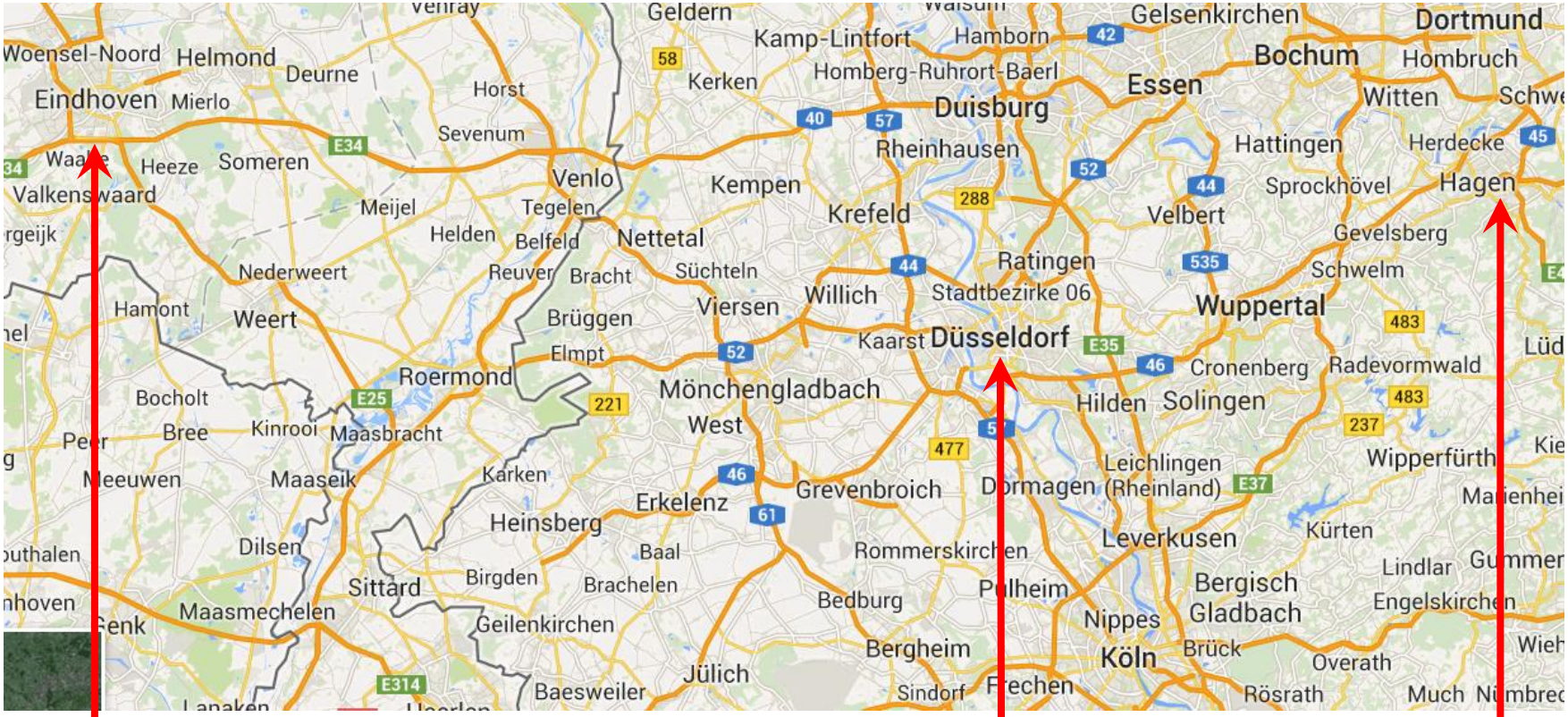
Analyzing a TCP/IP-Protocol with Process Mining Techniques

Christian Wakup
rubecon information technologies GmbH
Düsseldorf, Germany

Jörg Desel
FernUniversität in Hagen
Germany


Eindhoven
Düsseldorf
Hagen

the problem

- legacy software (client / server)
- proprietary protocols (ethernet)

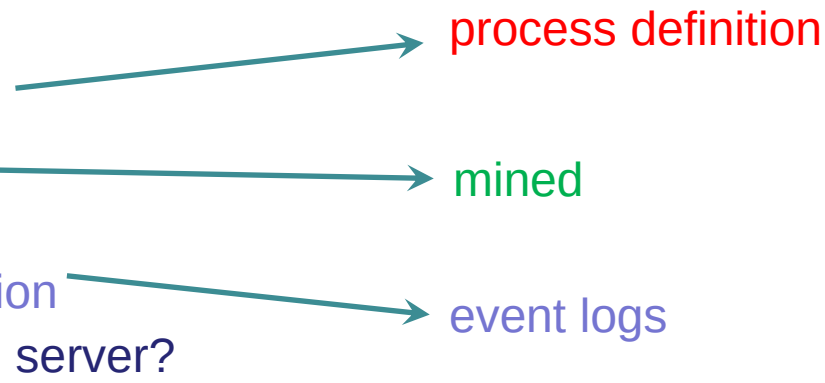
How can the
protocol definition
be reconstructed
from observations
of the communication
between client and server?

