TOWARDS SYSTEMATIC HONEYTOKEN FINGERPRINTING

SINCONF 2020

SHREYAS SRINIVASA, JENS MYRUP PEDERSEN, EMMANOUIL VASILOMANOLAKIS

AALBORG UNIVERSITY, DENMARK

AALBORG UNIVERSITY
DENMARK

center for Communication, Media and Information technologies

Honeytokens

- **Honeypots** are deception systems that emulate the **services** of an end system

- **Honeytoken** is an umbrella term for honeypot-like entities/resources that can be deployed on a network or a system
- Honeytokens emulate a **resource** and hence are light-weight and flexible

- Honeytokens are efficient to detect indirect attacks (malware) and direct attacks like unauthorized access
- Popular honeytokens include the open source service Canarytokens [1]

Honeytoken operation example

Related work: Honeypot Fingerprinting

- The process of determining that the vulnerable end system is indeed a honeypot

- Honeypot Fingerprinting relies on [2] [3]:
  - observing for static response,
  - partial or
  - invalid response due to limited simulation or library dependency

- This the first attempt towards Honeytoken Fingerprinting

Honeytoken Fingerprinting Techniques

- Honeytokens are classified based on operation levels – System, Network, Data and File
  - For example: a fake user access information in a database that operates at the data level
- Fingerprinting techniques are based on these operational levels

Honeytoken	Deceptive Entity/Resource	Alerting Mechanism
Honeyentries [4],[12]	Table data set	DB Monitor
Honeyword [14]	Password	DB Monitor
Honeyaccount [8]	User-account	Event Logger
Honeyfile [17]	File-Google Sheets	Session Log
Honeyfile [10]	File	Event Logger
Honeypatch [1], [2]	Vulnerability	Session Log
HoneyURL [17]	URL	DNS Trigger
CanaryTrap [7]	Email	Email
Honeyport [10]	Network port	Session Log
CanaryToken [26]	File-pdf, docx	DNS Trigger
CanaryToken [26]	Directory	DNS Trigger
CanaryToken [26]	URL	DNS Trigger
Honeybits[15]	Email	DNS Trigger

Alerting Mechanism	Operating Level	Fingerprinting Technique
DB Monitor	Data	Modified Date
Event Logger	System	Last Used, grep search
Session Log	Application	Grep search
DNS Trigger	Application, Network	Reverse Engineering Network Sniffing

Proof of concept

- We fingerprint honeytokens generated through the open source Canarytoken service
- Canarytokens provide honeytokens:
  - files(pdf, docx, exe, dll),
  - directories,
  - URLs,
  - image embeds
  - Etc.
- We propose fingerprinting techniques for the pdf, docx and the directories through
  - decomposition
  - reverse engineering techniques

Canary Token - docx

- We exploit the alerting mechanism
- Employ reverse engineering
- Rename the file extension from docx to .zip (compressed)
- Unzip the the zip folder to find xml files
- In the footer.xml file, we find a DNS call made to a Canary Tokens domain

```
xmlns:wpc="http://schemas.microsoft.com/office/word/2010/wordprocessingCanvas">
    <w:bookmarkStart w:name="_GoBack" w:id="0"/>
    <w:bookmarkEnd w:id="0"/>
  - <w:p w:rsidRDefault="009E0DC7" w:rsidR="009E0DC7">
      - <w:pPr>
            <w:pStyle w:val="Footer"/>
        </w:pPr>
      - <w:r>
            <w:fldChar w:fldCharType="begin"/>
        </w:r>
      - <w:r>
            <w:instrText xml:space="preserve"> INCLUDEPICTURE
              "http://canarytokens.com/about/lgj266un6nsmfbqpfyvnwym0w/submit.aspx"
              \d \* MERGEFORMAT </w:instrText>
        </w:r>
```

PDF & Windows Directory Honeytoken Fingerprinting

- exploit the hardcoded URL in the embedded triggering mechanism
- Decompose the pdf file by parsing it (python)
- The pdf file contains an embedded hidden object that makes DNS call to a Canarytokens domain

