Jacobs University Bremen

Cybermetrics: User Identification Through Network Flow Analysis

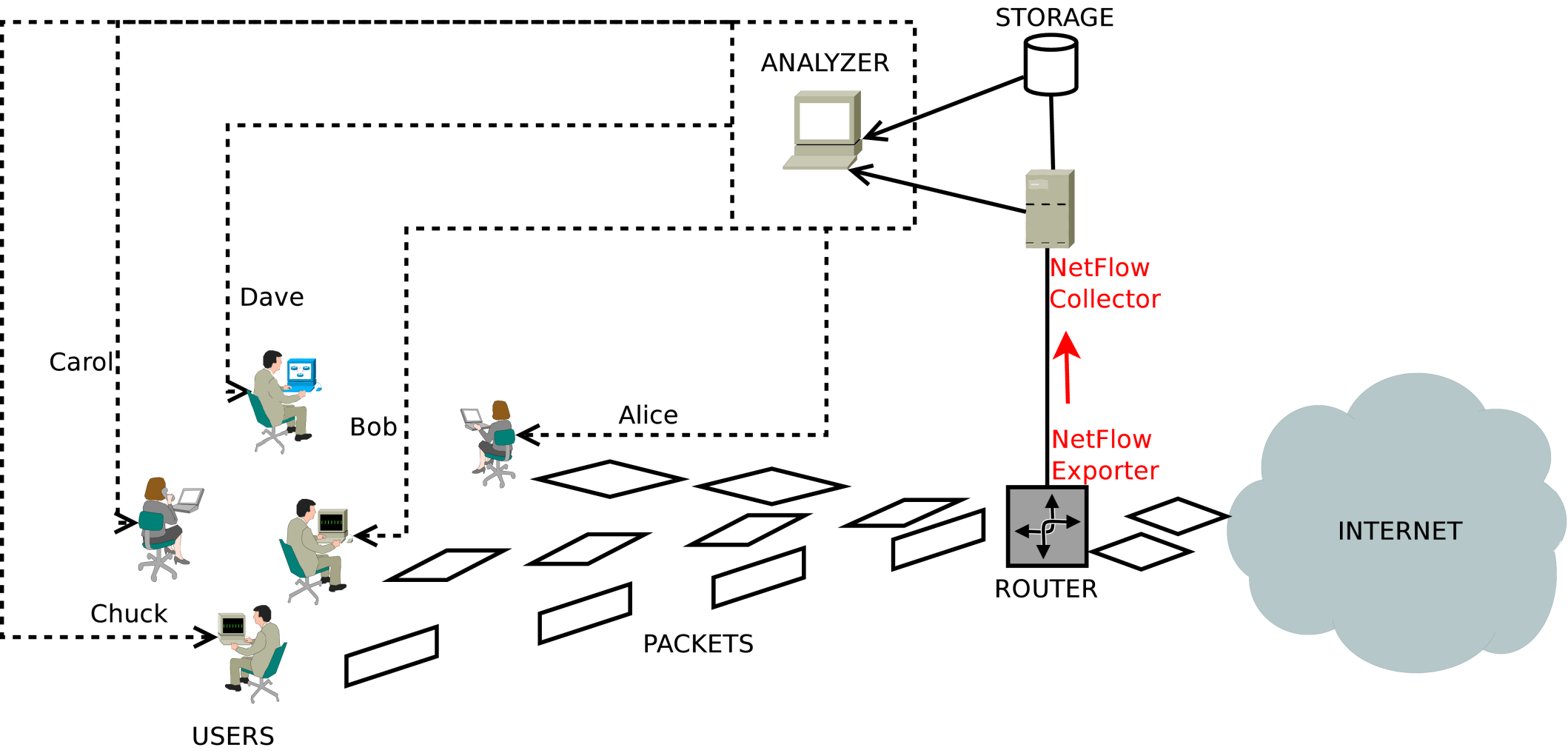
Authors: N. Melnikov & J. Schönwälder
Supervisor: Prof. J. Schönwälder
Presentation date: 25.06.2010

Overview

- Problem statement
- Chosen approach and results to date
- Why is the problem hard/important/relevant?
- Future research plan