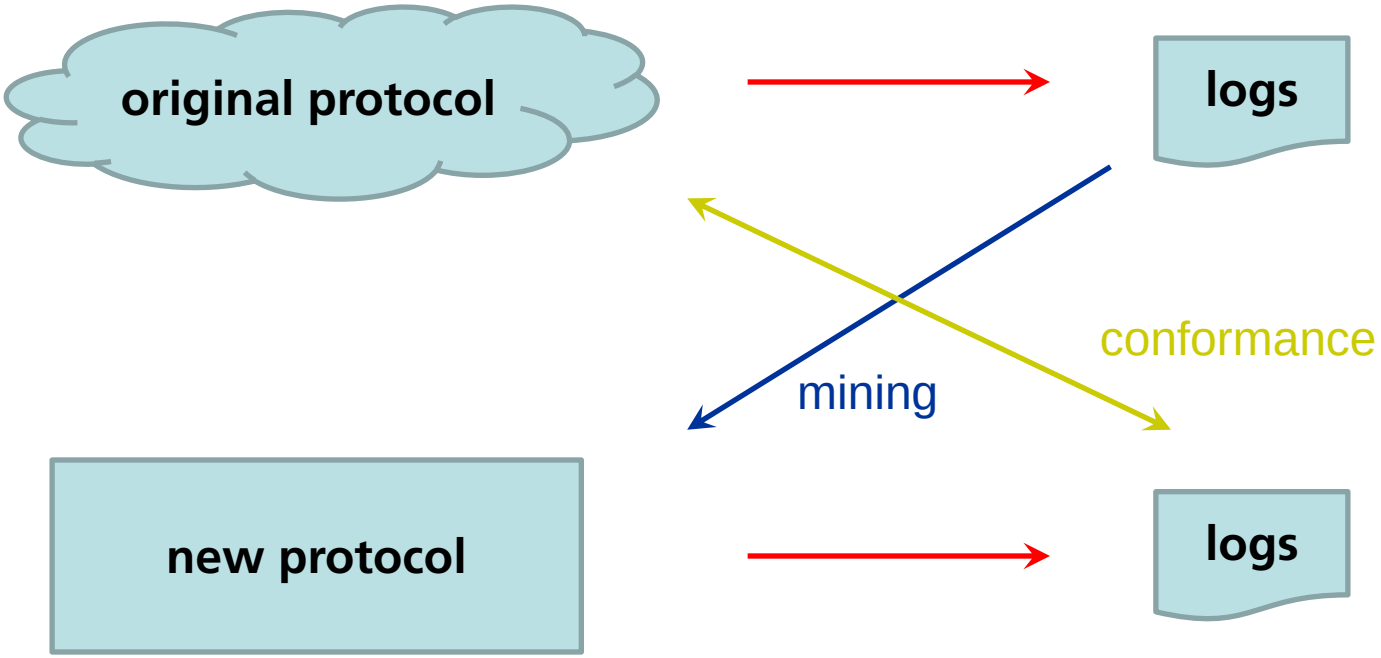
for correct substitution of a communication partner…

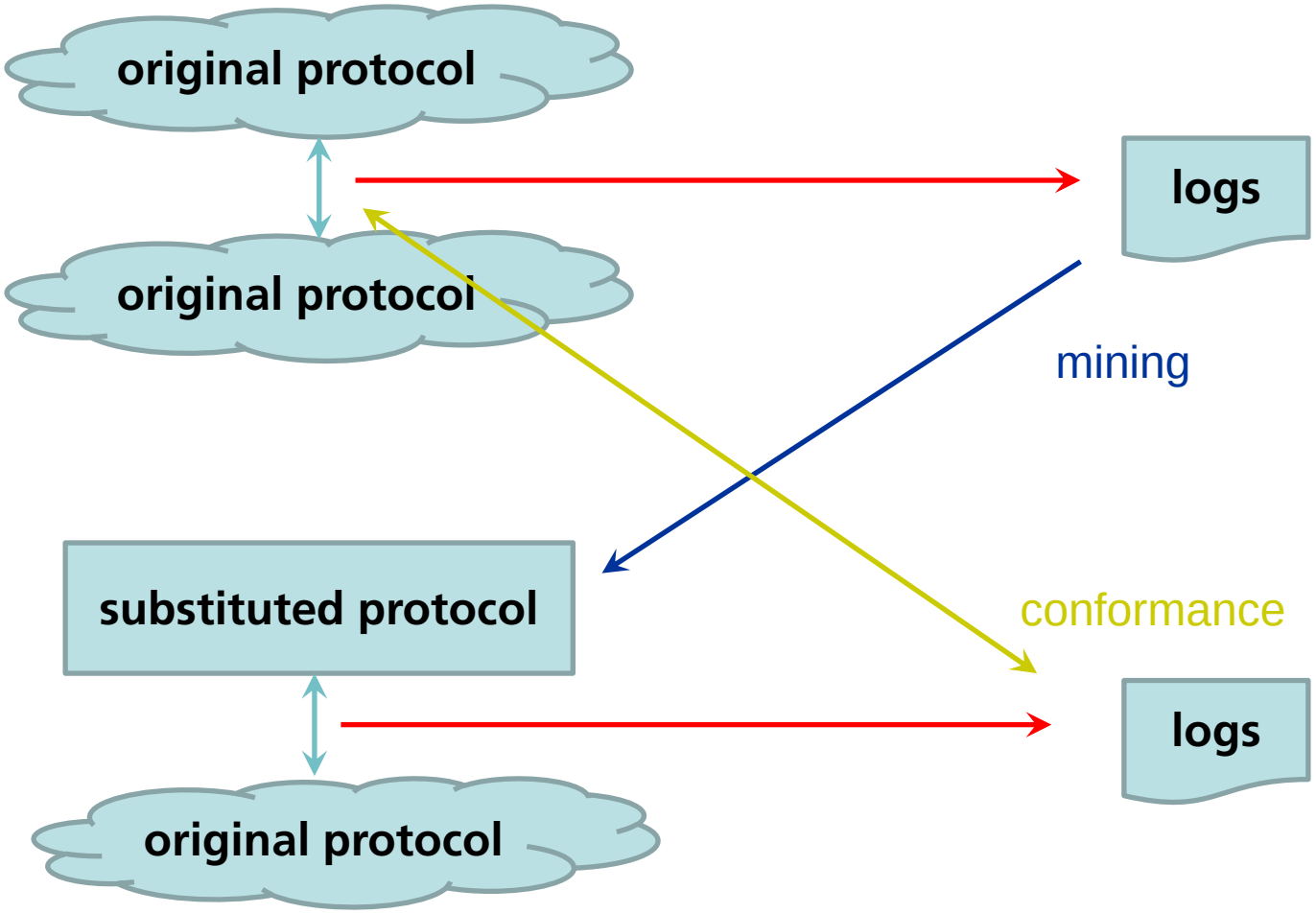
the problem

- legacy software (client / server)
- proprietary protocols (ethernet)

