A Tête-a-Tête with RSA Bots

...

Jens Frieß and Laura Guevara
jens.friess@stud.tu-darmstadt.de
laura.guevara@fkie.fraunhofer.de

Nov. 30th- Dec. 2nd 2016, Lyon

Fraunhofer FKIE

Outline for this Talk

- Motivation
- Bypassing Asymmetric Encryption
  - Common Scenario
  - Approach
- Implementation
  - Method
  - Components
- Results
  - A Tête-a-Tête with UrlZone and Zeus Panda
- Conclusion
  - Limitations
  - Future Work

Motivation

Motivation

Why intercepting and tampering the bot‘s communication

- Daily work:
  - “How does family XYZ communicate with its C&C?”
- Extracting communication features
  - Obfuscated e.g., using strong cryptography

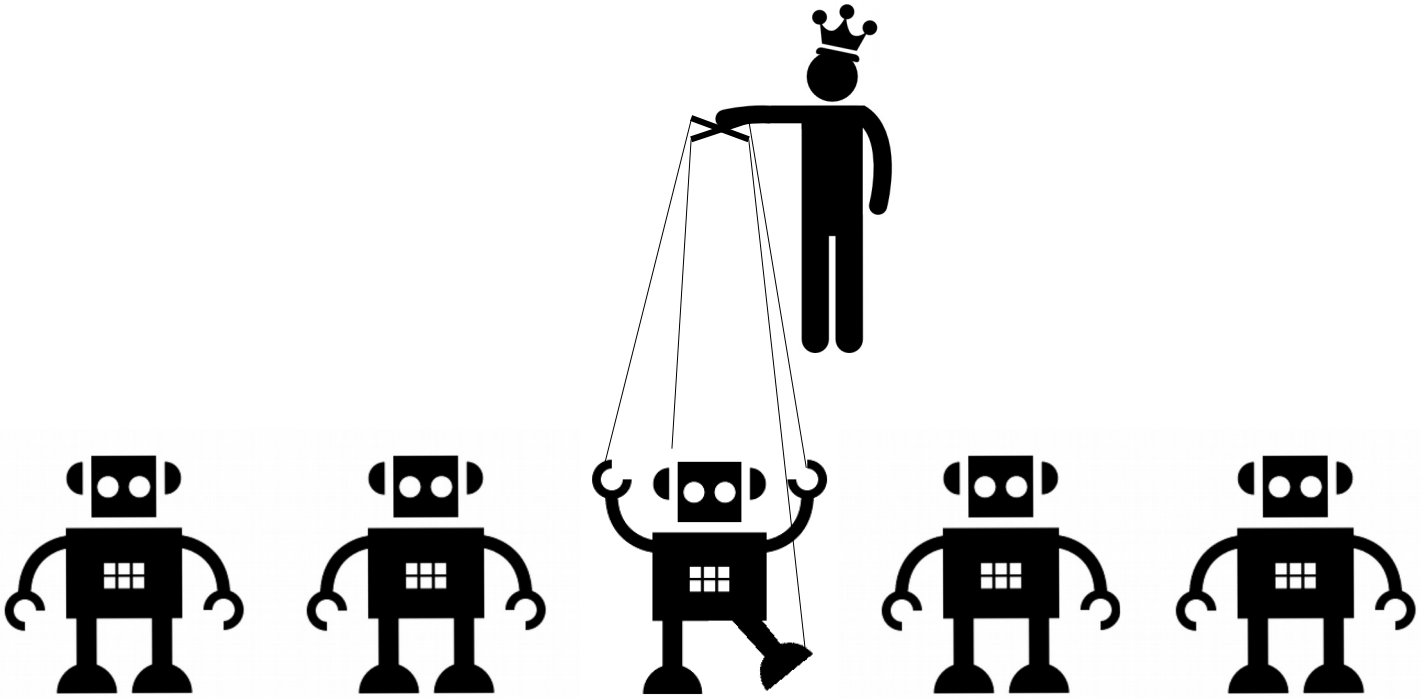
Goal

Why building automated tools

- Reduce reverse engineering efforts
  - Automated recovery of message structures
  - Bot Interaction → observe/induce behaviour
  - Version history