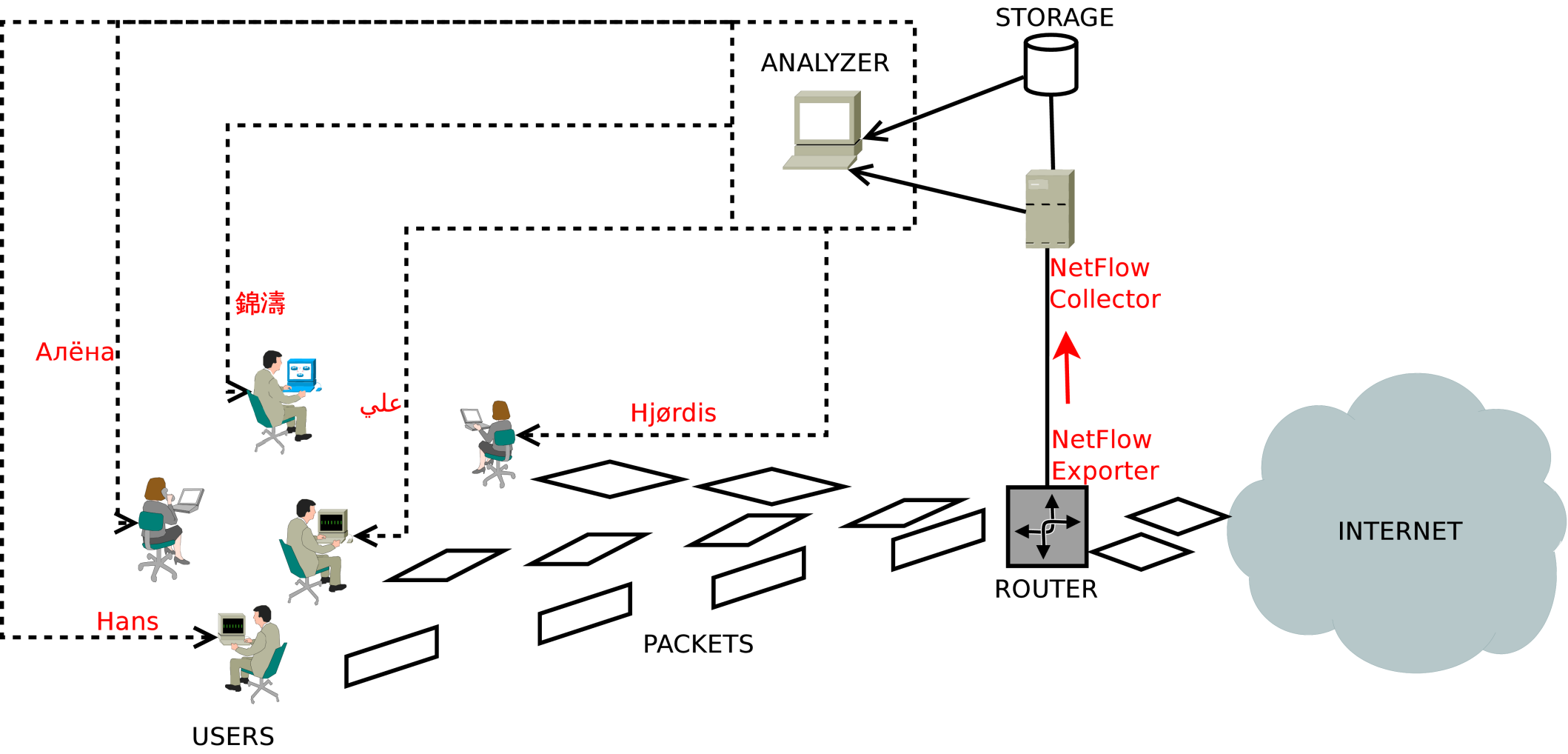
Problem Statement

Identify users based on their network activity (expressed in flow records)

