S21sec labs

Tomorrow's Digital Security, Today

*[Bug hunting]

Jose Miguel Esparza

7th November 2007

Pamplona

AGENDA

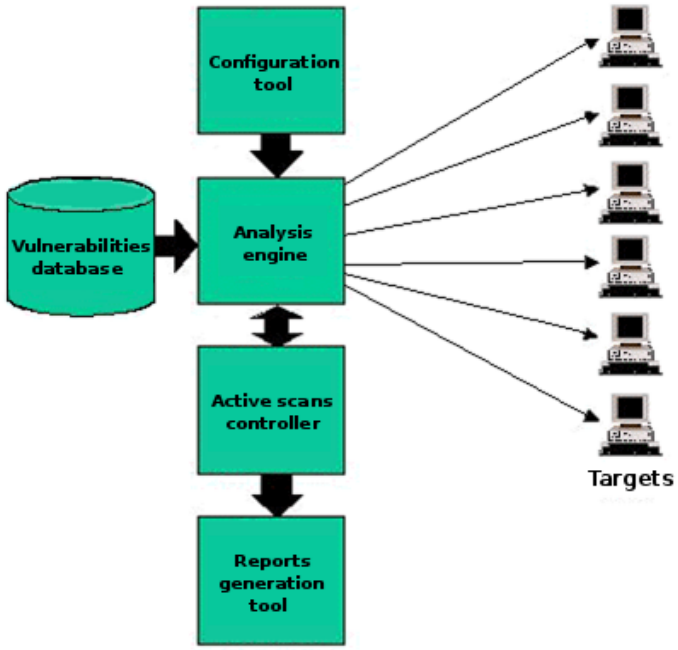
- Finding holes
- Fuzzing
  - What is this?
  - How to obtain data?
  - Phases
  - Tools
  - Pros and cons
- Malybuzz
  - What is this?
  - Protocol specifications
  - Differentiation
  - Future work

FINDING HOLES

- Manual analysis
  - Low false positives and negatives number
  - Based on experience
  - Annoying
- Automatic analysis
  - Vulnerability scanners
  - Source code scanners
  - Fuzzers

FINDING HOLES

- Vulnerability scanners



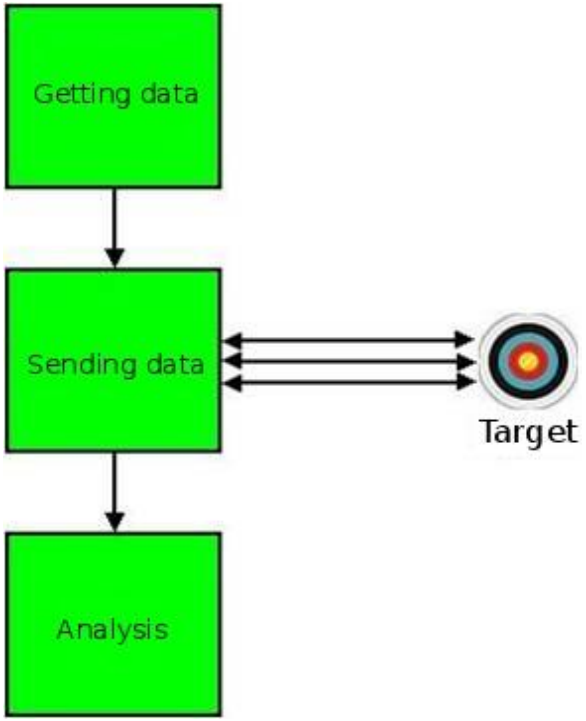
  - Known vulnerabilities

FINDING HOLES

- Source code scanners
  - Dangerous functions list
    - Buffer overflow: *gets(), scanf(), sprintf(), strcat(), strcpy()*
    - Format string: *printf(), fprintf(), vprintf(),snprintf(), vsnprintf()*
    - Race condition: *access(), chown(), chgrp(), mktemp(), tmpfile()*
  - Effectivity depending on complexity
    - Simple analysis
    - Lexical analysis
    - Semantic analysis
  - Source code dependent

FUZZING

- What is this?
  - Sending unexpected data to applications inputs
    - Pseudo random data
    - Valid but malformed data
  - Phases
    - Obtaining data to send
    - Sending data
    - Analysing target behaviour
    - Analysing bug?