Goal

Why building automated tools



Observations

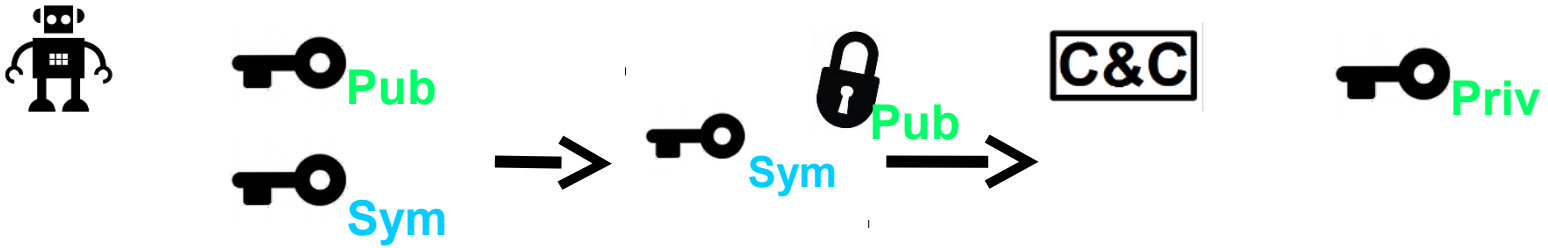
- Evolution of version within one family: families introduce minor modifications to their C&C protocol within new releases
- Malware authors seem to rely on standard cryptographic libraries: adding complex crypto-routines can be a source of errors reducing malware success

Bypassing Message Encryption

Common Scenario

1. Public Key is Hardcoded in Bot

2. The Bot generates a Symmetric Key and uses the Public Key for Key Exchange



3. The C&C obtains the Symmetric Key and sends Encrypted Commands to the Bot

Fraunhofer FKIE

Common Scenario

Challenge

- Asymmetric encryption:
  - C&C's private key is not available to decrypt bot messages
  - Symmetric key is not available → generated at runtime

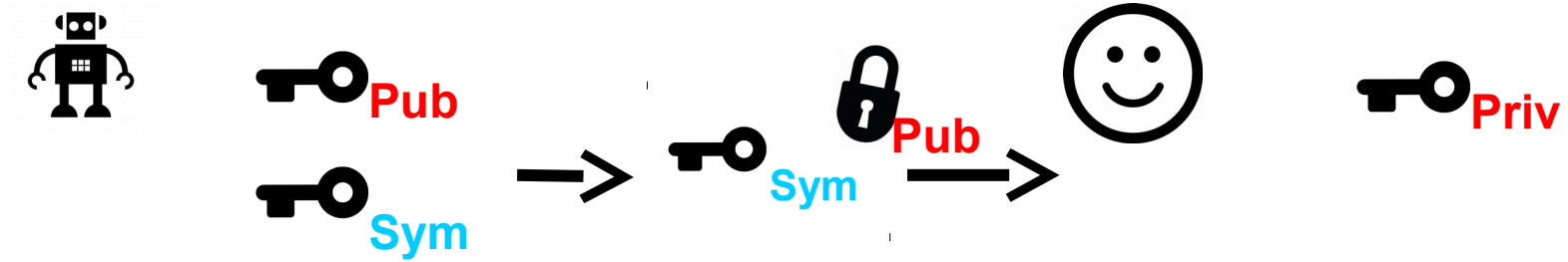
Our Approach

- Dynamically exchange encryption keys
- Automate retrieval and decryption of asymmetrically encrypted traffic.
- Proper enciphering/ deciphering of crafted replies to observe malware behaviour.