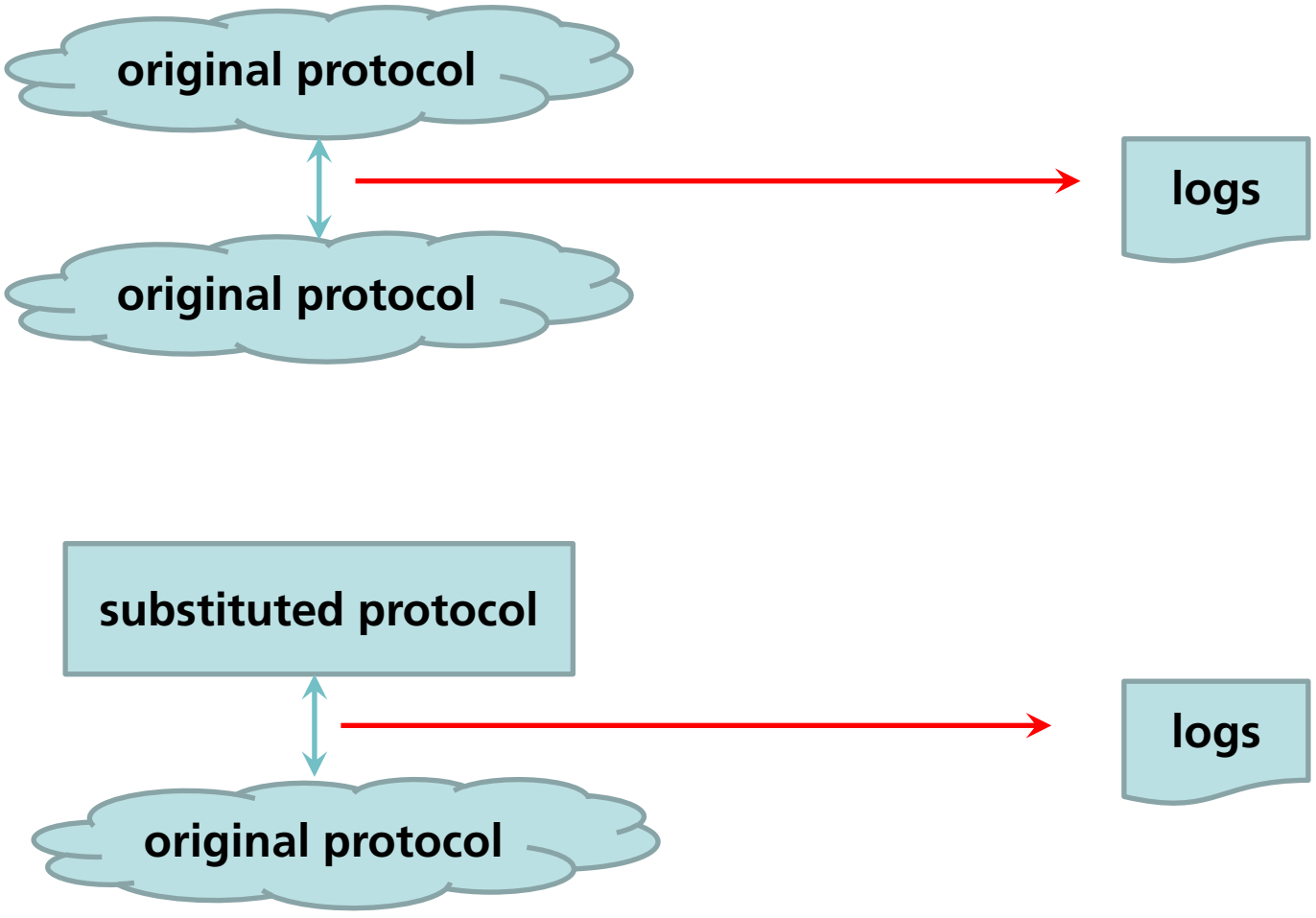
How can the
protocol definition
be reconstructed
from observations
of the communication
between client and server?




original protocol
logs
conformance
mining
new protocol
logs


original protocol
original protocol
logs
mining
substituted protocol
conformance
logs
original protocol

actual situation (case study)



outline

- first motivation
- outline
- requirements of event logs for mining
- from communication logs to event logs
- application of mining techniques
- derived processes
- a case study

requirements of data in event logs for mining

[Wil van der Aalst, Boudewijn van Dongen:
Discovering Petri nets from event logs, TOPNOC VII, 2013]

- each entry refers to a single process **case**
- each process case refers to a specific **process definition**
- each entry refers to an **event** of the case at **one point of time**
- each entry refers to a **single**, **uniquely identifiable** event
- each entry includes a **description** of that event

how do we get communication logs?

observe network traffic!

[Dustdar, Gombotz:
Discovering web service workflows using web service interaction mining,
Int. J. Business Process Integration and Management 2006]

proposes:

an HTTP-listener that filters out all relevant data traffic of the web server

But: we do not have HTTP here!

how do we get communication logs?

We developed the tool **TCPLog2EventLog:**

- **Input: TCP log from Wireshark**
- **Output: event log**

What can we get from wireshark?

- complete data exchange in a time interval
  filter: only data using a single port

- duplicate data (TCP sends lost packets again!)
  filter: rules out duplicates

- data concerning establishment and deletion of sessions
  filter: deletes such messages, according to specific flags

To be done by TCPLog2EventLog?

