
# Network Analysis with SiLK

Ron Bandes

SEI/CERT Network Situational Awareness

**NO WARRANTY**

# Learning Objectives

At the end of this module, you will have the knowledge and skills needed to perform the following tasks:

- Name the major components of SiLK.
- Retrieve network flow records using the rwfilter command.
- Manipulate network flow records using basic SiLK commands.
- Count and profile network flow records using basic SiLK commands.

# Outline

**Introduction: SiLK**

Network flow

Basic SiLK tools

Advanced SiLK tools

Summary

# What SiLK Does

Optimized for extremely large data collections

- Very compact record format
- Large amount of history can stay online.

Command line interface

- Good for scripting & repeating commands with small modifications.

Retrospective analysis

- most useful for analyzing past network events
- may feed an automated report generator
- good for forensics (what happened ***before*** the incident?)

# Outline

Introduction: SiLK

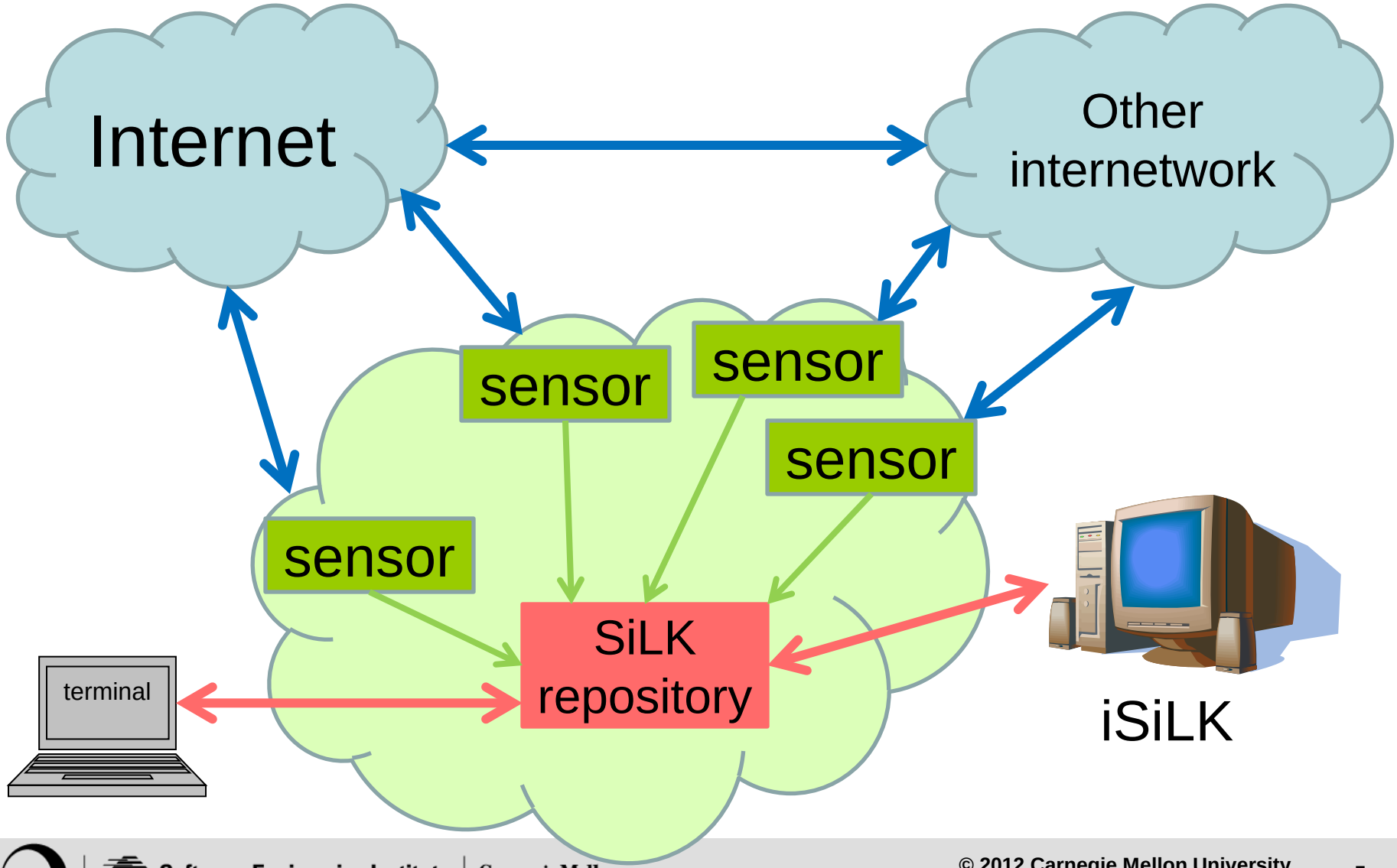
**Network flow**

Basic SiLK tools

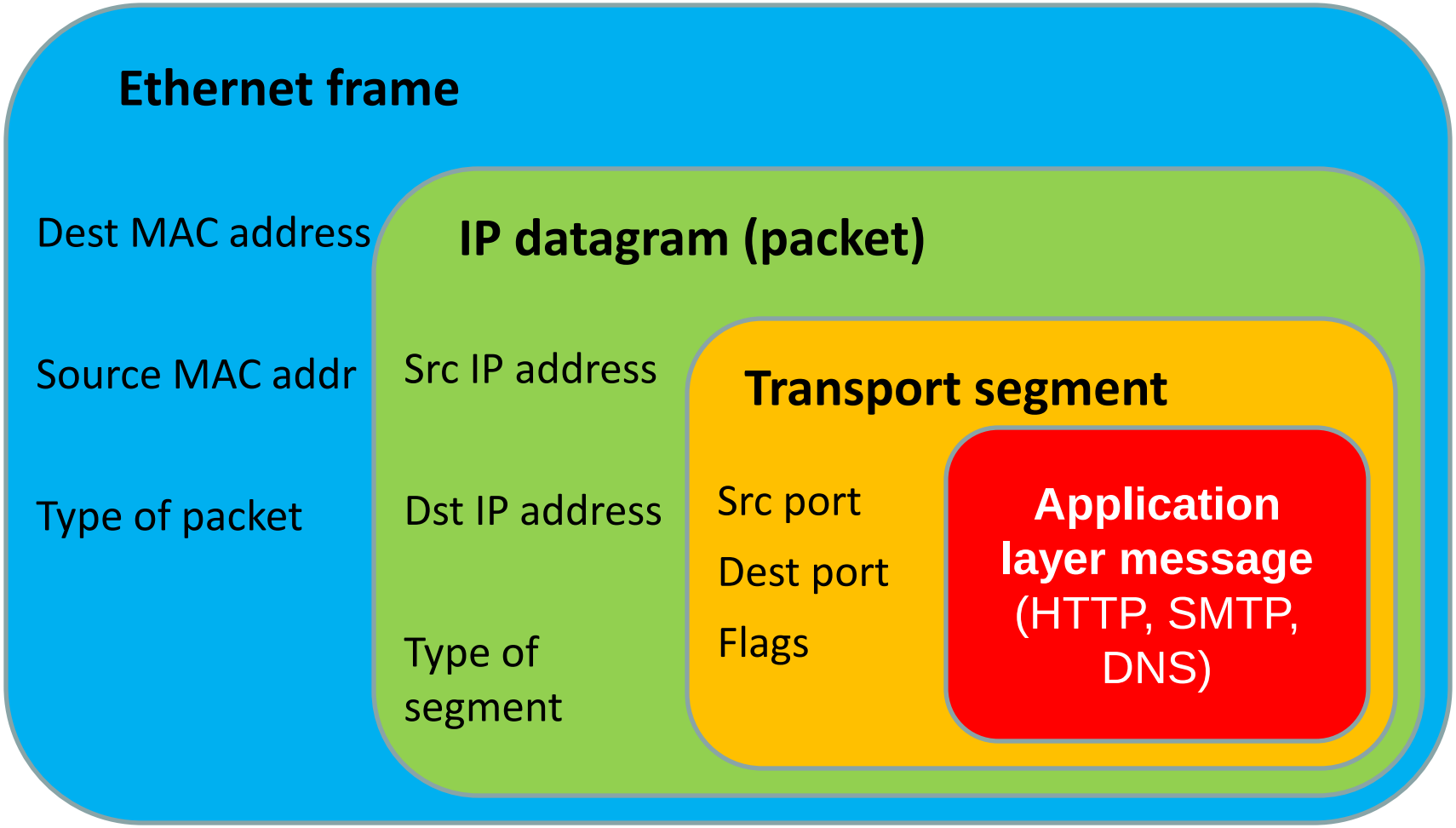
Advanced SiLK tools

Summary

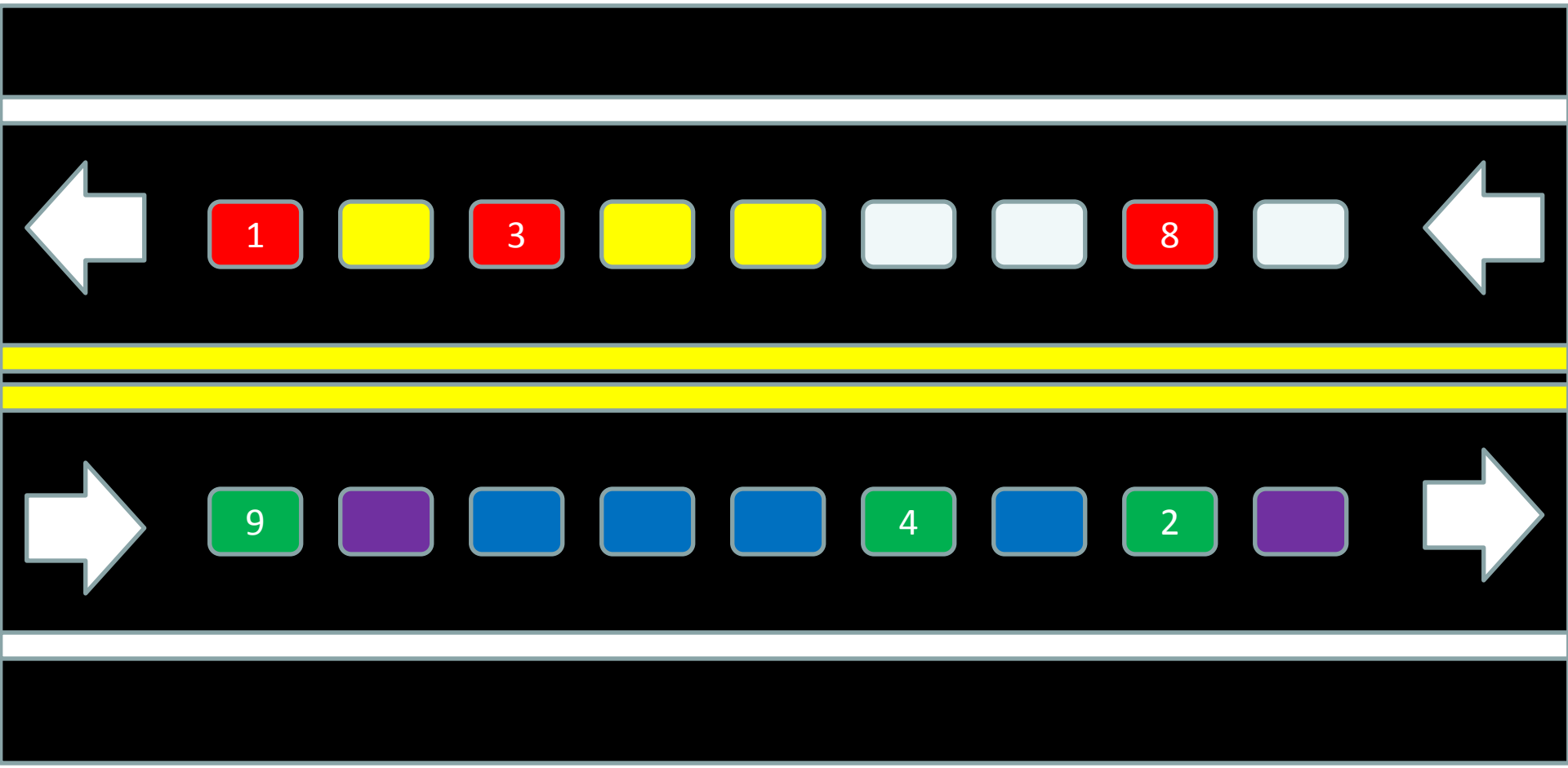
# Network Monitoring



# Packet Encapsulation



# Flows

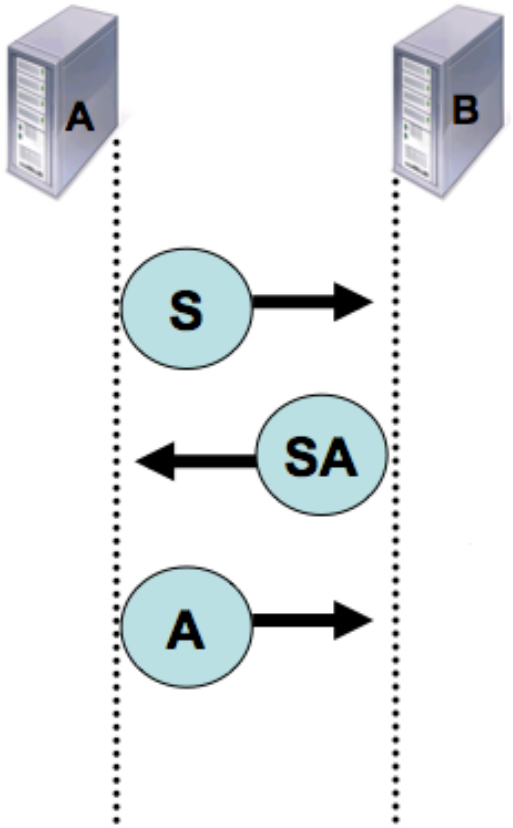


# What Is a Flow?

A flow is an aggregated record of packets.