FUZZING

- How to obtain data?
  - Mutation (dumb fuzzing)
    - Modification of existing valid inputs
  - Generation (intelligent fuzzing)
    - Make new data from file or protocol descriptions and modify
    - Better results
    - Recursion
      - Permutation
      - Repetition
    - Substitution
      - Static or generated lists depending on vulnerabilities: format strings, integer overflow, special strings, bad paths…

FUZZING

- How to obtain data?
  - Mutation (dumb fuzzing)
    - Modification of existing valid inputs

```
GET http://www.google.es/index.html HTTP/1.0
```

```
GET httxwprkmvjh92pwz12k?.google.es/index.html HTTP/1.0
```

FUZZING

- How to obtain data?
  - Generation
    - Recursion
      - Permutation

```
GET http://www.google.es/8302FA HTTP/1.0
```

```
GET http://www.google.es/000000 HTTP/1.0
…
GET http://www.google.es/45FD2A HTTP/1.0
…
GET http://www.google.es/FFFFFF HTTP/1.0
```

FUZZING

- How to obtain data?
  - Generation
    - Recursion
      - Repetition ⟶ Overflow!!

```
GET http://www.google.es/8302FA HTTP/1.0
```

```
GET A*512 HTTP/1.0
…
GET A*1023 HTTP/1.0
GET A*1024 HTTP/1.0
GET A*1025 HTTP/1.0
```

FUZZING

- How to obtain data?
  - Generation
    - Substitution
      - Fuzzing vectors

```
POST /Login.asp?validar=2 HTTP/1.1
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT
5.0)
Host: www.ejemplo.com
Content-Length: 71
Connection: Keep-Alive
```

```
POST /Login.asp?validar=2 HTTP/1.1
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT
5.0)
Host: www.ejemplo.com
Content-Length: 65536
Connection: Keep-Alive
```

FUZZING

- Sending data
  - Network requests
  - Inter-Process Communication
  - Executing by command line
- Analysing behaviour
  - Graphical signs
  - Debuggers
  - Network responses (or absence of them)
- Analysing bug
  - Exploitable?
  - How?
  - Versions affected?

FUZZING

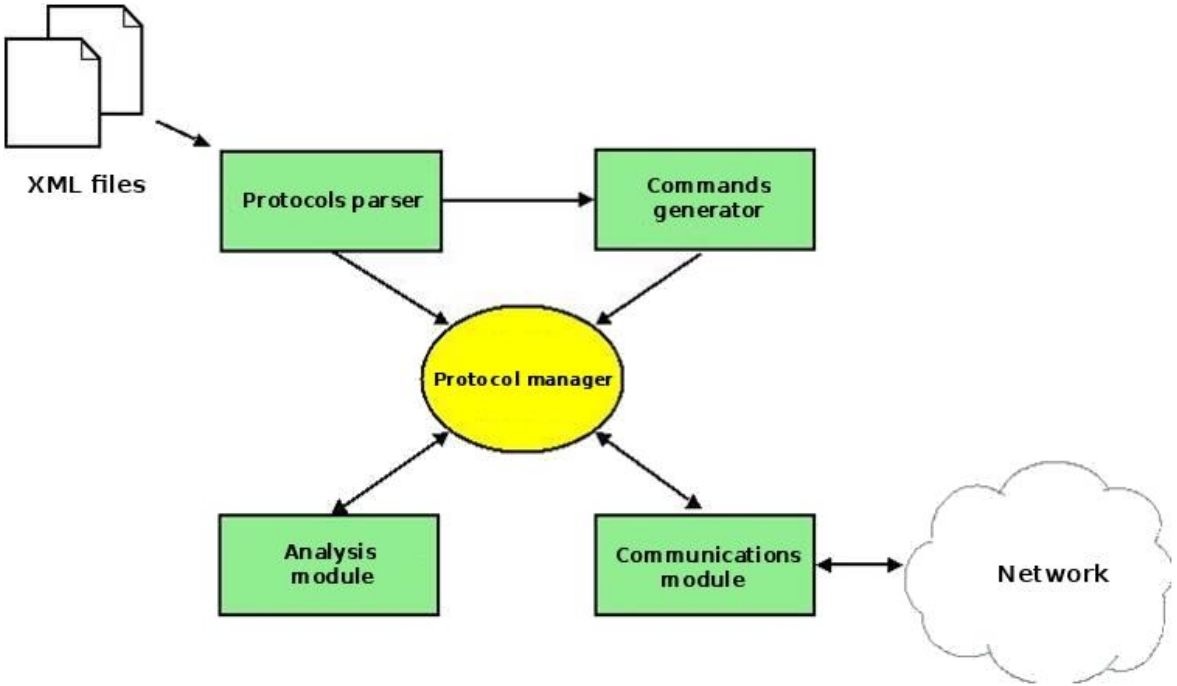
- Tools
  - Fuzzing frameworks
    - Generics
    - Functions for generation and sending data
    - Peach, SPIKE, Antiparser
  - Fuzzers
    - Normally specific tools
    - Oriented to
      - Network protocols: PROTOS, JBroFuzz, EFS, Taof, Malybuzz;)
      - File formats: Filefuzz, Ufuz3
      - COM objects: Axman, COMRaider
      - File systems: Fsfuzzer
    - The future?
      - Learn protocols or specifications from samples: EFS