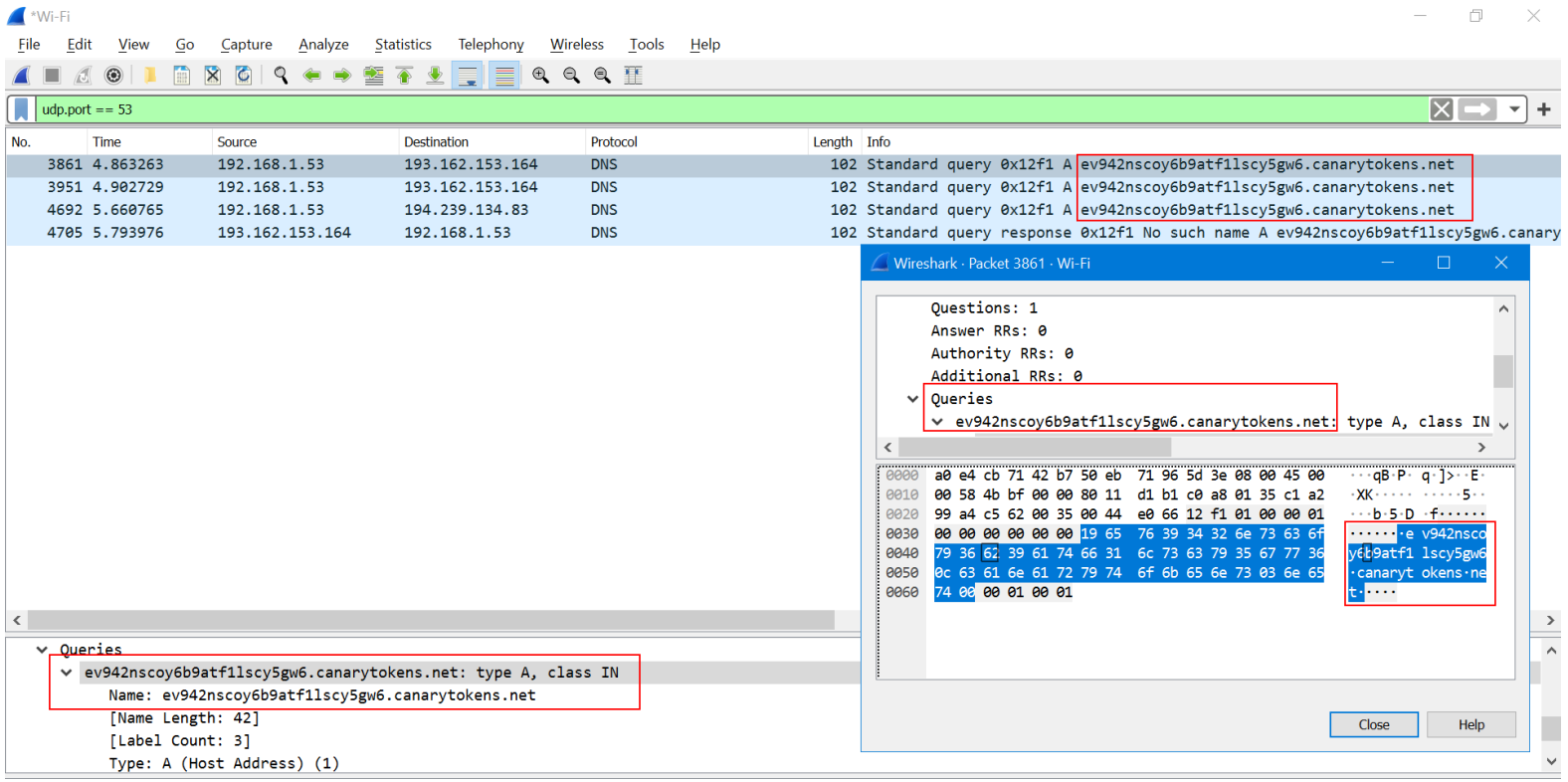
- Directory honeytoken contains a hidden .ini file
- The .ini file is configured to make a DNS call when the directory is accessed

PDF, Directory Honeytoken Fingerprinting

```
(base) C:\AAU_NetSec\tokengrabber>python folder.py --d C:\Employee_PaySlips
Found:  C:\Employee_PaySlips\My Documents\desktop.ini
Canary token detected in file:  C:\Employee_PaySlips\My Documents\desktop.ini

(base) C:\AAU_NetSec\tokengrabber>python pdf-parser.py -o 16 -O C:\Important\Salary.pdf
This program has not been tested with this version of Python (3.7.6)
Should you encounter problems, please use Python version 3.6.3
obj 16 0
 Containing /ObjStm: 14 0
 Type:
 Referencing:

  <<
    /S /URI
    /URI (http://[redacted].canarytokens.net/[redacted])
  >>



(base) C:\AAU_NetSec\tokengrabber>python docx.py -f C:\Important\Contract.docx
C:\AAU_NetSec\tokengrabber\.\temp\word\footer2.xml
Canarytoken detected
```

Passive DNS Fingerprinting

- Alerting mechanism works by triggering DNS requests
- Packet Sniffing can parse all requests made to the Canarytokens domain
- Disadvantage: Have to access the honeytoken to confirm the DNS calls




```
(base) C:\AAU_NetSec\tokengrabber>netsh interface ip show interfaces

Idx     Met         MTU          State                Name
---  ----------  ----------  ------------  ---------------------------
  1          75  4294967295  connected     Loopback Pseudo-Interface 1
 12          35        1500  connected     Wi-Fi
 17           5        1500  disconnected  Ethernet
  8          25        1500  disconnected  LAN-forbindelse* 1
 18          25        1500  disconnected  LAN-forbindelse* 2
 14          65        1500  disconnected  Bluetooth-netværksforbindelse
  5          25        1500  connected     Npcap Loopback Adapter
 28        5000        1500  connected     vEthernet (Default Switch)


(base) C:\AAU_NetSec\tokengrabber>python dns_sniffer.py -i Wi-Fi
Canarytoken detected!!
Canarytoken detected!!
```

Future Work

Extending the fingerprinting techniques to detect

- System level honeytokens (employing *inode*)
- Database level honeytokens
- User-account based honeytokens

Thank You!

Contact:

Shreyas Srinivasa

Email: shsr@es.aau.dk

AALBORG UNIVERSITY
DENMARK

References

- [1] Canarytokens, https://canarytokens.com/generate
- [2] Bitter Harvest - Vetterl A, Clayton R. Bitter harvest: Systematically fingerprinting low-and medium-interaction honeypots at internet scale. In12th {USENIX} Workshop on Offensive Technologies ({WOOT} 18) 2018.
- [3] Detect Me If You Can - Morishita S, Hoizumi T, Ueno W, Tanabe R, Gañán C, van Eeten MJ, Yoshioka K, Matsumoto T. Detect me if you… oh wait. An internet-wide view of self-revealing honeypots. In2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) 2019 Apr 8 (pp. 134-143). IEEE.