SiLK flows are ID'd by five attributes:

- source IP address
- destination IP address
- source port
- destination port
- transport protocol (any of about 130 in use)




SiLK flows are unidirectional:

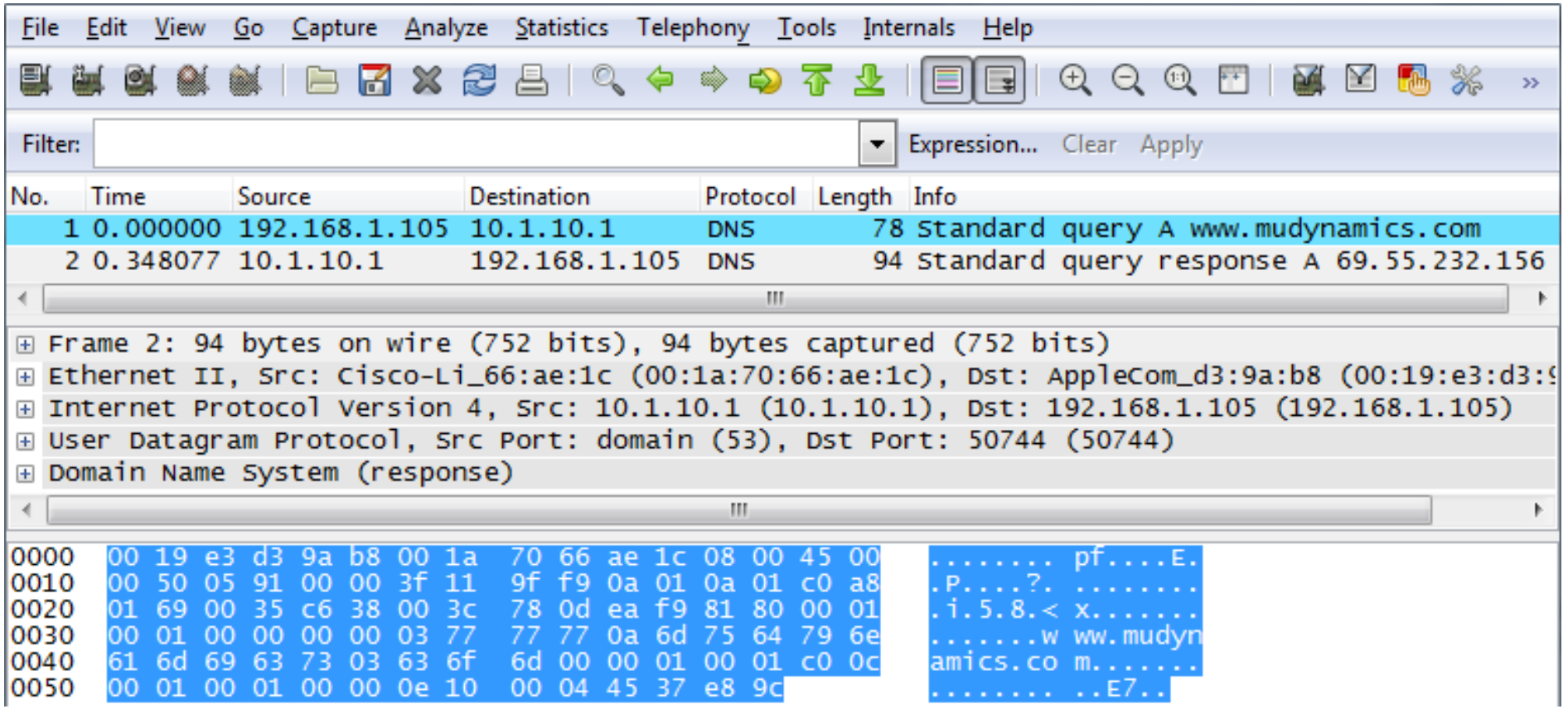
- Newly observed attributes, new flow
- Previously observed attributes, update flow

# What's in a Record?

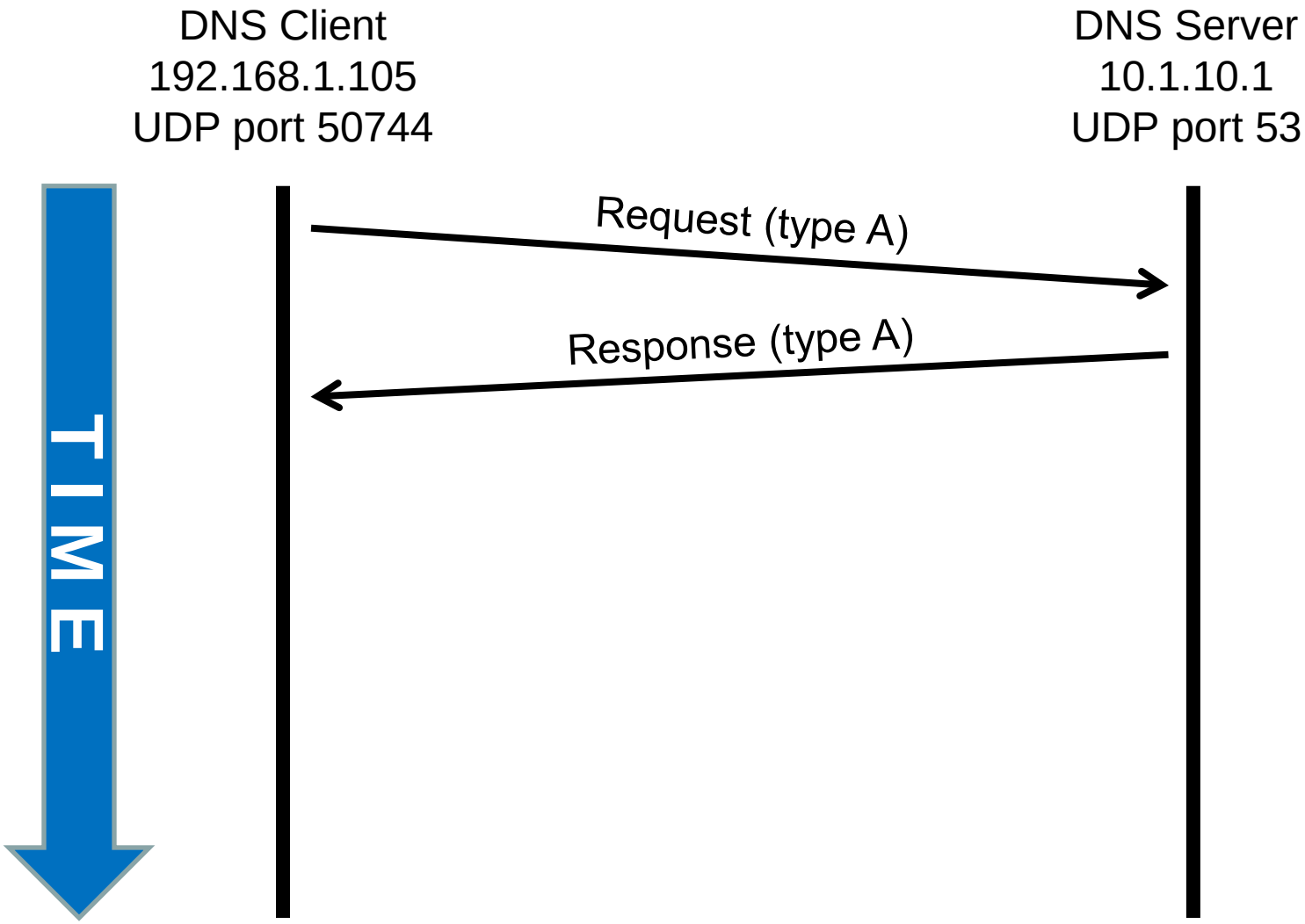
Fields found to be useful in analysis:

- source address, destination address
- source port, destination port (Internet Control Message Protocol [ICMP] type/code)
- IP [transport] protocol
- bytes, packets in flow
- accumulated TCP flags (all packets, first packet)
- start time, duration (milliseconds)
- end time (derived)
- sensor identity
- flow termination conditions
- application-layer protocol

