KiF: A stateful SIP Fuzzer

Humberto J. Abdelnur
Humberto.Abdelnur@loria.fr

Radu State
Radu.State@loria.fr

Olivier Festor
Olivier.Festor@loria.fr

Madynes team
http://madynes.loria.fr
LORIA-INRIA Lorraine, France

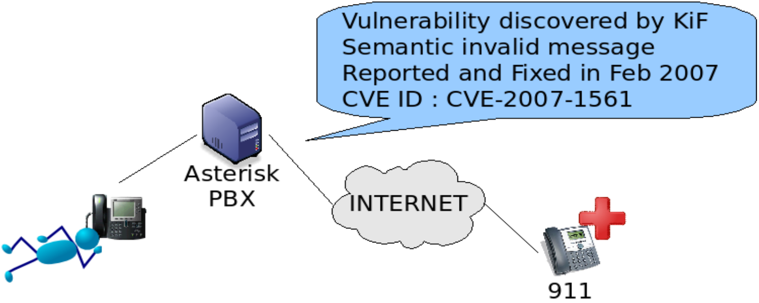
July 20, 2007

Why VoIP?

- VoIP network are becoming widely spread
- VoIP traffic is transported over Internet
  - Public network where access is granted to everyone
  - Exposes it to security threats (e.g. DoS, Evasdropping, Hijacking)
- Major signaling protocols are **SIP** and H.323
- No centralized smartness

Why Security?

DoS just sending one packet



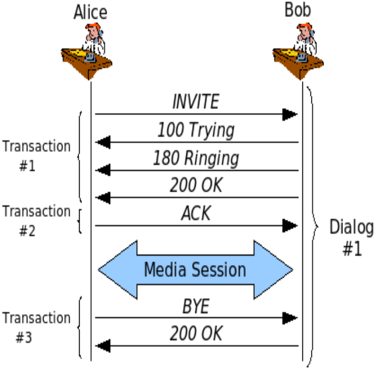
Vulnerability discovered by KiF
Semantic invalid message
Reported and Fixed in Feb 2007
CVE ID : CVE-2007-1561

“What if you are alone and dial 911 and no one answers?”. Die Hard 4

SIP Functional Hierarchy

SIP communication can be classified in:

- Dialogs:
  - Kept between 2 entities
  - Maintain a session state
- Transactions:
  - Define the handshake for each request
- Messages:
  - Individual data unit



The sequence of transactions defines the current state of the entity

Fuzzing

- Emerged as a branch of Software Testing
- Important topic for black box testing
- Based in input data validation
  - Random or invalid characters
  - Malicious data (e.g. string formatters)
- Functional verification is marginal
- Main objective is to find possible potential vulnerabilities

General limitations

- Limitates fuzzing to just a bunch of modifications
- Random data-base crafted generation only
- Hard to estimate what will be the generated output
- Hard to estimate the expected answer
- Success evaluation depends only in crashed or NOT-crashed
- Unavailable to test specific states of the target (i.e. stateless)
- Capitalized experience from the past is not considered

General limitations

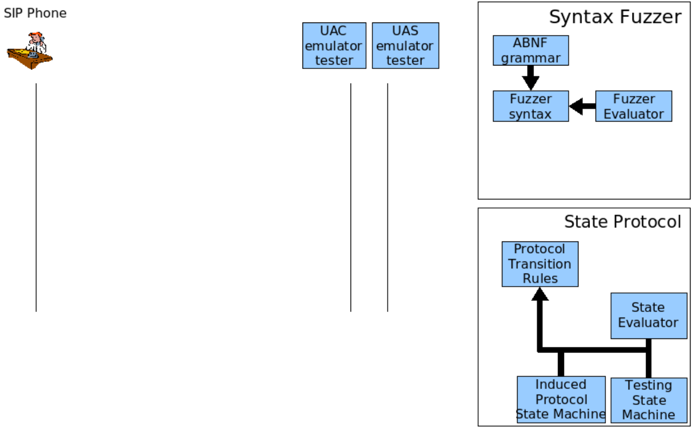
- Limitates fuzzing to just a bunch of modifications
- Random data-base crafted generation only
- Hard to estimate what will be the generated output
- Hard to estimate the expected answer
- Success evaluation depends only in crashed or NOT-crashed
- Unavailable to test specific states of the target (i.e. stateless)
- Capitalized experience from the past is not considered

Proposing solutions to these issues became our challenge

What to fuzz?