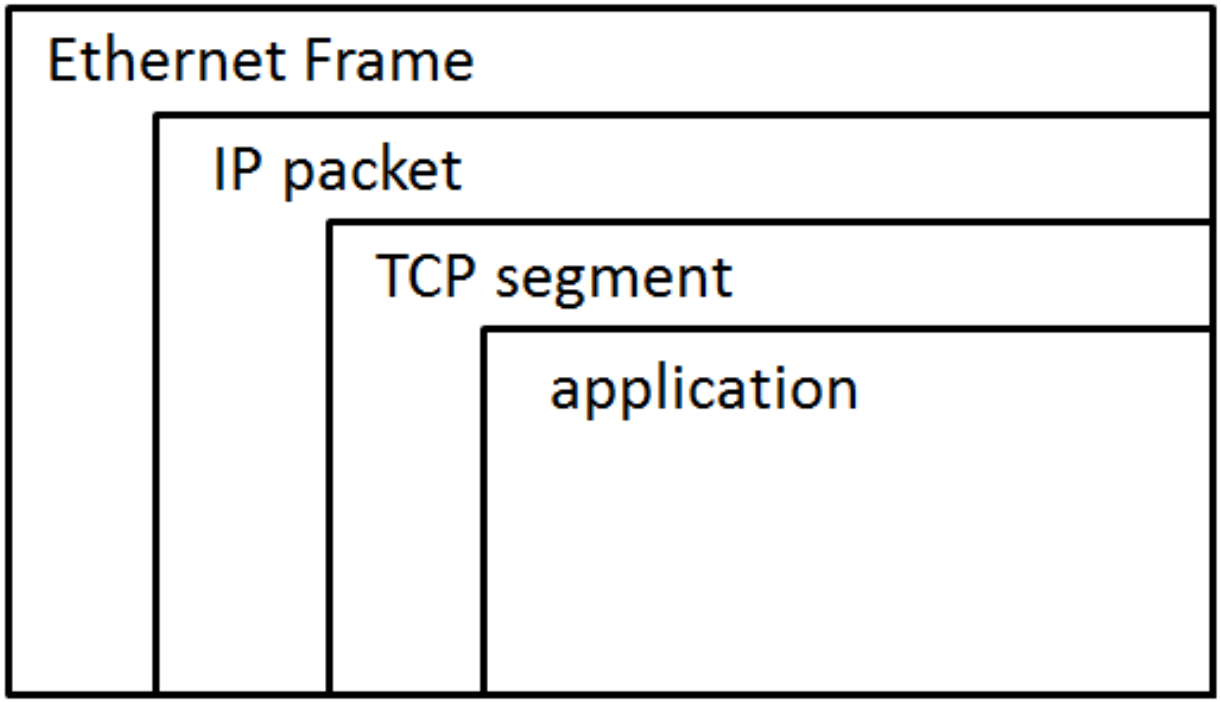
- (unreadable) hexadecimal data
  translation: readable ASCII

- time stamps missing
  add: recording time of messages

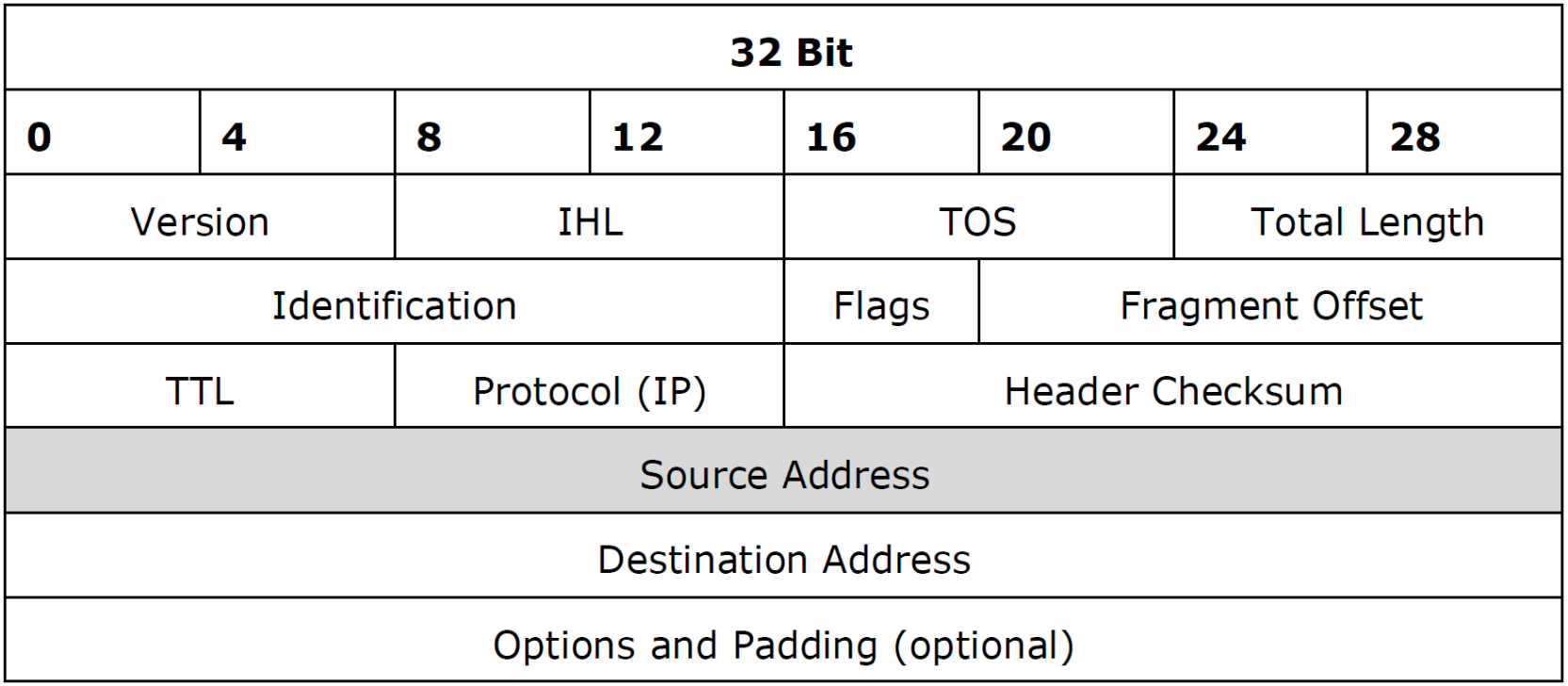
- cases missing
  calculate: from sequence numbers, ack numbers and data length: corresponding messages of a conversation