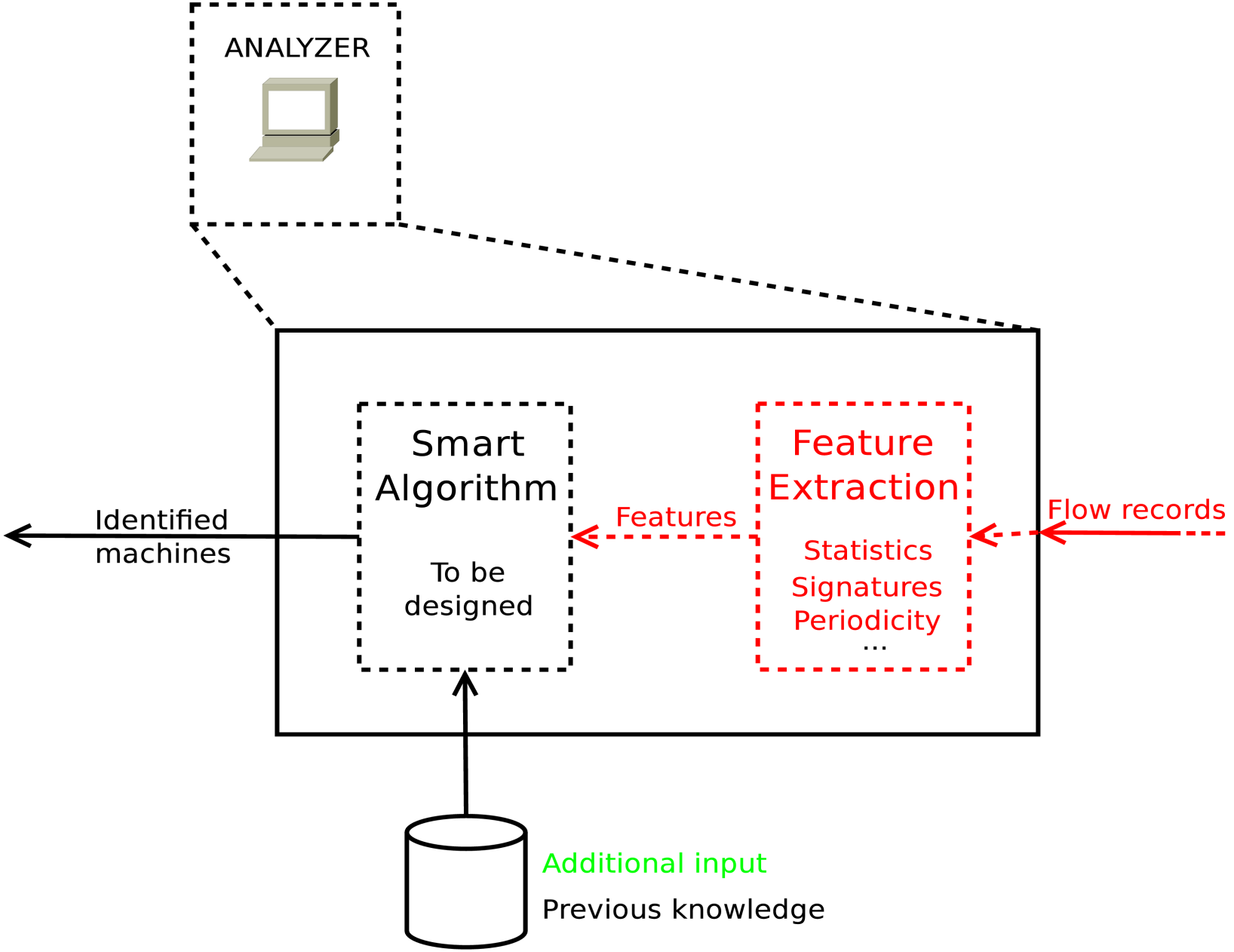


Problem Statement

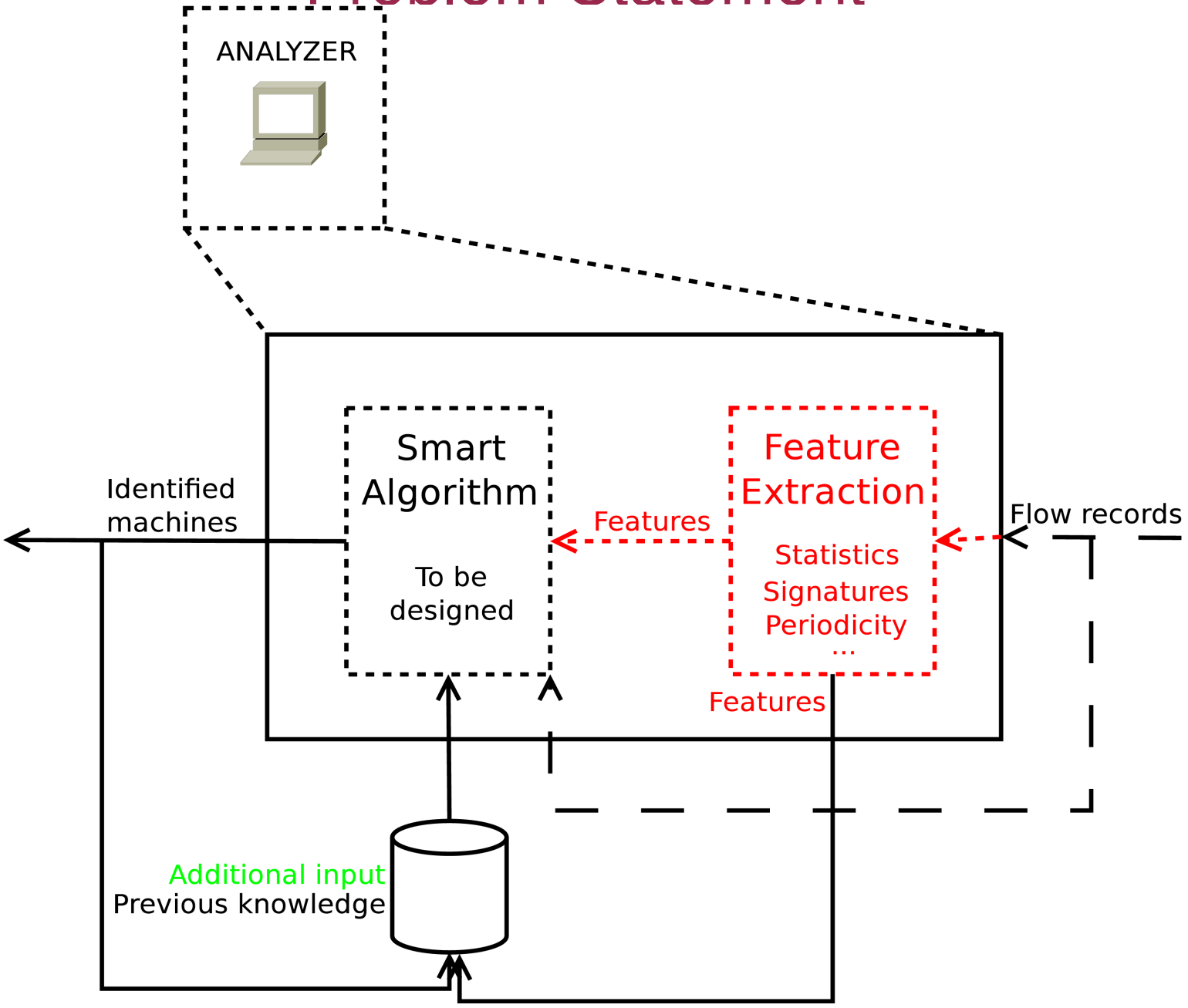
Identify users based on their network activity (expressed in flow records)