# DNS packets viewed in Wireshark



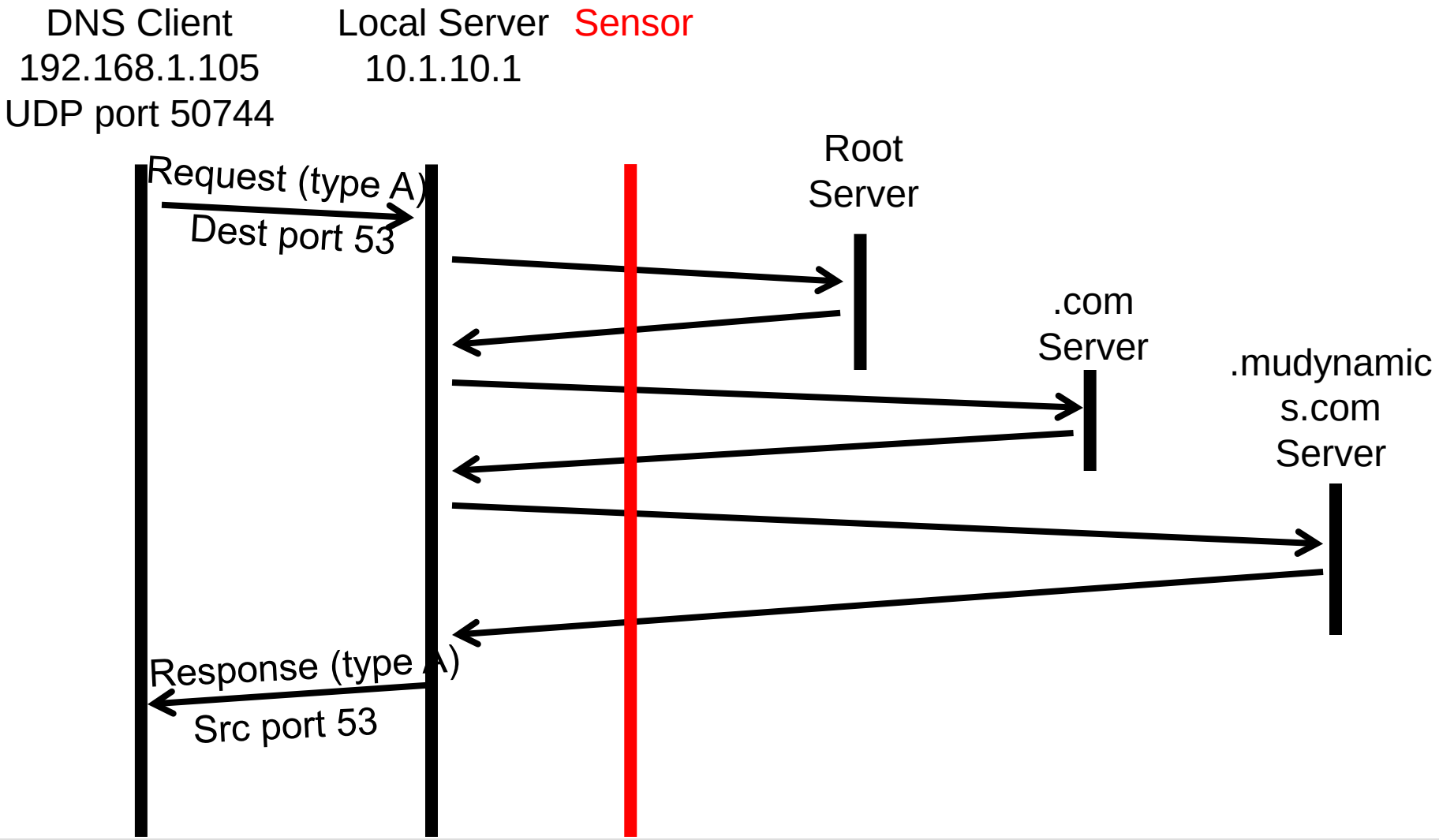
# Sequence Diagram



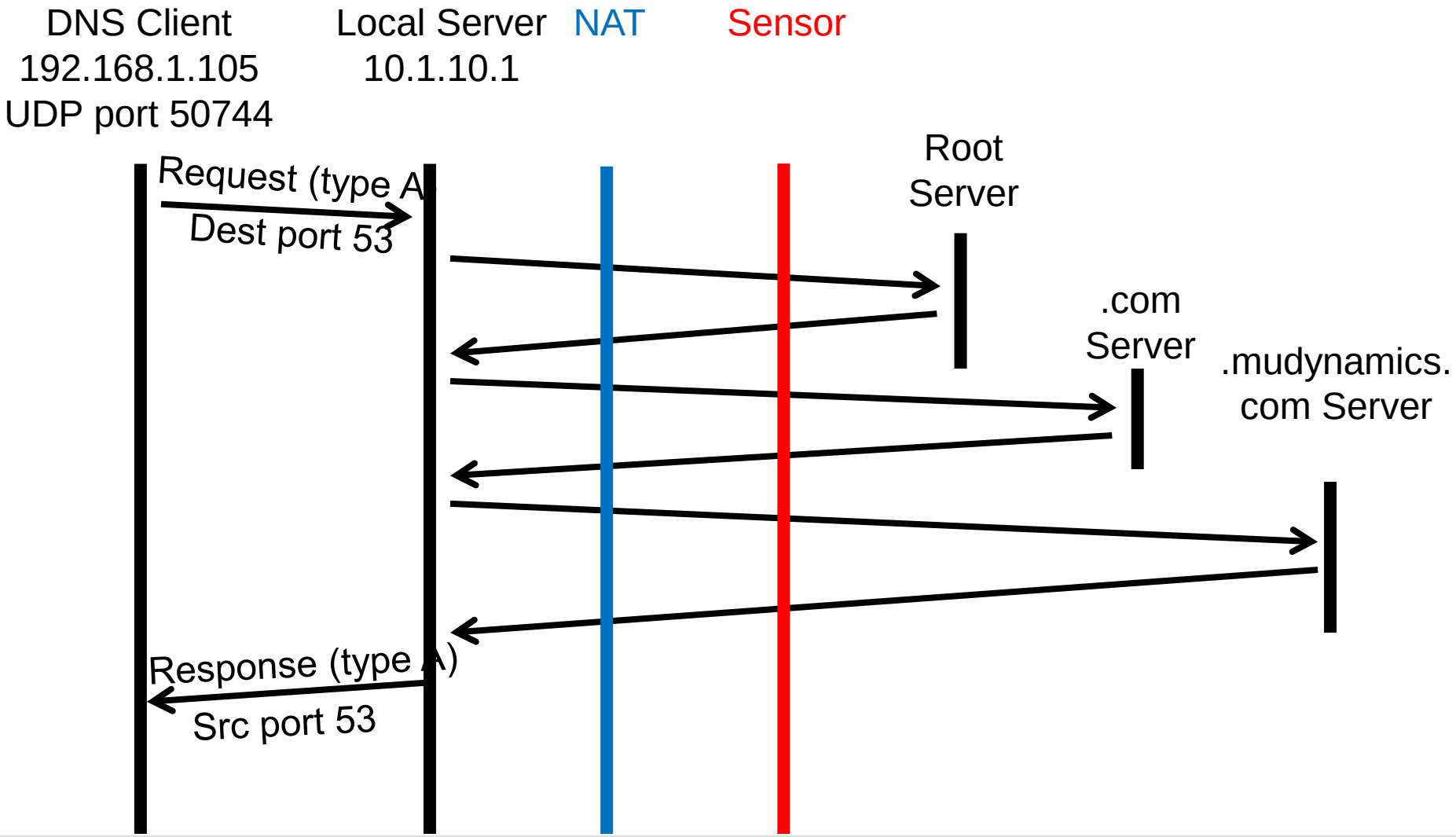
# SiLK tool (rwcut) output

```
          sIP|          dIP|sPort|dPort|pro|packets|bytes|sensor|type|
192.168.1.105|    10.1.10.1|50744|   53| 17|      1|   64|    S1| out|
    10.1.10.1|192.168.1.105|   53|50744| 17|      1|   80|    S1|  in|
```

# Realistic Sequence Diagram



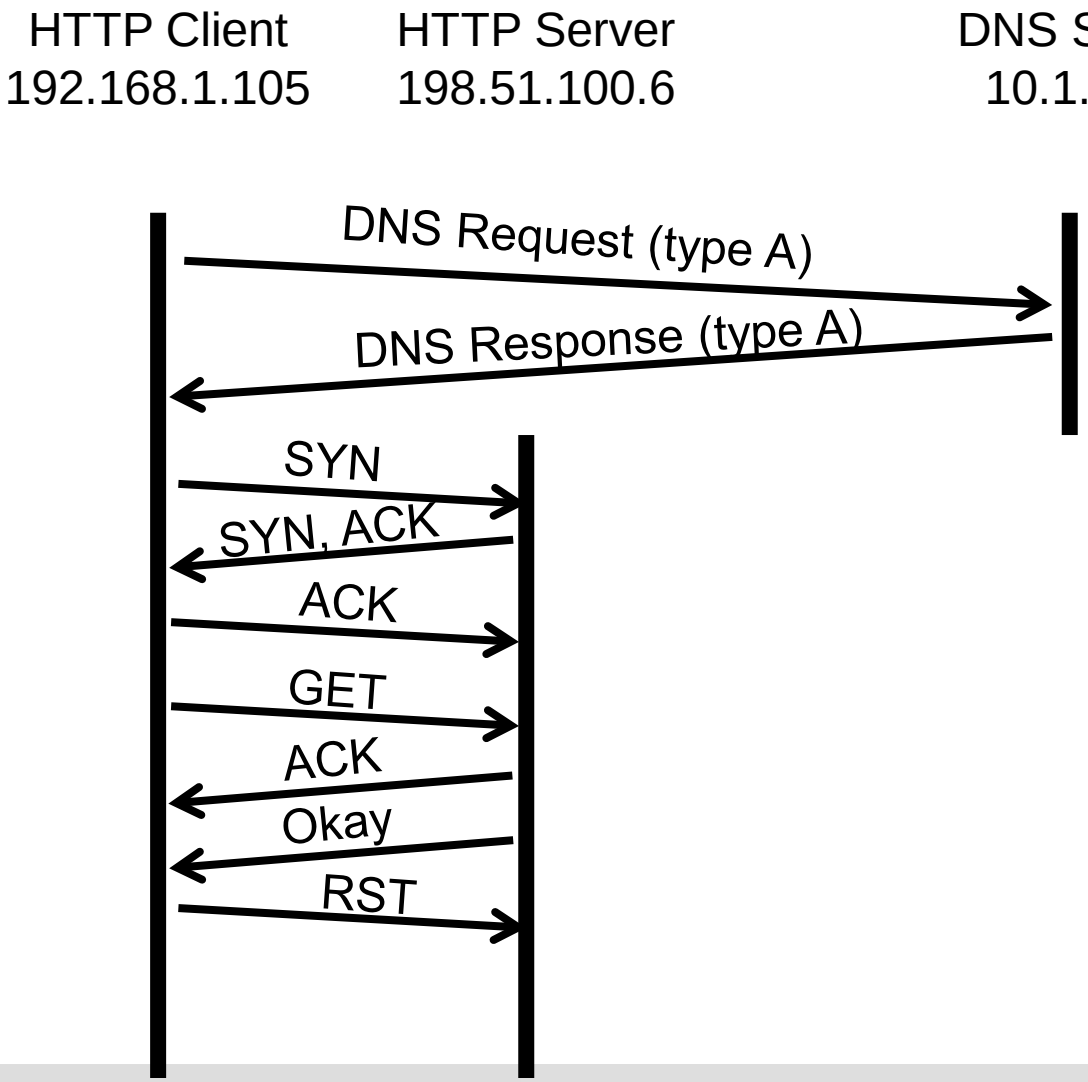
# More Realistic Sequence Diagram



# What is this?

| sIP | dIP | sPort | dPort | pro | packets | flags | initF | type |
|---|---|---|---|---|---|---|---|---|
| 192.168.1.105 | 10.1.10.1 | 50744 | 53 | 17 | 1 | | | out |
| 10.1.10.1 | 192.168.1.105 | 53 | 50744 | 17 | 1 | | | in |
| 192.168.1.105 | 198.51.100.6 | 49152 | 80 | 6 | 4 | SRPA | S | outweb |
| 198.51.100.6 | 192.168.1.105 | 80 | 49152 | 6 | 3 | S PA | S A | inweb |

# HTTP Sequence Diagram



# What Is This? #1

| sIP | dIP | sPort | dPort | pro | packets | bytes | flags |
|---|---|---|---|---|---|---|---|
| 30.22.105.250 | 71.55.40.253 | 52415 | 25 | 6 | 22 | 14045 | FSRPA |
| 71.55.40.253 | 30.22.105.250 | 25 | 52415 | 6 | 19 | 1283 | FS PA |
| 30.22.105.250 | 71.55.40.253 | 52415 | 25 | 6 | 1 | 40 | R |

# What Is This? #2

| sIP | dIP | pro | packets | bytes | sTime |
|---|---|---|---|---|---|
| 99.217.139.155 | 177.252.24.89 | 1 | 2 | 122 | 2010/12/08T00:04:30.172 |
| 99.217.139.155 | 177.252.149.249 | 1 | 2 | 122 | 2010/12/08T00:04:37.302 |
| 99.217.139.155 | 177.252.24.52 | 1 | 2 | 122 | 2010/12/08T00:04:37.312 |
| 99.217.139.155 | 177.252.24.127 | 1 | 2 | 122 | 2010/12/08T00:04:58.363 |
| 99.217.139.155 | 177.252.24.196 | 1 | 2 | 122 | 2010/12/08T00:05:04.327 |
| 99.217.139.155 | 177.252.149.30 | 1 | 2 | 122 | 2010/12/08T00:05:09.242 |
| 99.217.139.155 | 177.252.149.173 | 1 | 2 | 122 | 2010/12/08T00:05:12.174 |
| 99.217.139.155 | 177.252.24.13 | 1 | 2 | 122 | 2010/12/08T00:05:14.114 |
| 99.217.139.155 | 177.252.24.56 | 1 | 2 | 122 | 2010/12/08T00:05:15.383 |
| 99.217.139.155 | 177.252.24.114 | 1 | 2 | 122 | 2010/12/08T00:05:18.228 |
| 99.217.139.155 | 177.252.202.92 | 1 | 2 | 122 | 2010/12/08T00:05:22.466 |
| 99.217.139.155 | 177.252.202.68 | 1 | 2 | 122 | 2010/12/08T00:05:23.497 |
| 99.217.139.155 | 177.252.24.161 | 1 | 2 | 122 | 2010/12/08T00:05:30.256 |
| 99.217.139.155 | 177.252.202.238 | 1 | 2 | 122 | 2010/12/08T00:05:33.088 |

# What Is This? #3

| sIP | dIP | sPort | dPort | pro | packets | bytes | flags | sTime |
|---|---|---|---|---|---|---|---|---|
| 88.187.13.78 | 71.55.40.204 | 40936 | 80 | 6 | 83 | 3512 | FS PA | 2010/12/08T11:00:01.318 |
| 71.55.40.204 | 88.187.13.78 | 80 | 40936 | 6 | 84 | 104630 | FS PA | 2010/12/08T11:00:01.336 |
| 88.187.13.78 | 71.55.40.204 | 40938 | 80 | 6 | 120 | 4973 | FS PA | 2010/12/08T11:00:04.483 |
| 71.55.40.204 | 88.187.13.78 | 80 | 40938 | 6 | 123 | 155795 | FS PA | 2010/12/08T11:00:05.001 |
| 88.187.13.78 | 71.55.40.204 | 56172 | 80 | 6 | 84 | 3553 | FS PA | 2010/12/08T12:00:02.116 |
| 71.55.40.204 | 88.187.13.78 | 80 | 56172 | 6 | 83 | 103309 | FS PA | 2010/12/08T12:00:02.133 |
| 88.187.13.78 | 71.55.40.204 | 56177 | 80 | 6 | 123 | 5093 | FS PA | 2010/12/08T12:00:05.276 |
| 71.55.40.204 | 88.187.13.78 | 80 | 56177 | 6 | 124 | 157116 | FS PA | 2010/12/08T12:00:05.294 |

# It’s All a Matter of Timing

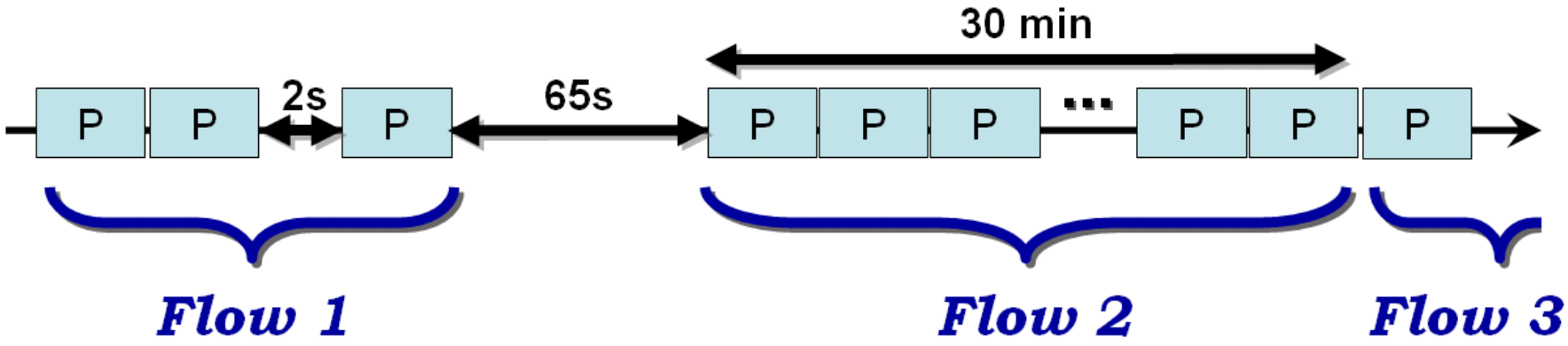
The flow buffer needs to be kept manageable.