- Syntax fuzzing.
  - Invalid messages may reveal vulnerabilities
  - Consider which item of the message should be fuzzed
  - Headers or input values may be fuzzed
  - Think about which value should be the one to replace
  - The new value may or may not be syntactically correct
- Behavioral fuzzing
  - Unexpected messages may reveal vulnerabilities
  - Decide what type of message to send
  - Decide when to send the next message

KiF: General Framework



SIP Phone

UAC emulator tester

UAS emulator tester

Syntax Fuzzer

ABNF grammar

Fuzzer syntax

Fuzzer Evaluator

State Protocol

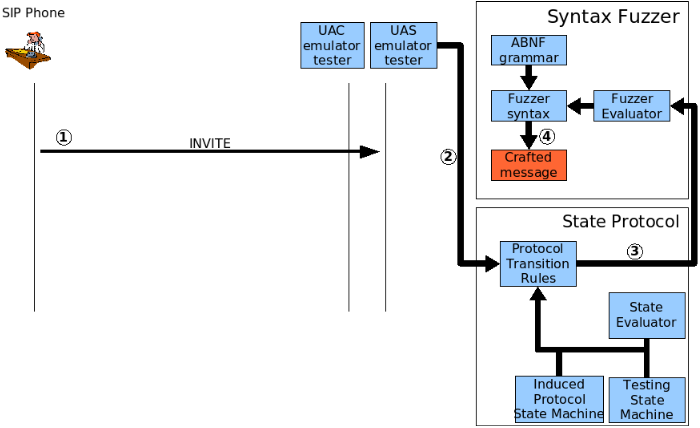
Protocol Transition Rules

State Evaluator

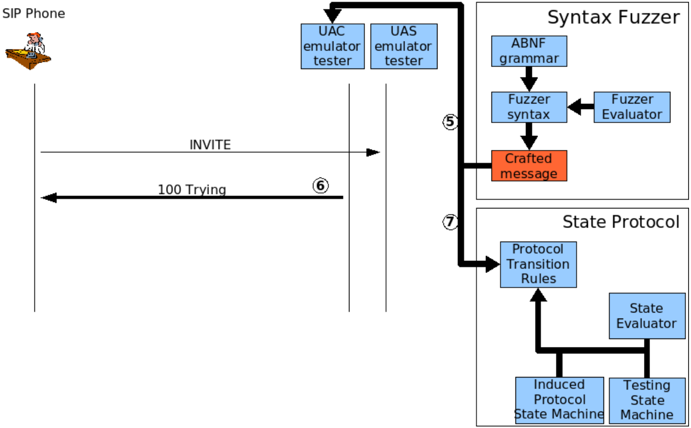
Induced Protocol State Machine

Testing State Machine

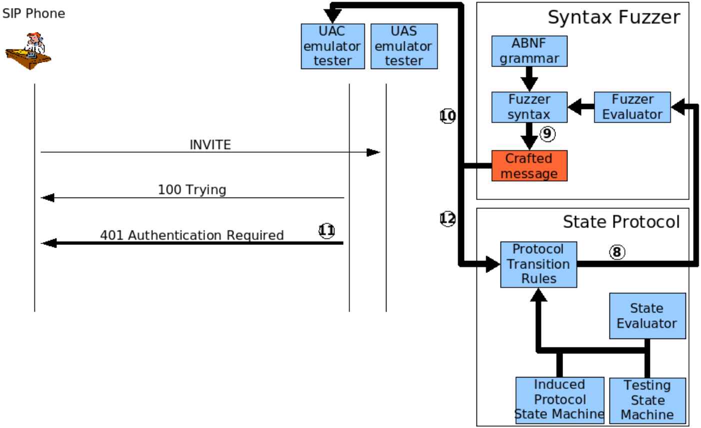
KiF: General Framework



KiF: General Framework



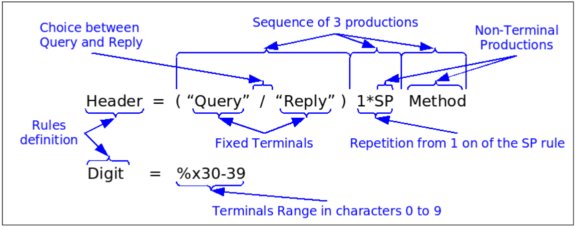
KiF: General Framework



An ABNF grammar

Grammar components:

- Σ - Terminals (e.g. “Querry”, “Reply”, %x30-39)
- N - Non-Terminals (e.g. Method, Header, Digit)
- e_1 .. e_n - Sequences
- $e_1/../e_n$ - Choices
- $e^{i,j}$ - Repetitions


Choice between Query and Reply
Sequence of 3 productions
Non-Terminal Productions
Header = (“Query” / “Reply”) 1*SP Method