- resources missing
  add: source address of IP header

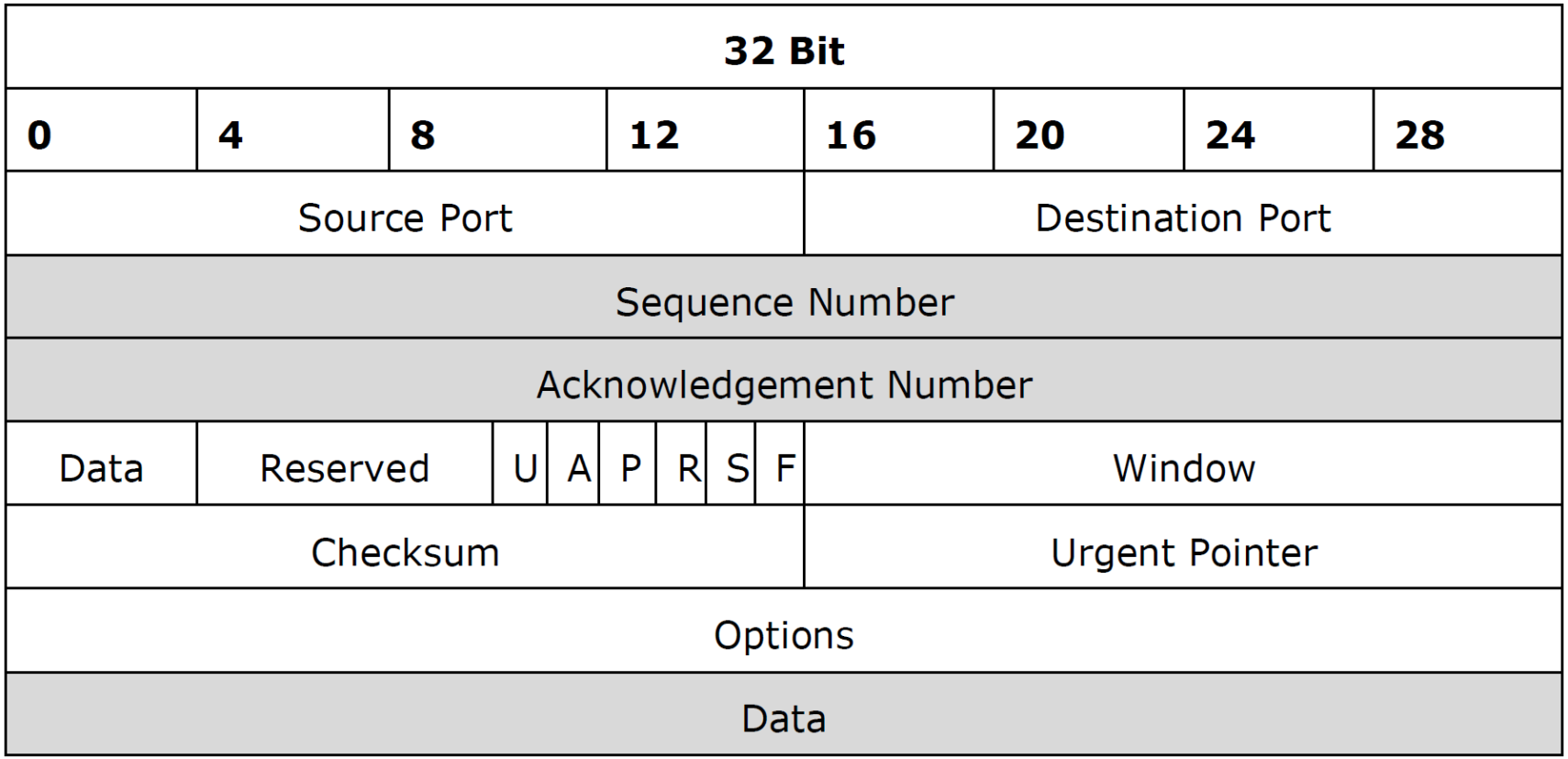
Internet packet



IPv4 header


32 Bit
0
4
8
12
16
20
24
28
Version
IHL
TOS
Total Length
Identification
Flags
Fragment Offset
TTL
Protocol (IP)
Header Checksum
Source Address
Destination Address
Options and Padding (optional)

TCP header


32 Bit
0
4
8
12
16
20
24
28
Source Port
Destination Port
Sequence Number
Acknowledgement Number
Data
Reserved
U
A
P
R
S
F
Window
Checksum
Urgent Pointer
Options
Data

To be done by TCPLog2EventLog?

still missing:

- activities (given a message, what activity does it represent?)

generate example message from single activities, yielding (unreadable) text in the TCP data field, which now is assigned a name

the same activity always generates similar text, therefore construct classes of similar activities (Levenshtein algorithm)