Idle timeout

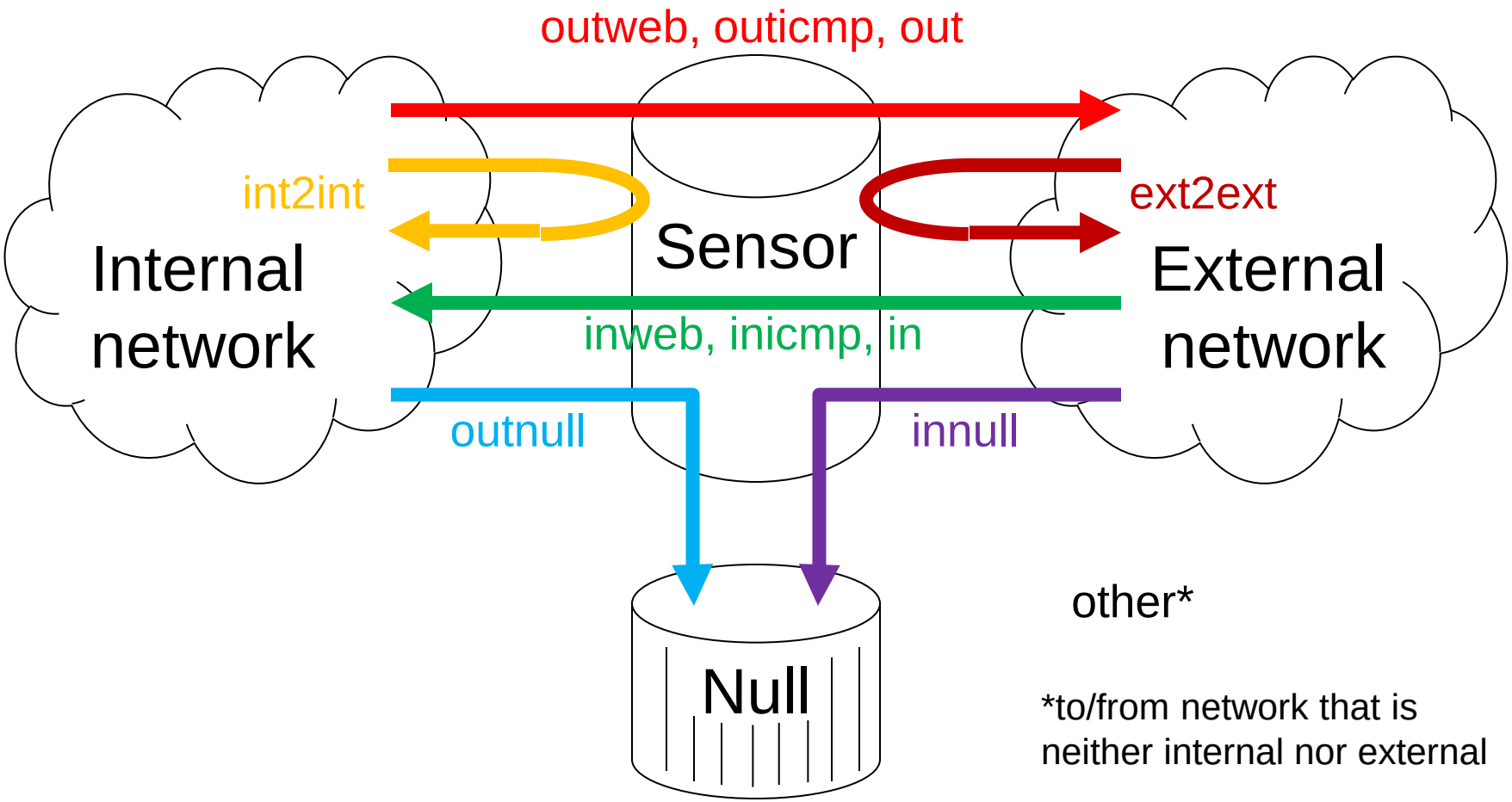
- If there is no activity within [30] thirty seconds, flush the flow.

Active timeout

- Flush all flows open for [30] thirty minutes.



# SiLK Types



# SiLK Types in SiLK

| Type | Description |
|---|---|
| **inweb**, outweb | Inbound/outbound TCP ports 80, 443, 8080 |
| innull, outnull | Inbound/outbound filtered traffic |
| **inicmp**, outicmp | Inbound/outbound IP protocol 1 |
| **in**, out | Inbound/outbound not in above categories |
| int2int, ext2ext | Internal to internal, external to external |
| other | Source not internal or external, or destination not internal, external, or null |

Names in **bold** are default types

# Got a Question? Flow Can Help

What's on my network?

What happened before the event?

Where are policy violations occurring?

What are the most popular websites?

By how much would volume be reduced with a blacklist?

Do my users browse to known infected web servers?

Do I have a spammer on my network?

When did my web server stop responding to queries?

Who uses my public servers?

# Outline

Introduction: SiLK

Network flow

**Basic SiLK tools**

Advanced SiLK tools

Summary

# UNIX / Linux commands

System prompt

Info + prompt character

e.g., **~ 101>**

User command

command name

options

arguments

redirections

pipe

e.g., rwcut --all-fields results.rw >results.txt

e.g., rwcut --fields=1-6 results.rw | more

# Some Terms

**SiLK**: A traffic analysis tool-suite which processes flow data.

**Flow**: the collection of packets travelling in the same direction in a TCP or UDP connection.

**Flow Record**: a single record containing summary information for a flow.

**Flow Repository**: a tree structure of flat files containing flow records.

# Collection, Packing, and Analysis

Collection of flow data

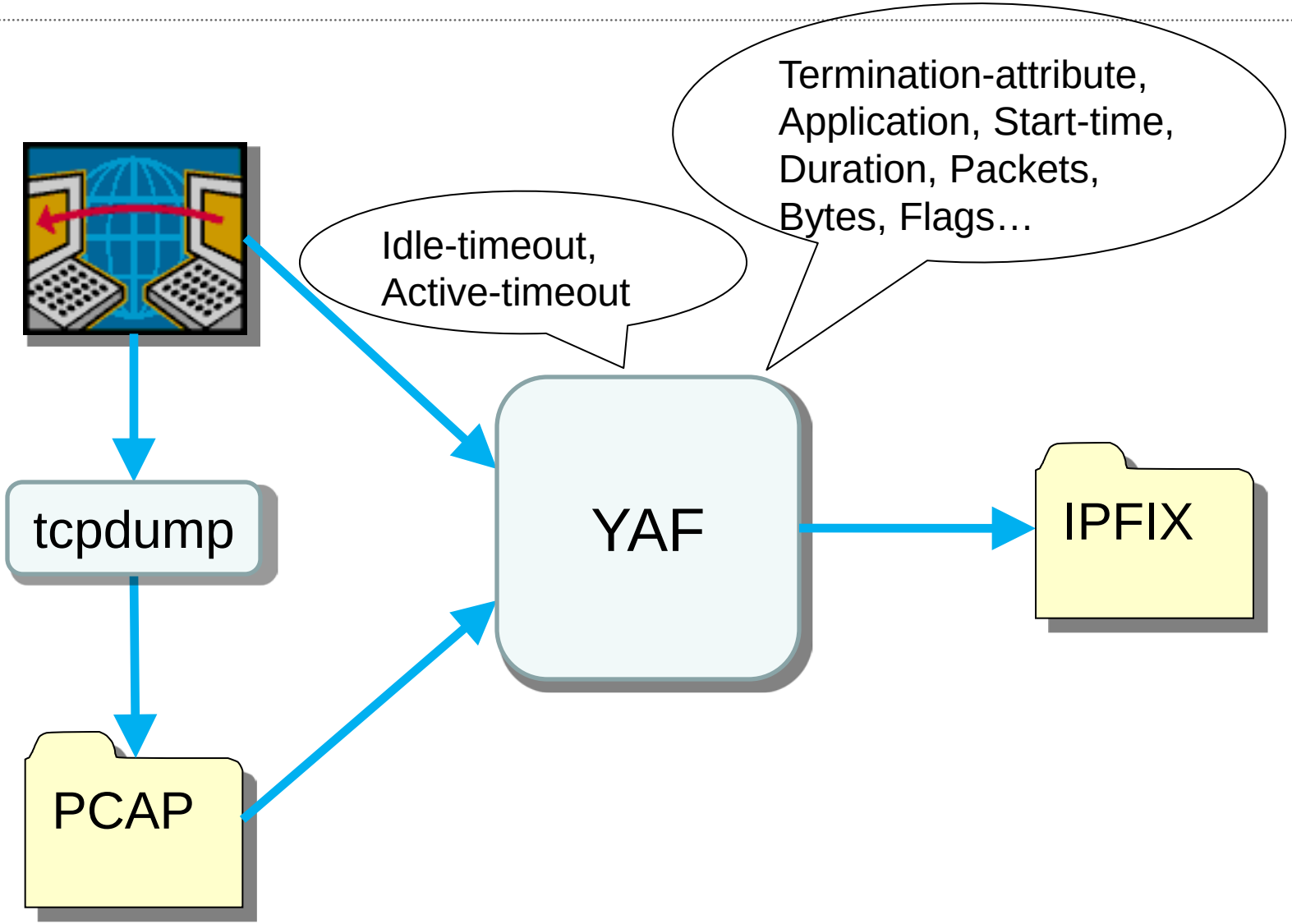
- Examines packets and summarizes into standard flow records
- Timeout and payload-size values are established during collection