Rules definition
Fixed Terminals
Repetition from 1 on of the SP rule
Digit = %x30-39
Terminals Range in characters 0 to 9

Note the e may be any of the Grammar items

Grammar inference

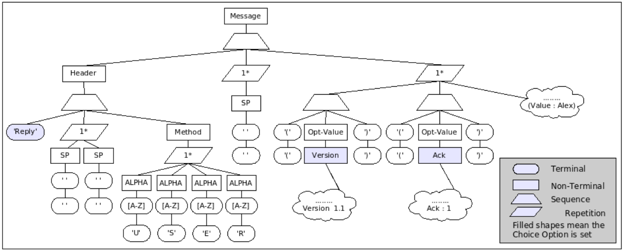
- Infer rules from a Context-Free Grammar (the use of an ABNF provides a complete knowledge of the messages syntax)
- Admits any grammar to create new fuzzers (i.e. genericity)
- Allows choosing the fields to fuzz (i.e. specificity to generate the crafted message)

Reply USER (Version 1.1)(Ack : 1)(Value : Alex)

(a) Example message compliant with the grammar show in (b)

```
Message       =     Header 1*SP 1*( "(" Opt-Value ")" )
Header        =     ("Query" / "Reply") 1*SP Method
Opt-Value      =    (Ack / Value / Version)
Method        =     1*ALPHA
Ack           =     "Ack" HCOLON 1*DIGIT
Value         =     "Value" HCOLON 1*ALPHA
Version       =     "Version" 1*SP DIGIT "." DIGIT
ALPHA         =     %x41-5A / %x61-7A          ; A-Z / a-z
DIGIT         =     %x30-39                    ; 0-9
HCOLON        =     *SP ":" *SP
SP            =     %x20                       ; space
```

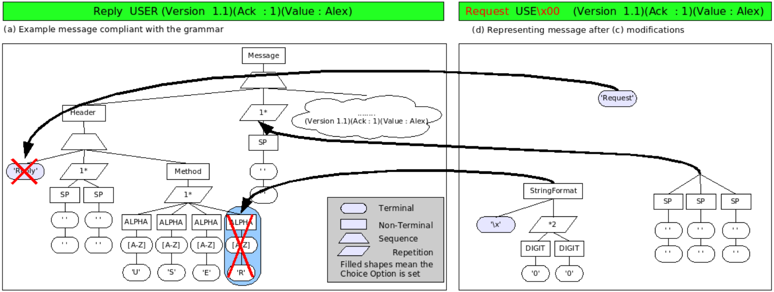
(b) ABNF Grammar (Toy Grammar)



(c) Infered structure from the Message in (a)

