peer update

- peer reply
Target device reporting

GET /some/exploit
Target device reporting

HTTP/1.1 404 Not Found

GET /some/exploit
Target device reporting

z 1.2.3.4 80
Target device reporting

GET /some/exploit
Target device reporting

GET /some/exploit

HTTP/1.1 200 OK
Hide’n’Seek - Summary

- Infects many kinds of IoT systems
- Decentralized P2P architecture
- Network controlled by the author
Updates
Updates

- ~30 samples
  - Code refactoring
  - New functionality
Updates

- ~30 samples
  - Code refactoring
  - New functionality
- Persistency (copy itself to /etc/init.d/S99abcd)
Updates

● ~30 samples
  ○ Code refactoring
  ○ New functionality

● Persistency (copy itself to /etc/init.d/S99abcd)

● Dropping other binaries
  ○ cpuminer
Updates

- HTTP-based exploits for more IoT vendors
Updates

- HTTP-based exploits for more IoT vendors
- RCE via OrientDB and CouchDB
Updates

- HTTP-based exploits for more IoT vendors
- RCE via OrientDB and CouchDB
- Hijacking devices via ADB
Monitoring

Hardcoded peers top 10

- South Korea: 106
- China: 78
- United States: 36
- Germany: 23
- Russia: 20
- France: 10
- Turkey: 9
- Vietnam: 8
- Hong Kong: 7
- Poland: 6
Monitoring

\[ \wedge 1.2.3.4 123 \]
\[ \wedge 5.6.7.8 456 \]
\[ \wedge 4.3.2.1 789 \]
Monitoring

- ~300,000 peers
Conclusions

- A new trend in the IoT landscape
  - Qbot, Mirai
Conclusions

● A new trend in the IoT landscape
  ○ Qbot, Mirai
  ○ Hajime
Conclusions

● A new trend in the IoT landscape
  ○ Qbot, Mirai
  ○ Hajime, Satori
Conclusions

- A new trend in the IoT landscape
  - Qbot, Mirai
  - Hajime, Satori, Reaper
Conclusions

● A new trend in the IoT landscape
  ○ Qbot, Mirai
  ○ Hajime, Satori, Reaper, VPNFilter
Conclusions

● A new trend in the IoT landscape
  ○ Qbot, Mirai
  ○ Hajime, Satori, Reaper, VPNFilter, HNS
Conclusions

- A new trend in the IoT landscape
  - Qbot, Mirai
  - Hajime, Satori, Reaper, VPNFilter, HNS
- More threats to come
Q&A