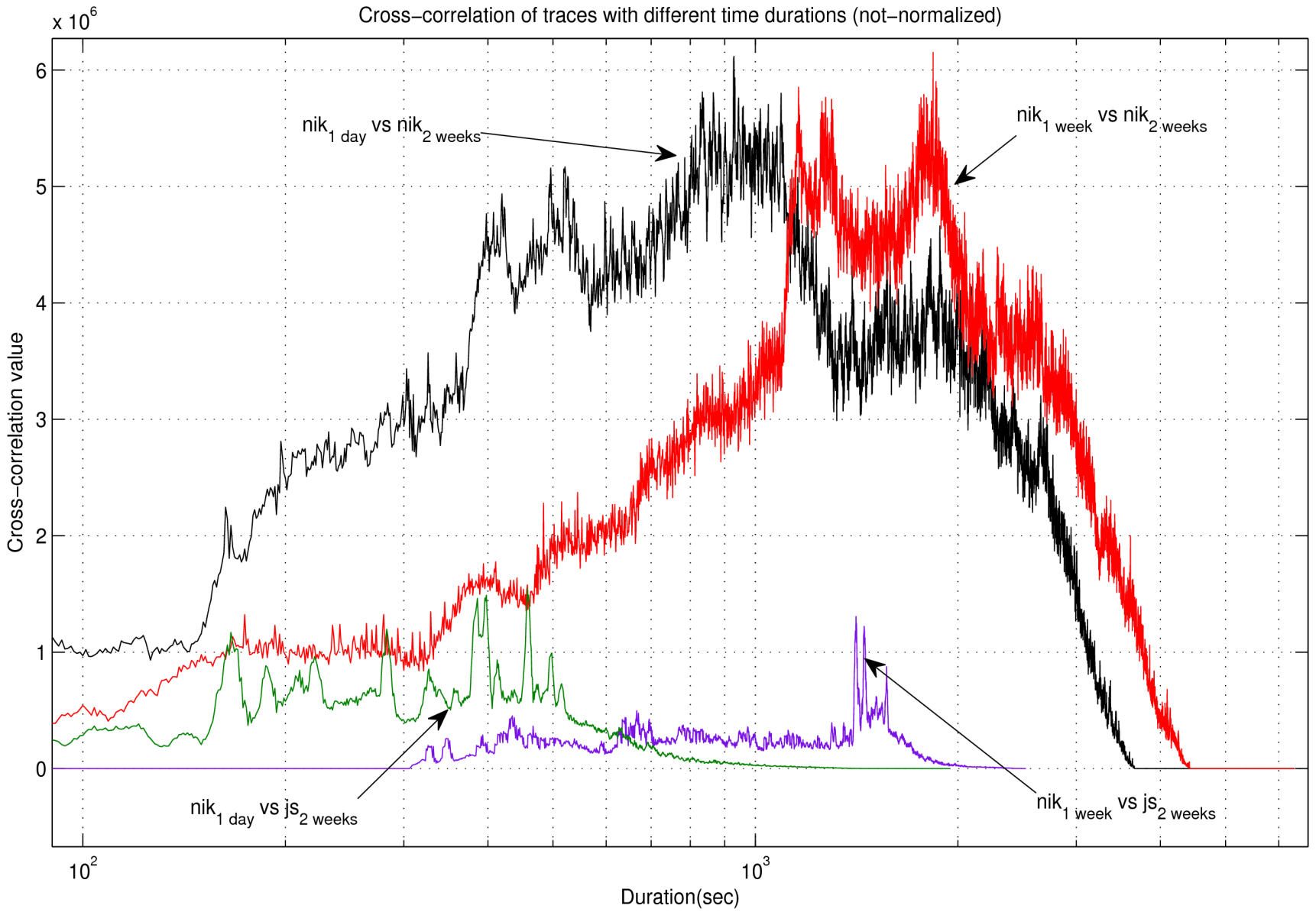
Problem Statement



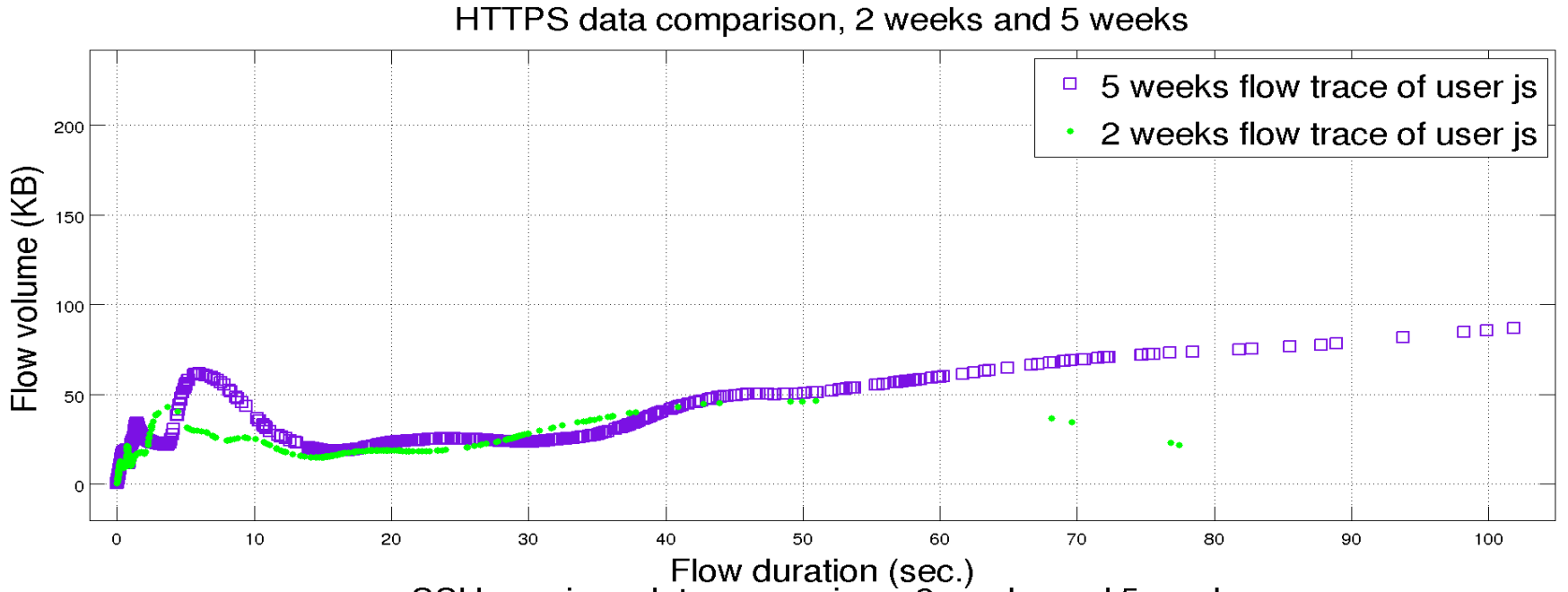
Problem Statement

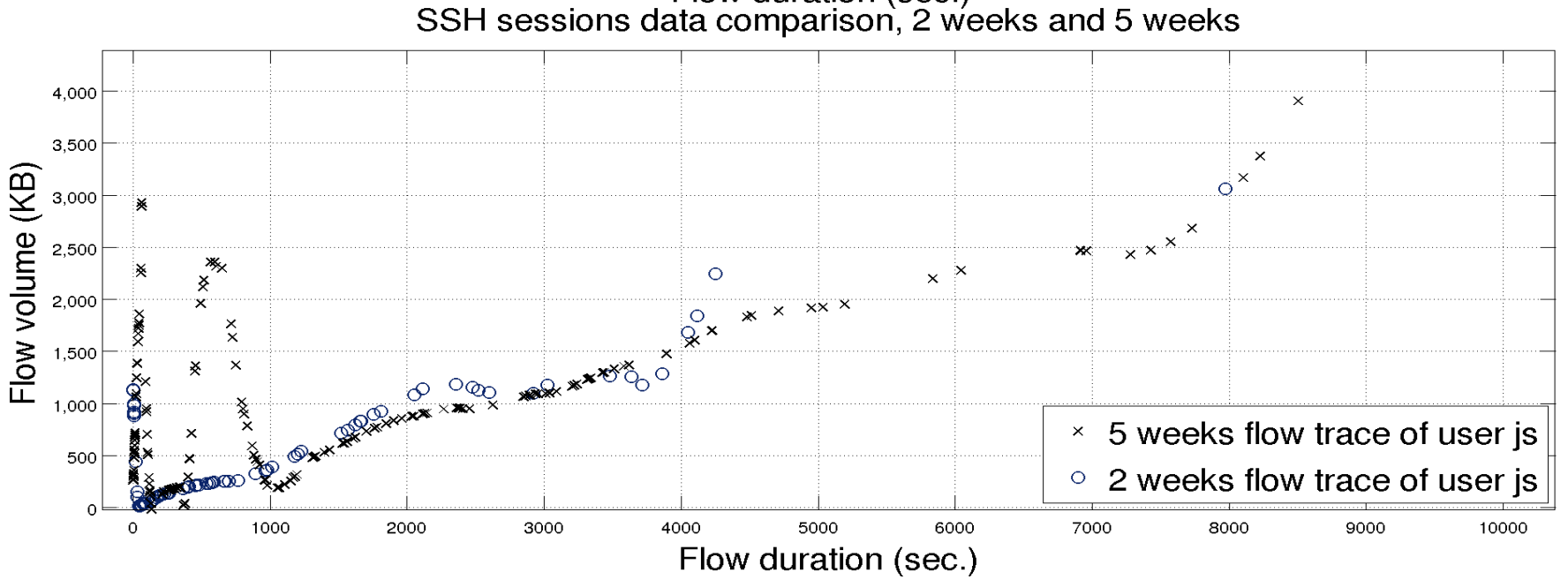


Chosen Approach - Statistics



Chosen Approach - Statistics

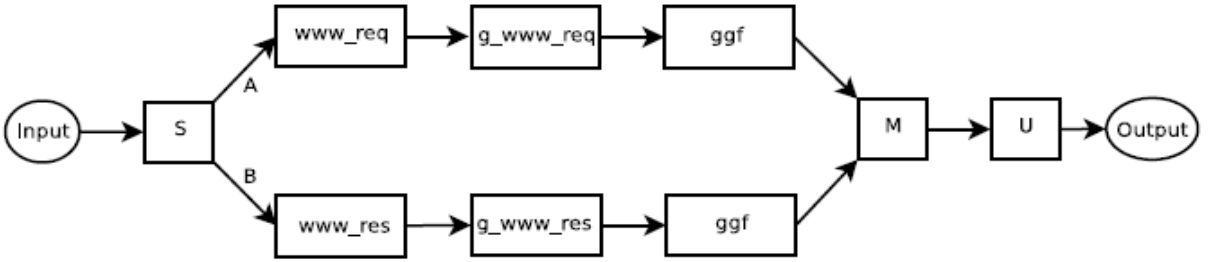




Chosen Approach - Signatures

- *We defined application signatures using a stream-based flow query language Flowy*

```
 1 splitter S {}
 2
 3 filter www_req {
 4    dstport = 80
 5 }
 6
 7 filter www_res {
 8    srcport = 80
 9 }
10
11 grouper g_www_req {
12    module g1 {
13       srcip = srcip
14       dstip = dstip
15       etime < stime delta 1s
16    }
17    aggregate srcip, dstip, sum(bytes) as bytes, count(rec_id) as n,
18              bitOR(tcp_flags) as flags, union(srcport) as srcports
19 }
20
21 grouper g_www_res {
22    module g1 {
23       srcip = srcip
24       dstip = dstip
25       etime < stime delta 1s
26    }
