Modular Network Trace Analysis

Wolfgang Kiess, Nadine Chmill, Ulrich Wittelsbürger, Martin Mauve

Computer Networks Research Group
Heinrich-Heine-University
Düsseldorf, Germany

27th October 2008
Overview

- Introduction
- Extensible data analysis toolkit (EDAT)
  - Philosophy
  - Caching
  - Executable Pieces of Code
- Demo
Introduction

• Our goal: evaluate (wireless multihop) networks in simulations and real-world experiments
• Results in a number of (packet) trace files
• Interpretation based on these files

<table>
<thead>
<tr>
<th>time</th>
<th>mac_src</th>
<th>mac_dst</th>
<th>ip_src</th>
<th>ip_dst</th>
<th>size</th>
<th>ip_hlen</th>
</tr>
</thead>
<tbody>
<tr>
<td>1194969596.6634</td>
<td>00022baf899c</td>
<td>ffffffffffff</td>
<td>192.168.5.55</td>
<td>192.168.5.51</td>
<td>95</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.66921</td>
<td>00013607670b</td>
<td>ffffffffffff</td>
<td>192.168.5.52</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.68486</td>
<td>0020e04d089a</td>
<td>ffffffffffff</td>
<td>192.168.5.51</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.70771</td>
<td>00013607670b</td>
<td>ffffffffffff</td>
<td>192.168.5.52</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.73287</td>
<td>0020e04d089a</td>
<td>ffffffffffff</td>
<td>192.168.5.51</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.75165</td>
<td>00013607670b</td>
<td>ffffffffffff</td>
<td>192.168.5.52</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.78087</td>
<td>0020e04d089a</td>
<td>ffffffffffff</td>
<td>192.168.5.51</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.79949</td>
<td>00013607670b</td>
<td>ffffffffffff</td>
<td>192.168.5.52</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.82488</td>
<td>0020e04d089a</td>
<td>ffffffffffff</td>
<td>192.168.5.51</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.84786</td>
<td>00013607670b</td>
<td>ffffffffffff</td>
<td>192.168.5.52</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>1194969596.86888</td>
<td>0020e04d089a</td>
<td>ffffffffffff</td>
<td>192.168.5.51</td>
<td>192.168.5.255</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>......</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Introduction

- Our goal: evaluate (wireless multihop) networks in simulations and real-world experiments
- Results in a number of (packet) trace files
- Interpretation based on these files
Often used approach

Some “quick hack” evaluation tools (created for simulations or real-world experiments):

- Paper for conference, new tool: 410 LOC (ruby)
- Paper for Elsevier journal, new tool: 305 LOC (C/C++)
- Master’s thesis, extension of existing tool: 1630 LOC (perl)
- Master’s thesis, new tool: 1220 LOC (ruby)

Observations:

1. Small/Medium amount of data (a few ten MB max)
2. Programming effort
3. Reusability?
Observations

In most programs, different operations occur repeatedly:

- Parsing data in one or multiple files
- Mangling/Processing
  - Selecting values
  - Building differences
  - Group similar items together and count them
  - ...
- Plotting
Observations

In most programs, different operations occur repeatedly:

- Parsing data in one or multiple files
- Mangling/Processing
  - Selecting values
  - Building differences
  - Group similar items together and count them
  - ...
- Plotting

Consequence: Make recurring components reusable
EDAT:
Extensible data analysis toolkit
Philosophy

1. Encapsulate recurring operations in an *operator*
2. Connect operators to a processing pipeline
Data format between pipeline elements

- Data is stored in generic container
- Contains a number of rows
- Row: associative array of key-value pairs
Processing in an Operator

- Get data container from previous operator
- Modify the data
- Return new container
Implementation

- An operator is a ruby class
- An analysis is a ruby script
- Each operator is instantiated and configured
- Requesting the result from an operator triggers the calculations

Example:

```ruby
... output_1 = PcapParser.new("Tcpdump_node51.cap")
output_2 = Filter.new(output_1, "size", "==100")
output_3 = CountLines.new(output_2)
Operator::showResult(output_3)
...```

Kiess, Chmill, Wittelsbürger, Mauve: Modular Network Trace Analysis
Graphical User Interface

Motivation:
- Writing the analysis scripts by hand takes too much time
- Graphically building the pipeline is more intuitive
Caching

Change parameter of one operator
Example: change filter from $x<20$ to $x<30$
Caching

Change parameter of one operator
Example: change filter from $x < 20$ to $x < 30$

Initial Analysis

Modified Analysis

Reusable

Adapt Parameter

$x < 20$

$x < 30$
Caching

What can be reused? How can this be determined?

- Operator has inputs: preceding operator and configuration
- When one of this inputs changes: Recalculate
- Easy for simple inputs like configuration parameters
- How about operators? How to know when their result changed?
Caching

What can be reused? How can this be determined?

- Operator has inputs: preceding operator and configuration
- When one of this inputs changes: Recalculate
- Easy for simple inputs like configuration parameters
- How about operators? How to know when their result changed?

Implementation: each operator has a fingerprint that changes with changing inputs
Caching - Fingerprint Calculation

- A fingerprint is a Hex string
- Each input is treated differently:
  - Operator: use its fingerprint
  - File: use modification time and filename
  - Parameter: use string representation
- MD5SUM over concatenation of these values is fingerprint
Executable Pieces of Code

Scripting language ruby: specify operations at runtime. Example: extract a packet identifier from the UDP payload of a packet

<table>
<thead>
<tr>
<th>IP</th>
<th>UDP</th>
<th>payload: packet ID</th>
</tr>
</thead>
</table>

- Take payload of IP packet (== UDP packet)
- Strip 8 byte UDP header
- Convert result to an integer
Conclusions

EDAT can be found under
http://www.cn.uni-duesseldorf.de/projects/EDAT

Questions?