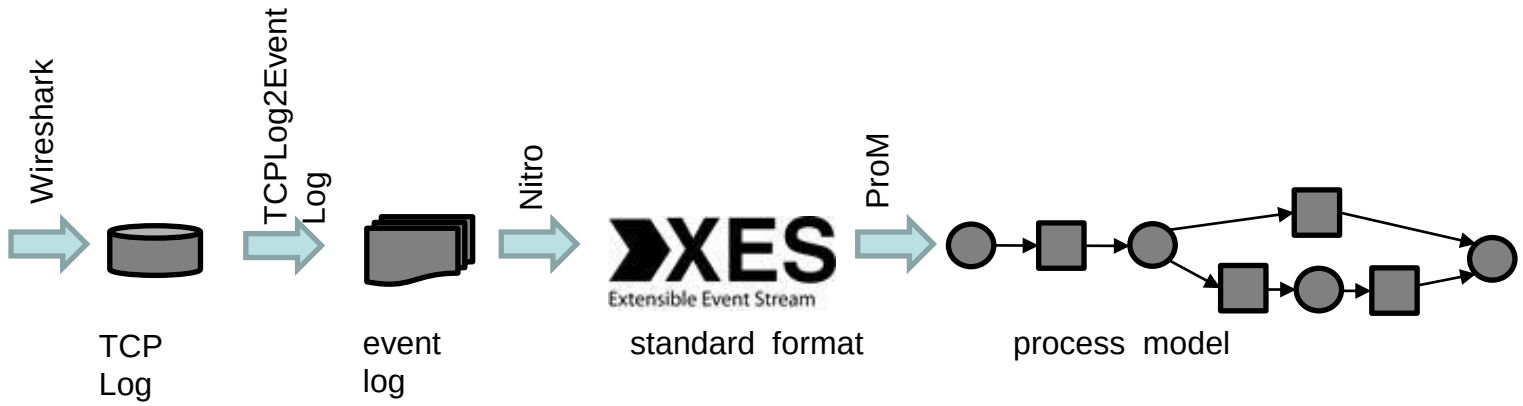
If an assignment is not possible, create a new activity <<number>>

Always add the prefix “server” or “client” (sender of the message)

The complete tool chain

Conversion of event logs to XES standard formal by Nitro (Fluxicon)

PROM (6.2) can read XES files to generate process models

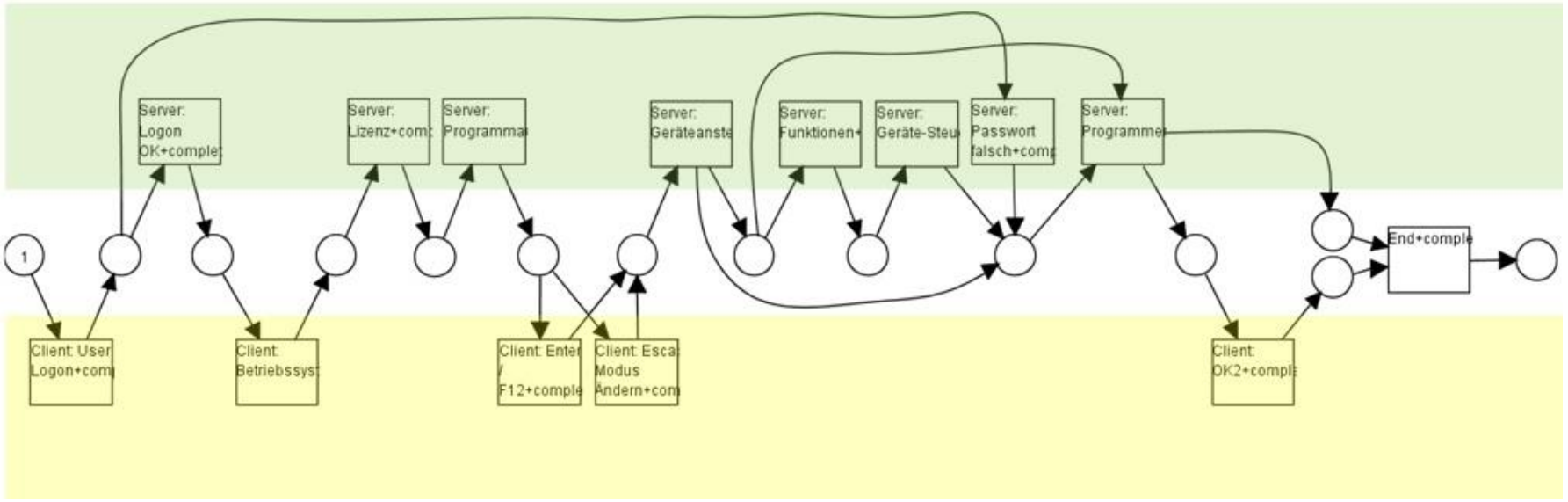


Mining of event logs

PROM (6.2):

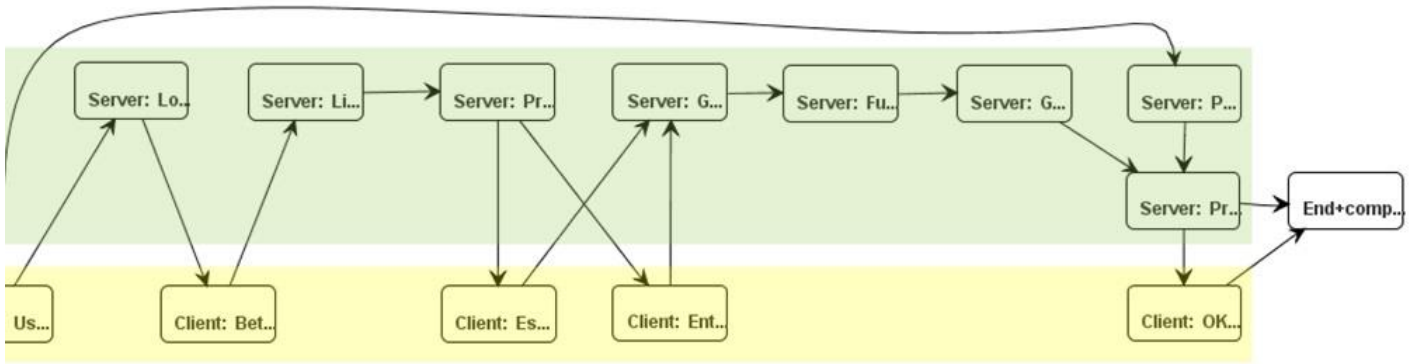
- Alpha Miner
- Heuristic Miner
- Genetic Miner