27    aggregate srcip, dstip, sum(bytes) as bytes, count(rec_id) as n,
28              bitOR(tcp_flags) as flags, union(dstport) as dstports
```



Chosen Approach - Signatures

- *We defined application signatures using a stream-based flow query language Flowy*

```
31 groupfilter ggf {
32    bitAND(flags, 0x13) = 0x13
33 }
34
35 merger M {
36    module m1 {
37       branches B, A
38       A.srcip = B.dstip
39       A.srcports = B.dstports
40       A.bytes < B.bytes
41       B oi A OR B d A
42    }
43    export m1
44 }
45
46 ungrouper U {}
47
48 "./netflow-trace.h5" -> S
49 S branch A -> www_req -> g_www_req -> ggf -> M
50 S branch B -> www_res -> g_www_res -> ggf -> M
51 M->U->"./ungrouped.h5"
```

Chosen Approach - Features

User	Skype	Opera	Amarok	Chrome	Live
U0	⊙	○	⊡	○	○
U1	⊙	○	○	○	○
U2	○	○	○	○	○
U3	⊙	○	⊡	○	○
U4	○	○	○	○	○
U5	⊙	○	⊙	⊙	○
U6	○	○	○	○	○
U7	○	⊙	⊙	○	○
U8	○	○	○	○	○
U9	⊙	⊙	⊙	⊙	○

TABLE I

RESULTS OF APPLICATION SIGNATURE IDENTIFICATIONS

Why is the problem hard/important/relevant?

No “step-by-step” methodology exists; large amounts of data

First ones to address it

As the application-generated traffic evolves and becomes specific

A shift from traffic just being used to being more network-aware

Think in terms of electric grid monitoring (appliances as apps): better job of distributing traffic where, what type of and how much of it is required

QoS, traffic engineering, monitoring

Future research plan

- *Establish more application signatures for Flowy*
- *Observation phase*
  - *Observe and understand dynamics of application signatures*
- *“Uncontrolled” testing phase for signatures*
  - *Using real-life flow records*
  - *Span several days/weeks to detect correlated variations*
  - *Day-time analysis of application employment*
- *Resolve some scalability issues (e.g., possible application of Map/Reduce)*

¡Thank you for attention!