FUZZING

- Pros
  - No need of source code
  - Good discovering **new** vulnerabilities
  - Faster finding superficial bugs
- Cons
  - Execution time
  - Slower or ineffective with deep bugs
  - Need some kind of debugging

MALYBUZZ

- Python tool
- Multiprotocol network fuzzer
- Intelligent fuzzing
- Session management



MALYBUZZ

- Protocol specifications
  - XML files ← RFCs
    - Commands
      - What commands?
      - What arguments?
      - What fields?
      - Where to fuzz?
    - Responses
      - Different responses for each command
      - What command to send when receiving X command?

MALYBUZZ

- Protocol specifications
  - Commands file

```
<protocol>
<port>5060</port>
<commands>
  <end-line>CRLF</end-line>
  <end-command>CRLF</end-command>

  <!-- Minimum commands -->
  <com send="yes" id="1">
    <com-name type="text">CANCEL </com-name>
  </com>
  <com send="yes" id="3">
    <com-name type="text">INVITE </com-name>
  </com>

  <!-- Global arguments (in all the commands) -->

  <global-arg type="text" send="yes" command="1,2,3,4,5,6" fuzzing="overflow,badString" fuzzCommand="3">sip:</global-arg>
  <global-arg name="URI_USER" type="text" send="yes" command="1,2,3,4,5,6" fuzzing="overflow,badString"
fuzzCommand="3">user@</global-arg>
  <global-arg name="URI_HOST" type="text" send="yes" command="1,2,3,4,5,6" fuzzing="overflow,badString,badHost"
fuzzCommand="3">$V:REMOTE_ADDRESS$ </global-arg>
  <global-arg name="PROTOCOL" type="text" send="yes" command="1,2,3,4,5,6">SIP/</global-arg>
  <global-arg name="PROTOCOL_VERSION" type="integer" send="yes" command="1,2,3,4,5,6" fuzzing="overflow,badString,badNumber"
fuzzCommand="3">2.0 </global-arg>
```