Packing stores flow records in a scheme optimized for space and ease of analysis

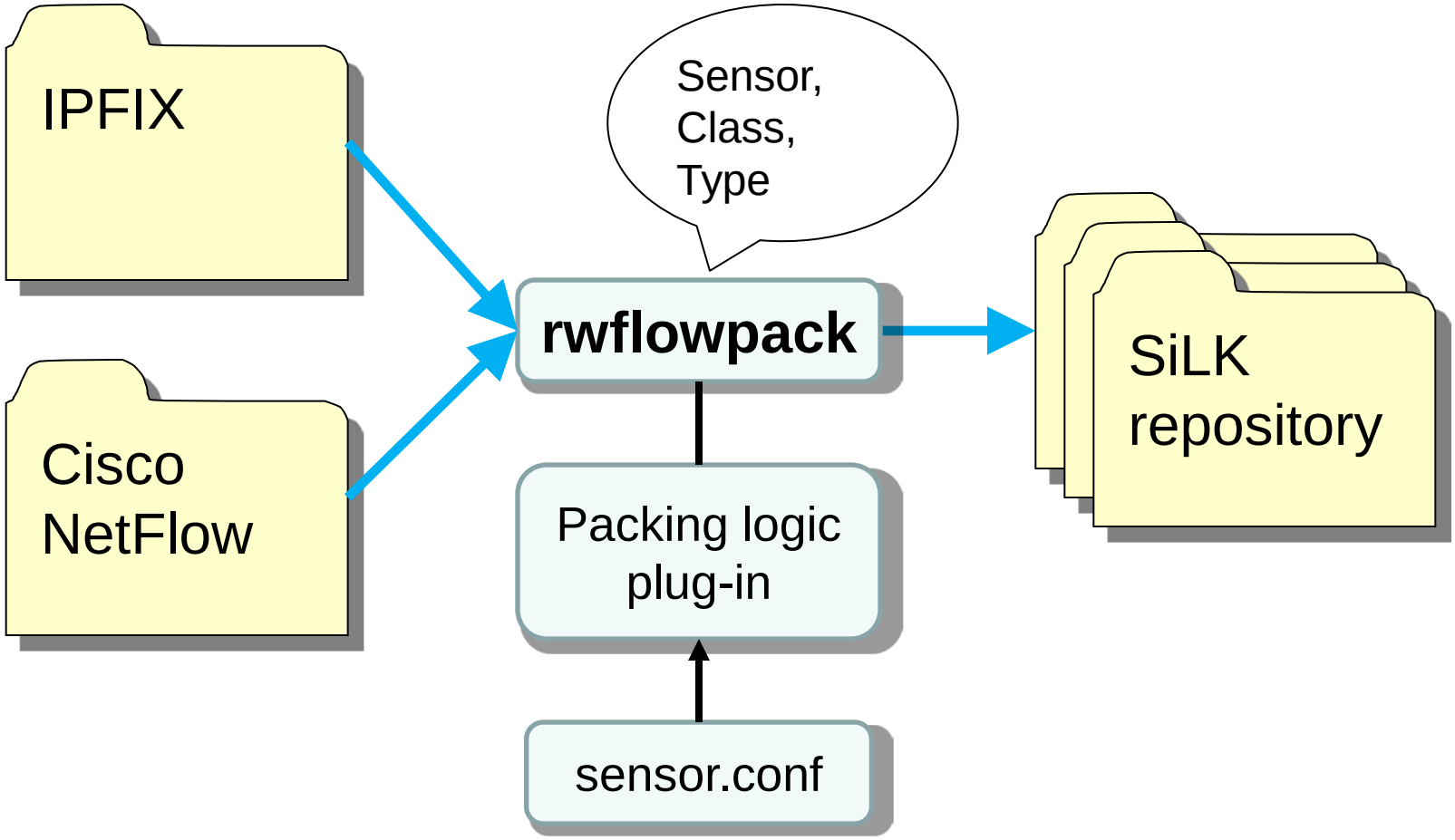
Analysis of flow data

- Investigation of flow records using SiLK tools

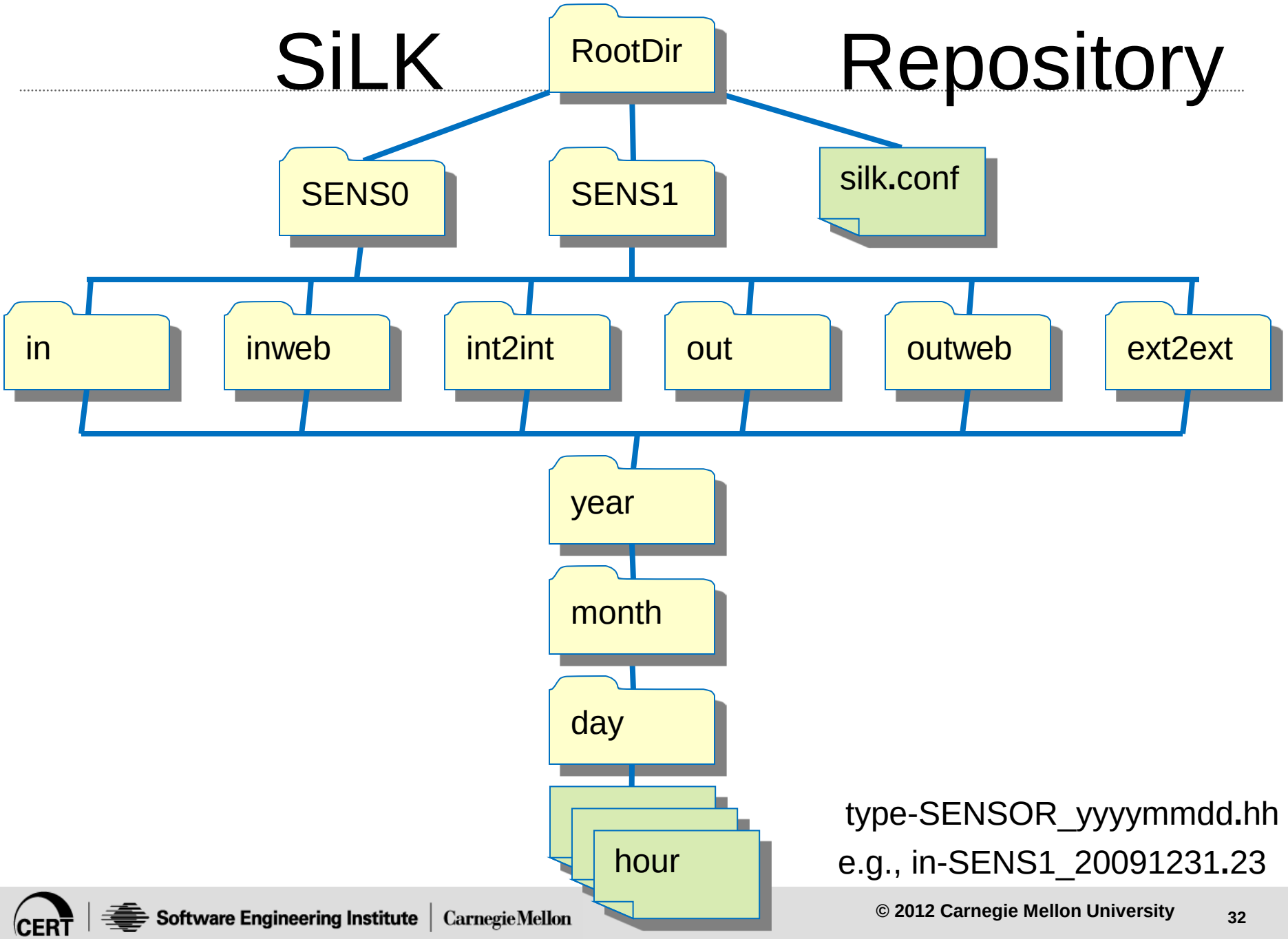
# Collection



# Packing



# SiLK Repository



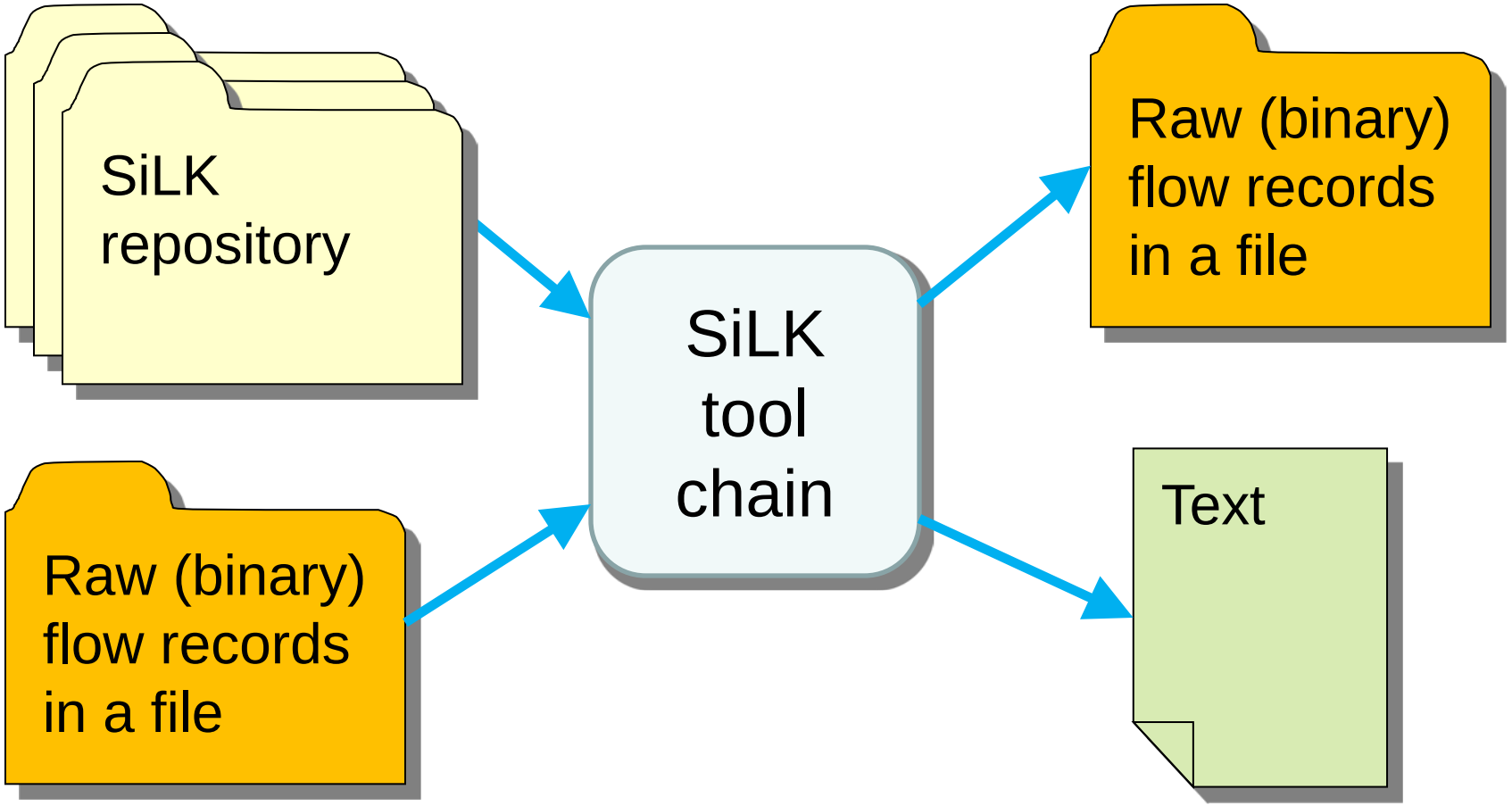

type-SENSOR_yyyymmdd.hh

e.g., in-SENS1_20091231.23

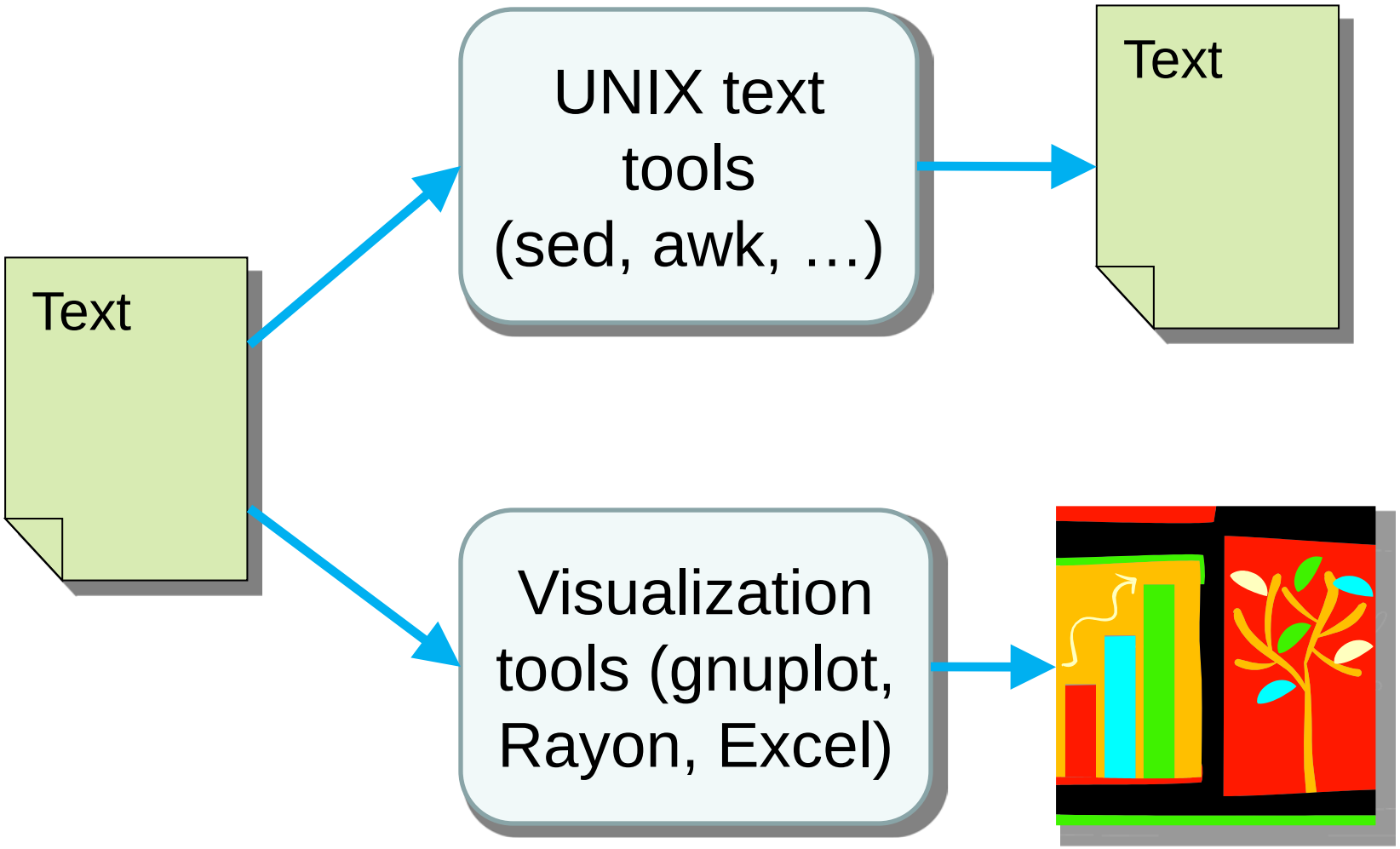
# Exercise

```
PS1='\W \!> '
export SILK_IPV6_POLICY=asv4
cd /data/bluered
ls -l silk.conf
less silk.conf   # type “q” to exit from less
export SILK_DATA_ROOTDIR=/data/bluered
cd
```

# Analysis



# Reporting



# So Much to Do, So Little Time...

We can’t discuss all parameters for every tool.