Circumventing Asymmetric Encryption

1. Locate and Replace Hardcoded Public Key

2. Redirect Bot Traffic to Fake C&C Server in Possession of Matching Private Key



3. Fake C&C Can Now Decrypt Bot Message and Send Crafted Responses

Fraunhofer FKIE

Implementation

Standard Cryptography Libraries

- A common way to analyze the behaviour of malware is by hooking critical API functions calls to collect the supported features

Dynamic Analysis

- API hooking is not a novel concept.
  - Sandboxes.
- Protocol and monitor API calls

API Hooks

General Idea

- Inline hooking of cryptographic functions to tamper with encryption key

Components

injector

1. create suspended
3. resume process

2. inject DLL

DLL

malware

5. contact C&C

fake C&C

6. send crafted response

Handlers

4. resolve C&C domains

DNS spoofer

Fake Server

- Collection of malware family handlers
  - Bot Interpreter
- Requirement
  - Prior knowledge of message structure and field semantics
- Protocol Tracker
  - If translator fails → modification of C&C protocol

DLL

- Account for spawned processes
  - Recursively inject DLL
- Extract and replace cryptographic keys
  - Replacement through pointer swapping

A Tête-a-Tête with Panda Banker

Panda Banker – Fake Server

```
Translator = Panda
Starting Secure server on port 4443 use <Ctrl-C> to stop
Starting Secure server on port 443 use <Ctrl-C> to stop
Starting server on port 8080 use <Ctrl-C> to stop
Starting server on port 80 use <Ctrl-C> to stop
('Accept: */*\r\n', '/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')
('Cache-Control: no-cache\r\n', '/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')
('User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET4.0C; .NET4.0E;
.NET CLR 2.0.50727; .NET CLR 3.0.04506.648; .NET CLR
3.5.21022)\r\n','/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')
('Host: secpressnetwork.com\r\n', '/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')
('Content-Length: 528\r\n', '/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')
('Connection: Close\r\n', '/MA4m5Or/46/CE7F1A/112C1/C76D2/94A6/1011A/77/370/')


decryption key = 2048-bit RSA private key
RSA decrypted AES-256 key:
51a95d8b7c3fe8191c09cac36865a7186d7b11b78fb45f44623c76ae4f84a88a
AES-256 CBC decrypted payload:
{
"BotInfo":      {
"systime":      1480351680,
"process":      "svchost.exe",
"user": "HuggableTeddy\\huggable_teddy",
"id":   "0B8F4B7724636E3B93A0CB255ED53E31",
"botnet":       "botnet-static",
"version":      "2.1.3"
},
"File": {
"name": "config.dat"
}
}
```

Panda Banker – DLL Trace

```
272:    DLL Init
272:    hook executed  NtCreateProcessEx
3872:   DLL Init
272:    hook executed  NtResumeThread
3872:   hook executed  NtCreateProcessEx
800:    DLL Init
3872:   hook executed  NtResumeThread
3872:   hook executed  NtCreateProcessEx
1508:   DLL Init
3872:   hook executed  NtResumeThread
272:    hook executed  NtCreateProcessEx
3872:   DLL Cleanup
2148:   DLL Init
272:    hook executed  NtResumeThread
272:    DLL Cleanup
2148:   DLL Cleanup
800:    hook executed        CryptImportKey       key pointer=0x002DF678
800:    key type=RSA public
800:    new key
800:    hook executed        CryptEncrypt  data location=0x001ED928
800:    key type=RSA public
plaintext (hex):             51a95d8b7c3fe8191c09cac36865a7186d7b11b78fb45f44623c76ae4f84a88a
ciphertext (hex):       e595fbf29ba23465564701fc35f772a4ddbebbc67e8ccd5d11644ad8566361a8673ac53724ccdd827cb
b75aaa2de4ef0c41c26d5b75ca1dc397ba203b0e648bf9d60a09b7550758680a9bc06cc6ef8b8d4085a131b477ed4a7ad41fc6
020a131a3578a71814d3b24bf132e3dc7415abded176cdf81b52556219e6dcbfec862a5d0cc68a219942268c11a4859a2e4bd9
d4e5db0c41e7530649718d97c1fb11ac125e1271fd5e197586c3b155a0d9d527d74e831c901e47e6ffaaa787b81638769f914cb
92f6989868b4b184785398aa6b9e93beaa99adfaf8b62c4ce73b4d52360c4dd56f13617494fa700986b677dc7305f153ba4e056
c84863780cb93f18cc2
```