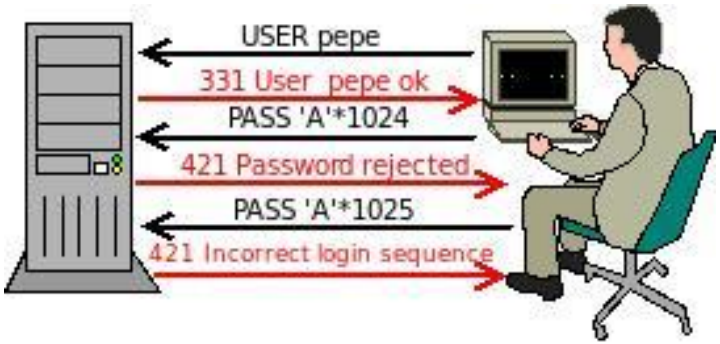
MALYBUZZ

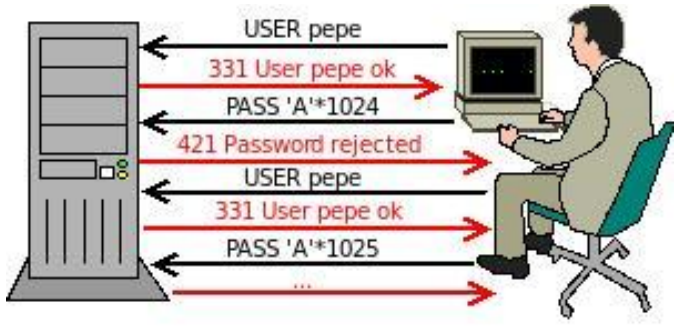
- Protocol specifications
  - Responses file

```
<protocol-states>
  <!-- Minimum commands -->
  <state id="1">
    <name>USER</name>
    <response-ok last="yes" next="PORT,PASV">230</response-ok>
    <response-err last="yes" next="PASS">331</response-err>
    <response-err last="yes" next="PASS">332</response-err>
    <response-err last="yes">421</response-err>
    <response-err last="yes">500</response-err>
    <response-err last="yes">501</response-err>
    <response-err last="yes" next="USER">530</response-err>
  </state>
  <state id="2">
    <name>PASS</name>
    <response-ok last="yes">202</response-ok>
    <response-ok last="yes" next="PORT,PASV">230</response-ok>
    <response-err last="yes" next="ACCT">332</response-err>
    <response-err last="yes" next="USER" action="wait">421</response-err>
    <response-err last="yes" next="USER">500</response-err>
    <response-err last="yes" next="USER">501</response-err>
    <response-err last="yes" next="USER">503</response-err>
    <response-err last="yes" next="USER">530</response-err>
  </state>
                                    ...
```

MALYBUZZ

- Differentiation
  - Generic and multiprotocol
    - Versus specific fuzzers
  - Full specification
    - Versus new fuzzers
  - Intelligence
    - Dynamic commands depending on responses
    - Hops in the protocol

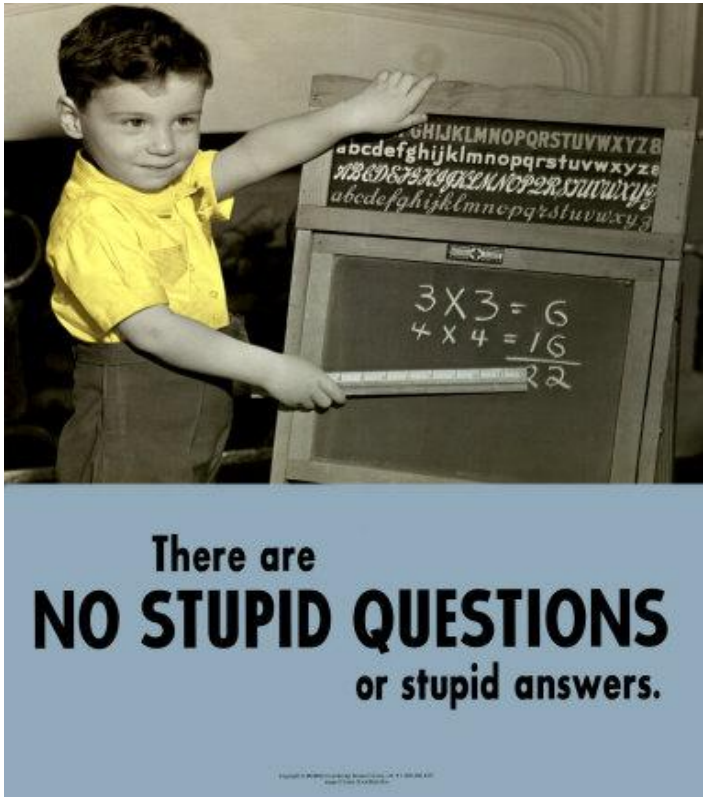




MALYBUZZ

- Future work
  - Simplify configuration
  - GUI
  - More text-based protocols: HTTP, SMTP, POP3...
  - Binary protocols: SMB, RPC...
  - Encrypted protocols: **SSH, SFTP, HTTPS...**
  - Improve fuzzing techniques
  - More intelligence
    - Explore paths in the protocol?
    - Unknown protocols?
  - ...
  - ...

QUESTIONS??



S21sec labs

Tomorrow's Digital Security, Today

*[Thank you very much]

AENOR ER Empresa Registrada UNE-EN ISO 9001 · IQNet · AENOR ER Gestión I+D+i UNE 166002 · AENOR ER Seguridad Información UNE 71502