Resources

- Analyst’s Handbook
- SiLK Reference Guide (hard-copy man pages)
- **`--help`** option
- **`man`** command
- **`http://tools.netsa.cert.org`**

# Exercise

```
mapsid --help
man mapsid   # type “q” to exit from man
mapsid
mapsid --print-descriptions
```

# Basic SiLK Tools: `rwfileinfo`

**`rwfileinfo`** displays a variety of characteristics for each file format produced by the SiLK tool suite.

It is very helpful in tracing how a file was created and where it was generated.

# rwfileinfo Example

```
SiLK> rwfileinfo sportSMTP.rw
sportSMTP.rw:
  format(id)           FT_RWGENERIC(0x10)
  version              16
  byte-order           littleEndian
  compression(id)      lzo1x(2)
  header-length        352
  record-length        88
  record-version       1
  silk-version         2.4.4
  count-records        5
  file-size            523
  command-lines
    1  rwfilter --type=in,out --start=2010/12/08
--end=2010/12/10 --pass=sportSMTP.rw
--any-address=139.72.231.133 --print-file --print-vol
```

# Basic SiLK Tools: `rwcut`

But I can't read binary...

**`rwcut`** provides a way to display binary records as human-readable ASCII:

- useful for printing flows to the screen
- useful for input to text-processing tools
- Usually you'll only need the **`--fields`** argument.

| | | | | |
|---|---|---|---|---|
| sip | packets | type | flags | application |
| dip | bytes | in | initialflags | *icmptypecode* |
| sport | sensor | out | sessionflags | attributes |
| dport | *scc* | dur | *dur+msec* | *stype* |
| protocol | *dcc* | stime | *stime+msec* | *dtype* |
| class | nhip | *etime* | *etime+msec* | |

Field names in italics are *derived* fields

# Rwcut Default Display

By default

- sIP, sPort
- dIP, dPort
- protocol
- packets, bytes
- flags
- sTime, eTime, duration
- sensor

**`--all-fields`**

# `--num-recs --start --end`

These allow analyst to specify a slice of the records to display.

**`num-recs`**: how many records should **`rwcut`** display

**`start`**: how far from the top should **`rwcut`** start

**`end`**: how far from the bottom should **`rwcut`** start

Quick data look:

```
rwcut myfile.raw --num-recs=20 --fields=1-7,9
```

# Pretty Printing SiLK Output

Default output is fixed-width, pipe-delimited data.

```
            sIP|            dIP|pro|pkts|bytes|
207.240.215.71| 128.3.48.203|  1|   1|   60|
207.240.215.71|  128.3.48.68|  1|   1|   60|
207.240.215.71|  128.3.48.71|  1|   1|   60|
```

Tools with text output have these formatting options:

- **`--no-titles`**: suppress the first row
- **`--no-columns`**: suppress the spaces
- **`--delimited`**: change how columns are marked
- **`--column-separator:`** just change the bar to something else
- **`--legacy-timestamps`**: better for import to Excel

# rwcut exercise

```
cd /data/bluered
rwcut --num-recs=20 --fields=1-6 \
  S0/in/2009/04/21/in-S0_20090421.00
cd
```

Try other values for **`--fields`**.

Try **`--end=2`** and **`--no-titles`**.

# Basic SiLK Tools: `rwfilter`

# `rwfilter` Syntax

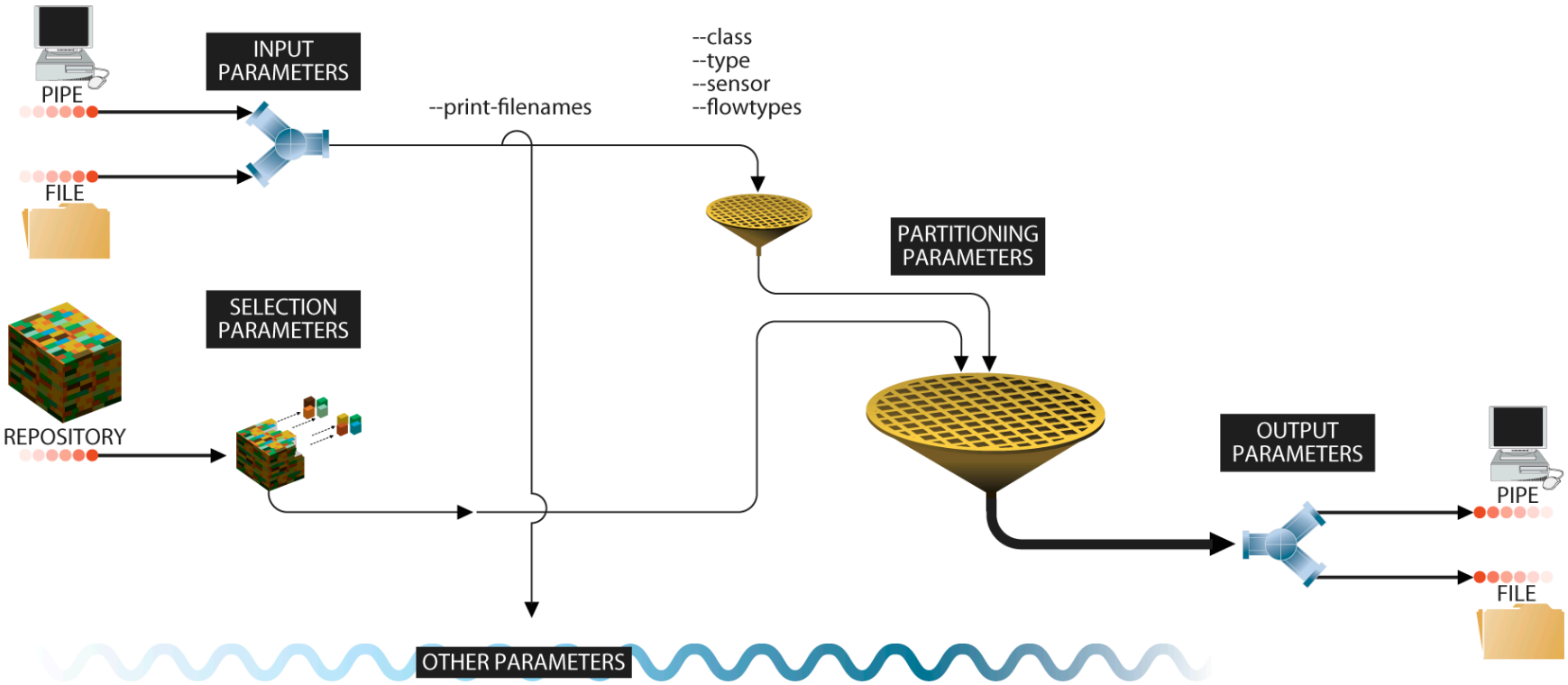
General form

```
rwfilter {INPUT | SELECTION}
  [PARTITION] [OUTPUT] [OTHER]
```

Example call

```
rwfilter --start-date=2010/12/10:00 \
  --end-date=2010/12/10:23 --type=in \
  --protocol=0-255 --pass=all-10.raw
```

# `rwfilter` Flow of Parameters

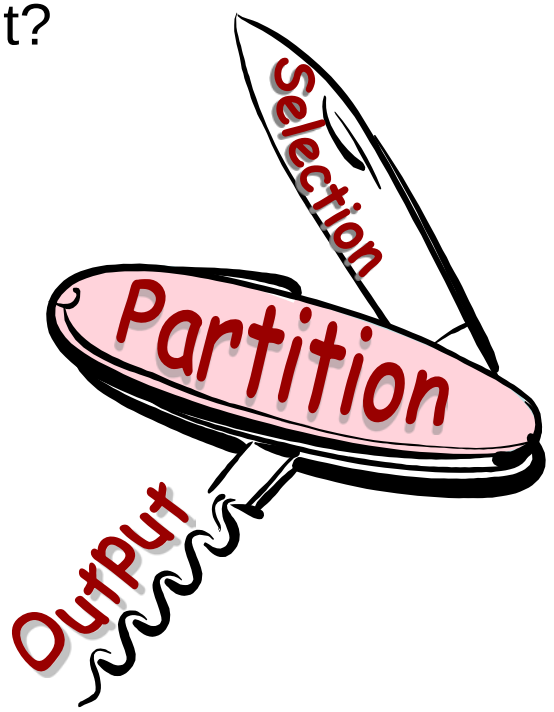


# `rwfilter` Command Structure

The rwfilter command requires three basic parts:

- selection criteria or input criteria (which files are input?)
  - repository: class, sensor, type, start/end date/hour
- Partition (which records pass my criteria? Which fail?)
  - filter options: Which flows do I really want?
- output options

Partitioning is the most complex part.



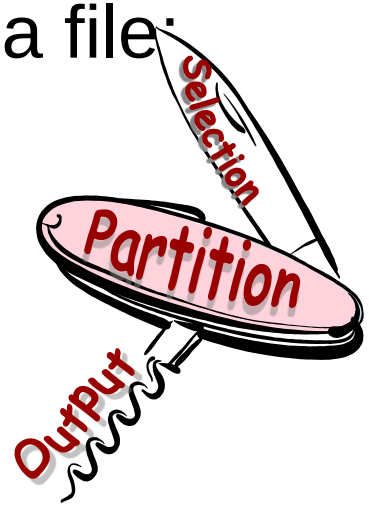
# Selection Criteria

These options control access to repository files:

- **`--start-date=2007/10/03:00`**
- **`--end-date=2007/10/03T03`** (ISO format)
- **`--sensor=S0`**
- **`--class=all`**
- **`--type=in`**

Alternatively, use input criteria for a pipe or a file:

- **`myfile.raw`**
- **`--input-pipe=stdin`**
- useful for chaining filters through stdin/stdout



# --start-date and --end-date

| | | --start-date | | |
|---|---|---|---|---|
| | | **Hour** | **Day** | **None** |
| **--end-date** | **Hour** | Hours in explicit range | Ignore end-date hour. Whole days. | Error |
| | **Day** | End-hour is the same as start-hour. #hours = 1, 25, 49, … | Whole days. | Error |
| | **None** | 1 hour | 1 day | Current day to present time. |

# How Many Files are Selected?

#Files = Sensors
x Types
x Hours
– missing files

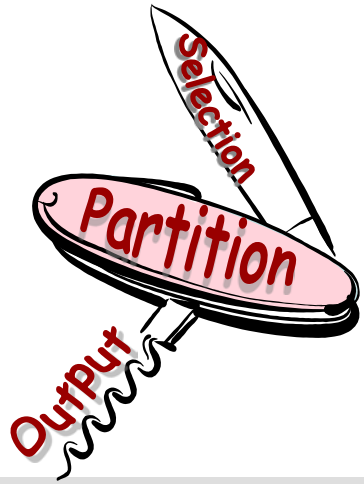
# `rwfilter` Partitioning Parameters - 1

Splits flows into “pass” and “fail” groups

Lots of options

- **`saddress, daddress, any-address, not-*, next-hop-id`**
- **`sport, dport, aport`**
- **`protocol`**
- **`bytes, packets, bytes-per`**
- **`stime, etime, active-time, duration`**
- **`tcp-flags, flags-all, flags-init`**
- **`sipset, not-sipset, dipset, not-dipset`**

Frequently expanding options



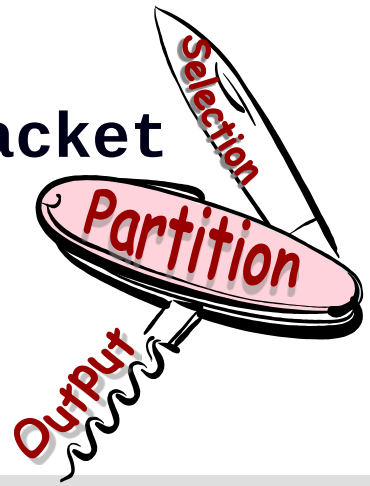
# Flow Partitioning Criteria: IP Data

Pass records based on IP fields; one is required.