Small use case: Alpha Miner



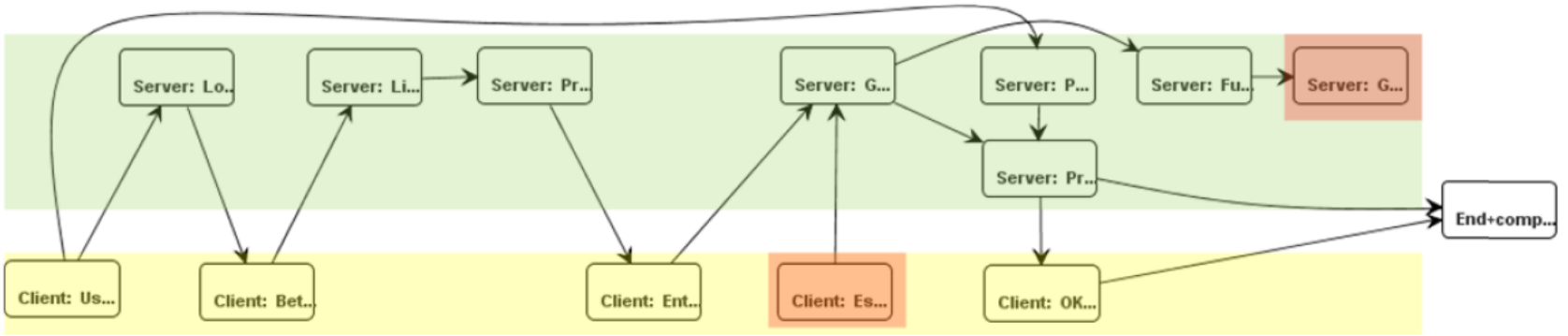
surprisingly: best result

Small use case: Heuristic Miner



also very good result

Small use case: Genetic Miner



not well suitable

Experiment

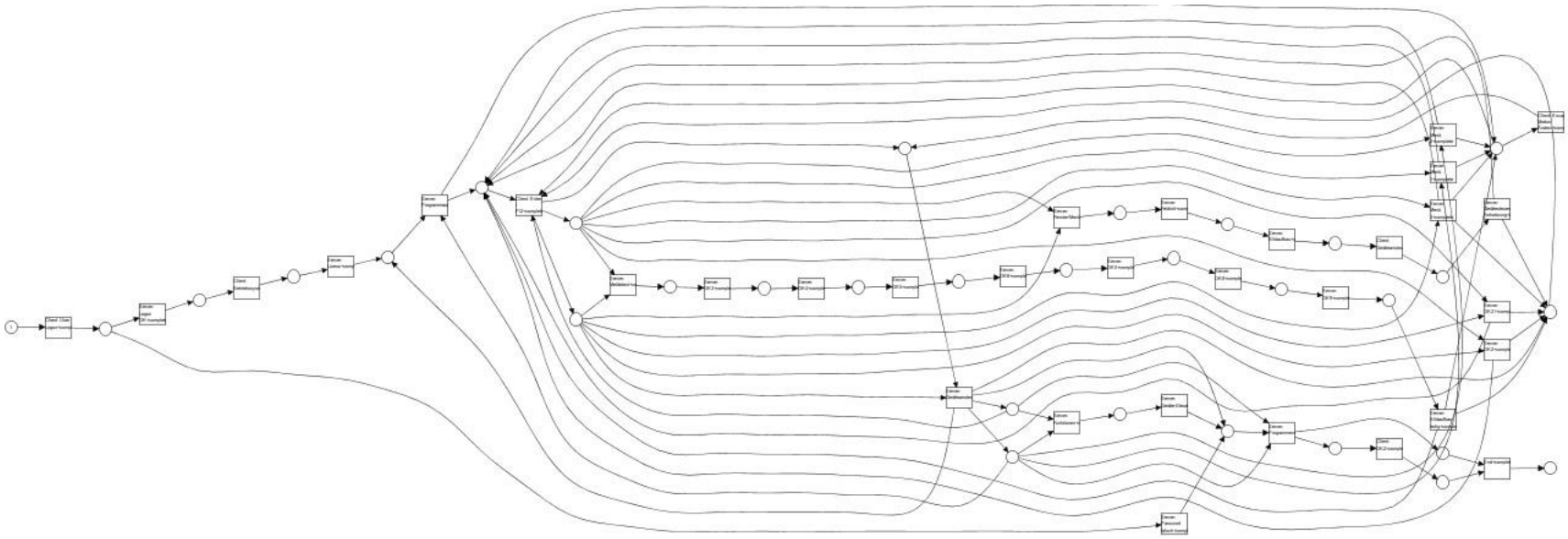
extend the small use case

establish connection / close connection

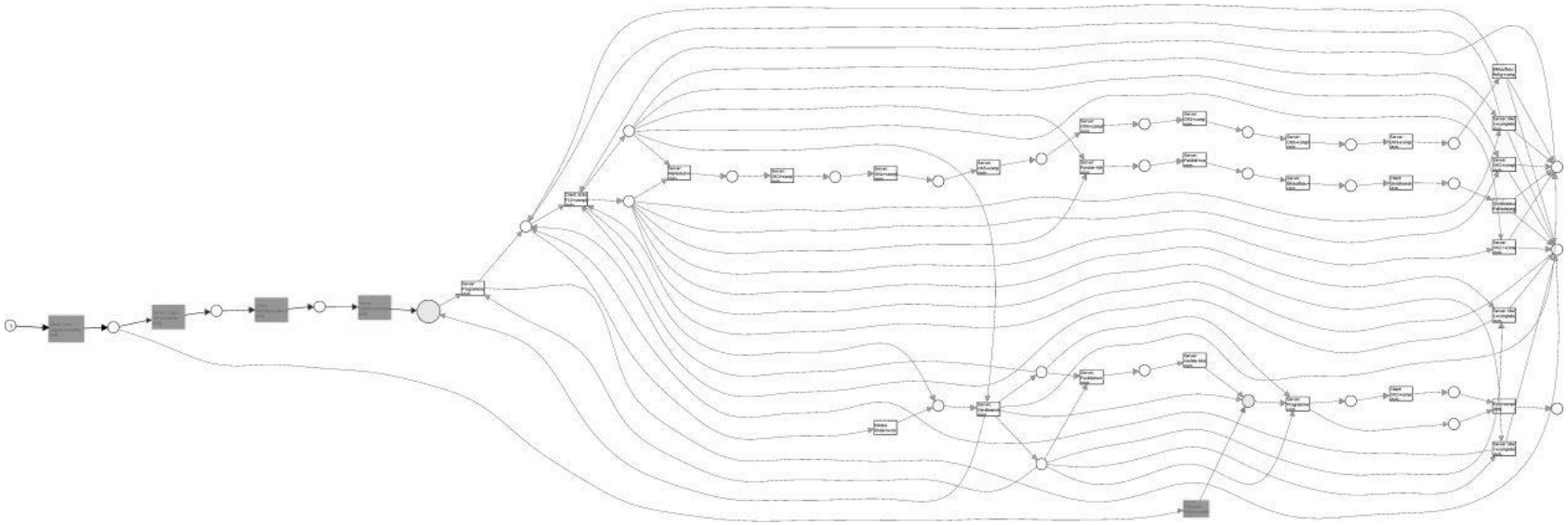
by some intermediate actions.