A Tête-a-Tête with URLZone

URLZone – Fake Server

```
Translator = URLZone
Starting Secure server on port 4443 use <Ctrl-C> to stop
Starting Secure server on port 443 use <Ctrl-C> to stop
Starting server on port 8080 use <Ctrl-C> to stop
Starting server on port 80 use <Ctrl-C> to stop
User-Agent: Microsoft-CryptoAPI/6.1
/itc/
Host:ox2ybk1nf4muo3.net
/itc/
Content-Length: 128
/itc/
Cache-Control: no-cache
/itc/
decryption key = 1024-bit RSA private key
{'Payload': {'Keys': '31364638453741304639333242304433', 'Bot SHID':
'2cf1265fea5fdcbdef64f7dd46d8', 'Command': '2', 'Something': '9ca099e2c7f72c62', 'Sequence':
'0001', 'Bot ID': '1406a60890d50d5a6b24db596325b60292b5', 'Timestamp': datetime.datetime(2016,
11, 29, 2, 54, 38), 'Infected process': 'Explorer 6.0.2900.5634', 'OS version': '5.1'}, 'Family':
'URLZone'}


**************************************************
```

URLZone – DLL Trace

```
3264:   DLL Init
3264:   hook executed  NtCreateProcessEx
3316:   DLL Init
3264:   hook executed  NtResumeThread
3264:   DLL Cleanup
3316:   hook executed  NtCreateProcessEx
3164:   DLL Init
3316:   hook executed  NtResumeThread
3164:   hook executed  NtOpenProcess          PID=1552        Access=0x0000043A
1552:   DLL Init
3164:   hook executed  NtAllocateVirtualMemory
3164    allocated      0x01A10000 - 0x01A33000 in PID=1552
3164:   hook executed  NtAllocateVirtualMemory
3164    allocated      0x01A40000 - 0x01A41000 in PID=1552
3164:   hook executed  NtAllocateVirtualMemory
3164    allocated      0x02F10000 - 0x02F50000 in PID=1552
3164:   hook executed  NtAllocateVirtualMemory
3164    allocated      0x02F41000 - 0x02F50000 in PID=1552
3164:   DLL Cleanup
3316:   DLL Cleanup
1552:   hook executed  CryptImportKeykey pointer=0x02F4FC9C
1552:   key type=AES
1552:   hook executed  CryptImportKeykey pointer=0x01A129DE
1552:   key type=RSA public
1552:   new key
1552:   hook executed  CryptEncrypt    data location=0x02F4EB48
1552:   key type=RSA public
```

Limitations & Future Work

Limitations

- Dynamic analysis
  - dormant behavior
  - malware may anticipate DLL injection
- Function hooks
  - Relies on the malware's use of the Windows CryptoAPI

Future Work

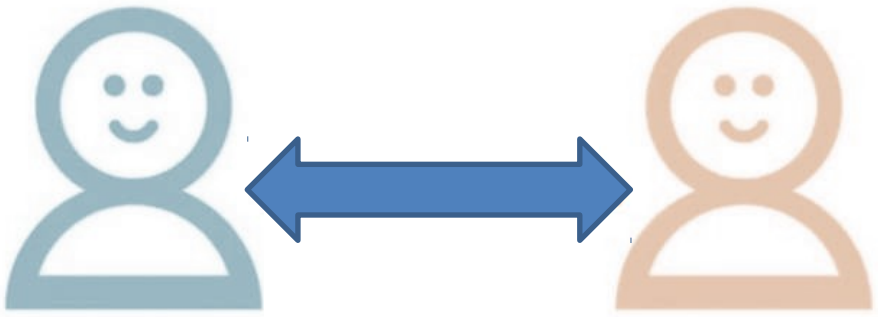
- Hooking on request
- Extend translators to allow crafted C&C responses
- Incorporation of educated guesses
  - Predictable occurrence of cryptographic API functions

Conclusion

Conclusion

- Proof-of-concept for recovering plaintext from asymmetrically encrypted botnet communication
- Expandable to allow crafted commands for behavioral analysis

“Sharing is caring“



jens.friess@stud.tu-darmstadt.de