- **`--[not-]saddress`**, **`--[not-]daddress`**: wildcard like 12.5,7,9.2-250.x or block notation like 12.5.2.0/24
- **`--protocol`**: IP protocol
- **`--sport`**, **`--dport`**, **`--aport`**: TCP, UDP ports (caveat: ICMP)
- **`--tcp-flags=SAF; --flags-all=S/SAFR; --fin-flag=1;`** ...
- **`--icmp-type`**, **`--icmp-code`**
- **`--bytes`**, **`--packets`**, **`--bytes-per-packet`**

At least one criterion is required.

- Use **`--proto=0-`** to pass all.



# What Is This? #4

```
rwfilter --type=in \
--start=2010/12/08:00 --end=2010/12/08:07 \
--daddress=71.55.0.0/16 --print-volume-stat
```

```
      |      Recs|     Packets|         Bytes| Files|
Total|  10588603|    13582511|    1756192286|     8|
 Pass|     29022|      788884|     627291737|      |
 Fail|  10559581|    12793627|    1128900549|      |
```

# Rwfilter exercise

```
rwfilter --sensor=S0 --type=in \
  --start-d=2009/04/21:00 --proto=0- \
  --pass-dest=T2100.rw --max-pass=20
ls -l T2100.rw
rwfileinfo T2100.rw  # look at format(id) and
                     # at count-records
hexdump –C T2100.rw  # any readable text?
rwcut --fields=1-6 T2100.rw
```

# Rwfilter exercise continued

```
rwfilter --sensor=S0 --type=in \
  --start-d=2009/04/21:00 --proto=0- \
  --pass-d=stdout --max-pass=20 \
| rwcut --fields=1-6
```

# Blacklists, Whitelists, Books of Lists...

Too many addresses for the command line?

- spam block list
- malicious websites
- arbitrary list of any type of addresses

Create an IP set!

- individual IP address in dotted decimal or integer
- CIDR blocks, 192.168.0.0/16
- wildcards, 10.4,6.x.2-254

Use it directly within your filter commands.

- **`--sipset`**, **`--dipset`**, **`--anyset`**

# Set Tools

**`rwsetbuild`**: Create sets from text.

**`rwset`**: Create sets from binary flows.

**`rwsetcat`**: Print out an IP set into text.

**`rwsetmember`**: Test if IP is in given IP sets.

**`rwsettool`**: Perform set algebra (set, union, intersection) on multiple IP sets.

# What Is This? - #5

```
SiLK> more MSSP.txt
171.128.2.0/24
171.128.212.0/24
SiLK> rwsetbuild MSSP.txt MSSP.set
SiLK> rwfilter --start=2010/12/8 --anyset=MSSP.set \
>  --pass=MSSP.rw --print-vol
      |        Recs|     Packets|         Bytes|Files|
Total|    30767188|    81382782|    35478407950|   48|
 Pass|    26678669|    31743084|     1464964676|     |
 Fail|     4088519|    49639698|    34013443274|     |

SiLK> rwset --sip-file=MSSPsource.set MSSP.rw
SiLK> rwsettool --intersect MSSP.set MSSPsource.set \
>   --output=activeMSSP.set
SiLK> rwsetcat --count-ips activeMSSP.set
22
```

# What Is This? - #6

```
SiLK> rwfilter --type=out --start=2010/12/08 \
> --proto=0-255 --pass=stdout \
> | rwset --sip-file=outIPs.set
SiLK> rwsetcat --network-structure=24 outIPs.set
     71.55.40.0/24| 246
  149.249.114.0/24| 256
   155.208.66.0/24| 256
   177.71.129.0/24| 80
   177.249.19.0/24| 256
   177.252.24.0/24| 256
  177.252.202.0/24| 256
```

# Exercise

Make a set-file of addresses of all actual inside hosts.

Should we examine incoming or outgoing traffic?

Make a set-file of all outside addresses.

Can you make both sets with one command?

# Exercise solution

```
rwfilter --sensor=S0 --type=out,outweb \
    --start-d=2009/04/21 --end=2009/04/23 \
    --proto=0- --pass=stdout \
| rwset --sip-file=insidehosts.set \
         --dip-file=outsidehosts.set
```

# Exercise

Examine the two set-files.

# Exercise solution

```
ls –l insidehosts.set
rwfileinfo insidehosts.set
rwsetcat insidehosts.set

ls –l outsidehosts.set
rwfileinfo outsidehosts.set
rwsetcat outsidehosts.set | less
```

# Exercise

Which /24 networks are on the inside?

Which /24 networks are on the outside?

# Exercise solution

```
rwsetcat --network-struc=24 insidehosts.set
rwsetcat --network-struc=24 outsidehosts.set
```

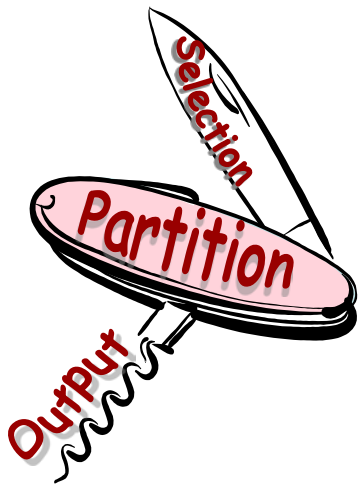
# Flow Partitioning Criteria: Time

Start-date and end-date choose repository files but do not look at the actual flow records.

- **`--stime`**, **`--etime`**: choose flows which start (or end) within a time range
- **`--active-time`**: flows active in a time range
- time format: YYYY/MM/DD:HH:MM:SS.mmm
  examples: 2009/12/16:01:14:30.043 or 2009/12/16:01:14
- time range format: [Time]-[Time]

Duration

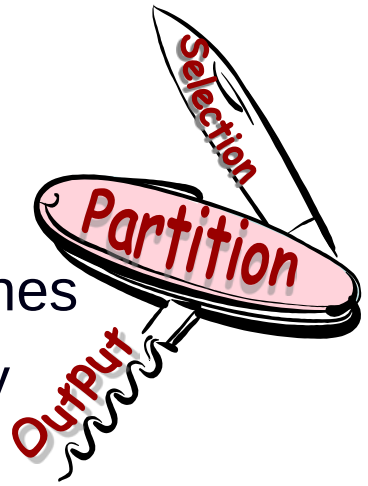
- **`--duration=1-10`**: number of seconds the flow was active



# Flow Partitioning Criteria: Advanced

Tend to use these as you gain experience:

- **`--max-pass`**: limit the number of records passed
- **`--tuple-file`**: specific combinations of addr, port, proto
- **`--scc`**, **`--dcc`**: country codes
- **`--pmap`**: prefix map
- **`--python-exp`**: use an expression
- **`--python-file`**: run a script to create new switches
- **`--dynamic-library`**: dynamically loaded library



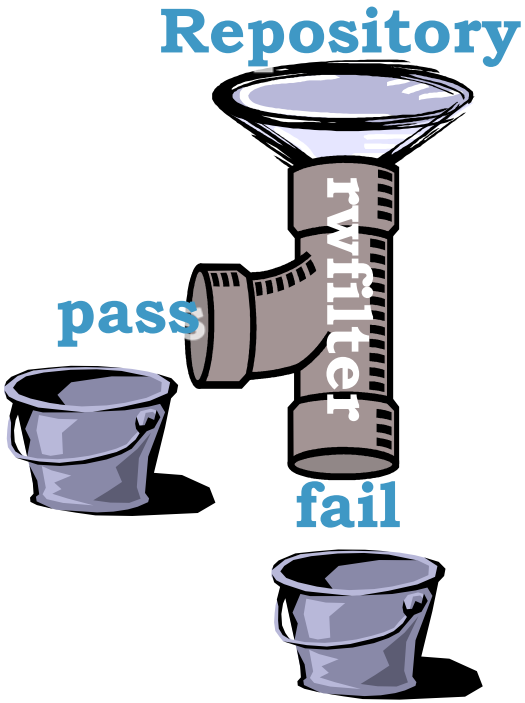
# Output Criteria

rwfilter leaves the flows in binary (compact) form.

- **`--pass`**, **`--fail`**: direct the flows to a file or a pipe
- **`--all`**: destination for everything pulled from the repository
- One output is required but more than one can be used (no screen allowed).

Other useful output

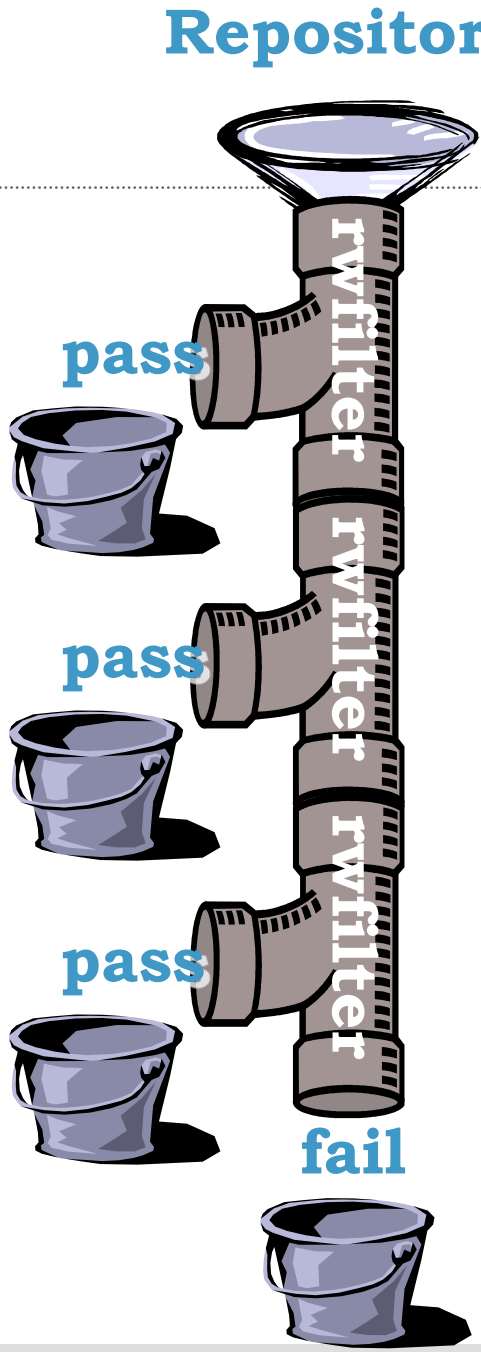
- **`--print-filenames`**, **`--print-missing-files`**
- **`--print-statistics`** or **`--print-volume-statistics`**



# Chaining Filters

It is often very efficient to chain rwfilter commands together:

- Use **`--pass`** and **`--fail`** to segregate bins.
- Use **`--all`**, so you only pull from the repository once.



# What Is This? #7

```
rwfilter \
      --start-date=2010/12/08 \
      --type=outweb \
      --bytes=100000- \
      --pass=stdout \
| rwfilter \
      --input-pipe=stdin \
      --duration=60- \
      --pass=long-http.rw \
      --fail=short-http.rw
```

# Other `rwfilter` Parameters

**`--help`**: lists the available rwfilter parameters

**`--dry-run`**: tests the command (useful for scripting)

**`--version`**: tells how rwfilter is configured

**`--ip-version`**: filters for ipv4 or ipv6 data (if configured)

**`--threads`**: uses multiple threads to filter

# Tips with `rwfilter`

Narrow time, type, and sensor as much as possible (fewer records to check).

Include as many partitioning parameters as possible (easy to be vague and get too much data).