Syntax modifications

- Any existing reduction may be replaced (i.e. mutation or merging)
- Any grammar rule may be generated (i.e. generation from scratch)
- Statistic measures may influence the reduction of new rules (i.e. learning from the past)



Reply USER (Version 1.1)(Ack : 1)(Value : Alex)

(a) Example message compliant with the grammar

Request USE\x00 (Version 1.1)(Ack : 1)(Value : Alex)

(d) Representing message after (c) modifications

(b) Infered structure from the Message in (a)

(c) Structure modifications to the Message in (b)

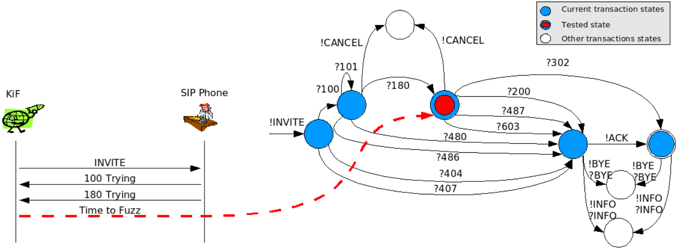
Fuzzer evaluator operations

5 operations were defined for replacing

1. Input a fixed string or randomly generated from a RegExp
2. Append a structure generated by another evaluator
3. Reduce from another rule defined
4. Reduce from a new rule defined on the fly
5. Generate a Function rule
    - Semantic purposes
    - Used for checksums, content lengths, etc.

Behavioral testing

- One induced state machine is used to supervise the testing
  - Deduces the normal behavior of the target entity



- Another state machine may be provided as the scenario
  - This will force the course of the testing

Reporting errors

- If the reply messages are syntactically incorrect
- The type of transition does not match any of the possible one from the induced State Machine
- When a message other than the expected one in the scenario occurs (i.e. when the scenario is trying to avoid the normal proceedings, e.g. for registering)
- And when the device is not responding anymore

Tested devices

All the 8 devices report vulnerabilities

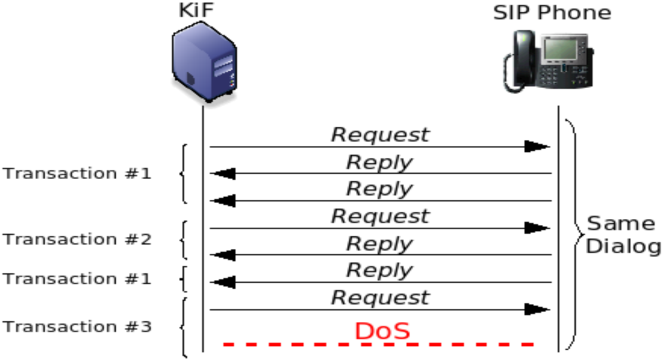
- Remote DoS **Asterisk** (PBX, SIP, H.323, PSTN, etc.)
- Tollfraud and DoS **Cisco Callmanager** 5.1
- Remote DoS **Cisco 7940**
- Remote DoS and auto-answering **GrandStream GXV-3000**
- Remote DoS **GrandStream BudgeTone 200**
- Remote DoS and String Overflows **Linksys SPA941**
- String Overflow **Thomson ST2020**
- Remote DoS **Thomson ST2030**

Thus, the vulnerabilities were related to:

- Just syntax fuzzing
- Others syntax fuzzing but state aware
- Some more were syntactically right but not corresponding to the current state

Cisco 7940 0-day Vulnerability

- DoS after sending 3 or either 10 messages
- All messages are SIP compliant
- Vulnerability reported in February 2007
- Fix release expected to be in August 2007



Future work

- Improve the learning capacity of the State Machine
- Measure the testing coverage
- Improve the evaluation of the impact of a message on the target
- Use Genetic Algorithms to improve the fuzzing for each devices
    - Some devices just forward the data, they do not interpret it
    - Some others are really strong for syntax validation
    - However, semantic issues can be found