The runs of the small use case should be still possible!

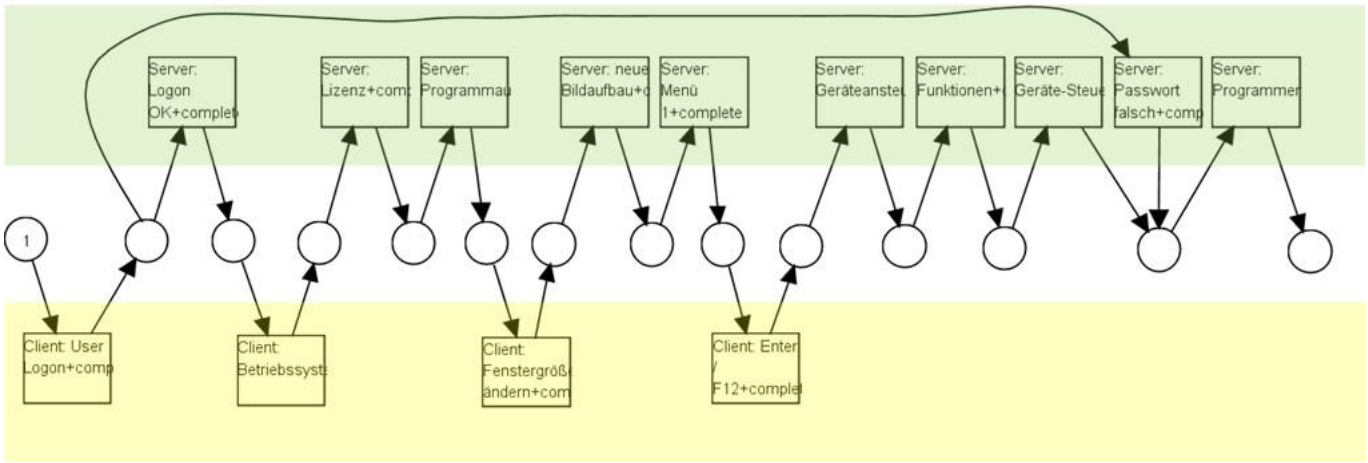
extended use case



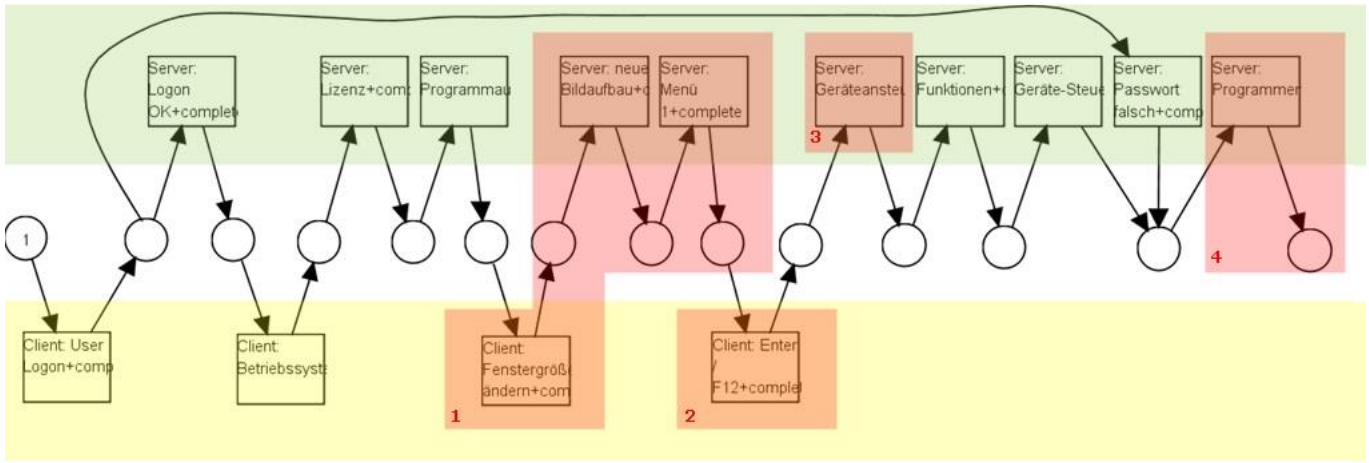
PROM: replay a log on Petri net for conformance analysis (small use case)



Case study: simple case study, new protocol



differences to old protocol



Results of the case study

The new protocol has substantial differences to the old one.

These differences are identified.

Most of them are harmless
(such as “window is opened” in new protocol vs.
“window can be opened” in old protocol)

Some differences are quite unexpected.

Experiences

Workflow mining is useful for protocol identification!

Identified problems:

- sometimes different activities in same class (unnecessary loops)
- sometimes same activity in different classes (isolated transitions)

repaired manually…

Analyzing a TCP/IP-Protocol with Process Mining Techniques

Christian Wakup
rubecon information technologies GmbH
Düsseldorf, Germany

Jörg Desel
FernUniversität in Hagen
Germany