Can do multiple queries and merge results

Can do further filtering to narrow results

Iterative exploration

# Example Typos

| | |
|---|---|
| --port=<br>--destport=<br>--sip= or --dip= | No such keywords |
| --saddress=danset.set | Needs value not filename |
| --start-date=2006/06/12--end-date | Spaces needed |
| --start-date = 2006/06/12 | No spaces around equals |
| start-date=2006/06/12 | Need dashes |
| ---start-date=2006/06/12 | Only two dashes |
| --start-date=2005/11/04:06:00:00<br>--end-date=2005/05/21:17:59:59 | Only down to hour |

# SiLK Commandments

1. **Thou shalt use Sets instead of using several rwfilter commands to pull data for multiple IP addresses**
2. **Thou shalt store intermediate data on local disks, not network disks.**
3. **Thou shalt make initial pulls from the repository, store the results in a file, and work on the file from then on. The repository is slower than processing a single file.**
4. **Thou shalt work in binary for as long as possible. ASCII representations are much larger and slower than the binary representations of SiLK data.**
5. **Thou shalt filter no more than a week of traffic at a time. The filter runs for excessive length of time otherwise.**
6. **Thou shalt only run a few rwfilter commands at once.**
7. **Thou shalt specify the type of traffic to filter. Defaults work in mysterious ways.**
8. **Thou shalt appropriately label all output.**
9. **Thou shalt check that SiLK does not provide a feature before building your own.**

# Basic SiLK Counting Tools: `rwcount`, `rwstats`, `rwuniq` (1)

"Count [volume] by [key field] and print [summary]"

- basic bandwidth study:
  - "Count bytes by hour and print the results."
- top 10 talkers list:
  - "Count bytes by source IP and print the 10 highest IPs."
- user profile:
  - "Count records by dIP-dPort pair and print the results."
- potential scanners:
  - "Count unique dIPs by sIP and print the sources that contacted more than 100 destinations."

# Basic SiLK Counting Tools: `rwcount`, `rwstats`, `rwuniq` (2)

**`rwcount`**: count volume across time

**`rwstats`**: count volume across IP, port, or protocol and create descriptive statistics

**`rwuniq`**: count volume across any combination of SiLK fields

"Volume" = {Records, Bytes, Packets} and a few others—measure

"Key field" = SiLK fields to be measured and listed

Each tool reads raw binary flow as input.

# Calling `rwcount`

- count records, bytes, and packets by time and print results
- fast, easy way of summarizing volumes as a time series
- great for simple bandwidth studies

# `rwcount` Counting Options (1)

| Key | Volume | Summary |
|---|---|---|
| **`--bin-size=S`** | no options | **`--skip-zeroes`** |
| **`--load-scheme=N`** | (always Records, Bytes, Packets) | **`--start-epoch`** **`--end-epoch`** |

Key field is *always* time.

- Specify **`--bin-size`** in seconds.
- Use **`--load-scheme`** to select a method for counting records whose sTime, eTime straddle several bins.

Volume is *always* three columns: Records, Bytes, Packets.

# `rwcount` Counting Options (2)

| Key | Volume | Summary |
|---|---|---|
| **`--bin-size=S`** | no options | **`--skip-zeroes`** |
| **`--load-scheme=N`** | (always Records, Bytes, Packets) | **`--start-epoch`** **`--end-epoch`** |

Limited summary options for printing output

- Include/exclude time bins with count = 0.
- Specify a minimum start and/or maximum end time.

# What Is This? #8

```
SiLK> rwcount MSSP.rw --bin-size=3600
                Date|    Records|       Bytes|    Packets|
2010/12/08T00:00:00| 1351571.66| 73807086.40| 1606313.61|
2010/12/08T01:00:00| 1002012.43| 54451440.59| 1185143.62|
2010/12/08T02:00:00| 1402404.61| 77691865.26| 1675282.27|
2010/12/08T03:00:00| 1259973.65| 68575249.90| 1491393.08|
2010/12/08T04:00:00|  939313.56| 51410968.24| 1118584.81|
2010/12/08T05:00:00|  459564.75| 80862273.32| 1742058.62|
2010/12/08T06:00:00| 1280651.23| 69881126.41| 1519435.24|
...
```

# Demo

Time series for all outgoing traffic on S0:

```
rwfilter --sensor=S0 --type=out,outweb \
    --start=2009/04/21 --end=2009/04/23 \
    --proto=0- --pass=stdout \
| rwcount --bin-size=3600
```

# Exercise

Produce a time-series with 30-minute intervals, analyzing incoming ICMP traffic collected at sensor S1 on April 21, 2009.

# Exercise solution

```
rwfilter --sensor=S0 --type=in,inicmp \
    --start=2009/04/21 --proto=1 \
    --pass=stdout \
| rwcount --bin-size=1800
```

# Calling `rwstats`

## `rwstats --overall-stats`

- Descriptive statistics on byte and packet counts by record
- See “man rwstats” for details.

## `rwstats --fields=KEY --value=VOLUME --count=N or --threshold=N or --percentage=N [--top or --bottom]`

- Choose one or two key fields.
- Count one of records, bytes, or packets.
- Great for Top-N lists and count thresholds
- (standard output formatting options – see “man rwstats”)

# `rwstats` Counting Options

| Key | Volume | Summary |
|---|---|---|
| **`--fields={`**<br>**`sport, dport, icmp, protocol, sip, dip, sport, dport, …}`** | **`--value={`**<br>**`records | bytes | packets | sip-distinct | dip-distinct}`** | **`--count=N`**<br>**`--threshold=N`**<br>**`--percentage=N`**<br><br>**`--top`**<br>**`--bottom`** |

Use **`--top`** or **`--bottom`** to specify

top N or bottom N keys (with --count)

- volume greater or less than N (with **`--threshold`**, **`--percentage`**)

# What Is This? #9

```
SiLK> rwfilter outtraffic.rw \
> --stime=2010/12/08:18:00:00-2010/12/08:18:59:59 \
> --pass=stdout \
> | rwstats --fields=sip --values=bytes --count=10
INPUT: 1085277 Records for 1104 Bins and 4224086177 Total Bytes
OUTPUT: Top 10 Bins by Bytes
                sIP|              Bytes|    %Bytes|   cumul_%|
        71.55.40.62|         1754767148| 41.541935| 41.541935|
       71.55.40.169|         1192063164| 28.220617| 69.762552|
       71.55.40.179|          331310772|  7.843372| 77.605923|
       71.55.40.204|          170966278|  4.047415| 81.653338|
     177.249.19.217|          122975880|  2.911301| 84.564639|
        71.55.40.72|          110726717|  2.621318| 87.185957|
       71.55.40.200|          101593627|  2.405103| 89.591060|
     177.71.129.255|           40166574|  0.950894| 90.541954|
        71.55.40.91|           35316554|  0.836076| 91.378030|
     149.249.114.204|           26634602|  0.630541| 92.008571|
```

# Exercise

What are the top 10 incoming [IP] protocols on April 22, 2009, collected on S0?

# Exercise solution

```
rwfilter --sensor=S0 --type=in,inweb \
   --start=2009/04/22 --prot=0- --pass=stdout \
| rwstats --fields=protocol --value=rec --count=10
```

# Exercise 2

Top 10 inside hosts according to how many outside hosts they communicate with.

Use **--value=dip-distinct**

# Exercise 2 answer

```
rwfilter --sensor=S0 --type=out,outweb --proto=0- \
    --start-d=2009/4/22 --pass=stdout \
| rwstats --fields=sip --value=dip-distinct --count=10
```

# Calling `rwuniq`

**`rwuniq --fields=KEYS --value=VOLUME`**

- most flexible of the counting tools
  - —does not support Top-N key sorting
  - —does support multiple key queries and multiple volume summaries
- runs much faster on input sorted by key fields
  - —Use **`--presorted-input`** when this is the case.
- (standard output formatting options – see "**man rwuniq**")

# `rwuniq` Counting Options

| Key | Volume | Summary |
|---|---|---|
| `--fields=KEYS`<br><br>`--bin-time=SECS` | `--value={ flows \| bytes \| packets \| sip-distinct \| dip-distinct \| stime \| etime}…` | `--sort-output`<br>`--VOLUME=MIN`<br>`--VOLUME=MIN-MAX` |

**KEYS** is any valid specification of SiLK fields:

- `rwuniq --fields=sIP,sPort,sTime --bin-time=60`
- `rwuniq --fields=1-5`

Choose *any* combination of volumes, or `--all-counts` for all.

Use `--sort-output` to sort by *key*, not by volume (no Top-N lists).

# What Is This? #10

```
SiLK> rwfilter outtraffic.rw \
> --stime=2010/12/08:18:00:00-2010/12/08:18:59:59 \
> --saddress=71.55.40.62 --pass=stdout \
>| rwuniq --fields=dip,sport --all-counts --sort-output
```

```
           dIP|sPort |     Bytes|Packets| Records|     sTime-Earliest|       eTime-Latest|
 12.113.41.190|   80 |     12782|     20|       4|2010/12/08T18:42:51|2010/12/08T18:58:49|
30.182.228.143|   80 | 203907933| 143611|       2|2010/12/08T18:53:59|2010/12/08T19:01:47|
 37.153.24.229|   80 | 205628625| 144829|       2|2010/12/08T18:29:11|2010/12/08T18:42:51|
 82.180.203.87|   80 | 213013145| 150896|      92|2010/12/08T18:06:36|2010/12/08T18:32:33|
82.180.203.197|   80 |       800|      8|       2|2010/12/08T18:43:30|2010/12/08T18:43:30|
88.124.166.233|   80 | 223930369| 158276|      97|2010/12/08T18:08:55|2010/12/08T18:32:25|
88.124.166.233|  443 |    509285|    732|      43|2010/12/08T18:06:57|2010/12/08T18:51:11|
94.239.226.247|   80 | 124833037|  96047|       3|2010/12/08T18:25:22|2010/12/08T19:21:34|
  109.95.61.80|   80 |   8467397|   6325|      90|2010/12/08T18:08:59|2010/12/08T18:10:09|
  139.65.186.4|   80 | 204123360| 143794|       3|2010/12/08T18:19:48|2010/12/08T18:26:36|
139.177.10.136|   80 | 407978375| 287354|       6|2010/12/08T18:20:03|2010/12/08T19:01:30|
198.237.16.172|   80 | 159066748| 112025|       1|2010/12/08T18:18:43|2010/12/08T18:46:55|
219.149.72.154| 1024 |        44|      1|       1|2010/12/08T18:50:40|2010/12/08T18:50:40|
249.216.88.172|   80 |        88|      2|       2|2010/12/08T18:44:42|2010/12/08T18:44:47|
250.211.100.88|   80 |   3295160|   2492|      42|2010/12/08T18:47:50|2010/12/08T18:58:53|
```

# What Is This? #11

```
SiLK> rwuniq outtraffic.rw --fields=dip \
> --values=sip-distinct,records,bytes --sip-distinct=400- \
> --sort-output
              dIP|sIP-Distin|     Bytes| Records|
    13.220.28.183|         512|     20480|     512|
     171.128.2.27|         448|  19069280|  476732|
    171.128.2.179|         448| 139501200| 3487530|
   171.128.212.14|         448| 139467440| 3486686|
  171.128.212.124|         448| 127664480| 3191612|
  171.128.212.127|         448|  66611560| 1665289|
  171.128.212.188|         448| 139467680| 3486692|
  171.128.212.228|         448| 139393160| 3484829|
  245.225.153.120|         763|     30520|     763|
  245.238.193.102|        1339|    179480|    4487|
```

# Basic SiLK Tools: `rwsort`

Why sort flow records?

- Records are recorded as received, not necessarily in time order.
- Analysis often requires finding outliers.
- You can also sort on other fields such as IP address or port to easily find scanning patterns.
- It allows analysts to find behavior such as beaconing or the start of traffic flooding.

# `rwsort` Options

**`--fields`** (same as rwcut) is required.

input, output (stdin/stdout are defaults.)

For improved sorts, specify a buffer size.

For large sorts, specify a temporary directory.
Temporary files stored in /tmp by default

```
rwsort myfile.raw --fields=stime,sip \
   --temp-dir=. >newfile.raw

rwsort --fields=sip,sport,dport myfile.raw    \
| rwuniq --fields=sip,sport,dport --presorted \
         --dip-distinct
```

# I Only Believe What I See

You'll be tempted to work with text-based records.

- It's easy to see the results and post-process with other tools (e.g., Perl, awk, sed, sort).
- It takes a lot of space, and it's ***much, much*** slower.

Guiding principle: Keep flows in binary format as long as possible.

# What Is This? #12

```
rwfilter --type=out --
   start=2010/12/08 \
   --aport=22 --pass=ssh.rw

rwfilter --dport=22 ssh.rw \
   --pass=stdout | rwcut

rwfilter --sport=22 ssh.rw \
   --pass=stdout | rwcut
```

# Outline

Introduction: SiLK

Network flow

Basic SiLK tools

**Advanced SiLK tools**

Summary

# PySiLK—Using SiLK with Python

- PySiLK—an extension to Python
- Allows Python to manipulate SiLK's data files
- Uses the "silk" python module, from SEI CERT.

# PySiLK example

```
#! /bin/env python
import silk
myfile = silk.SilkFile("MyFlows.rw", silk.READ)
for rec in myfile:
        if rec.sport < 2500 and rec.sport == rec.dport:
        print  ("%d %s %s %s" %
                (rec.sport, rec.stime, rec.sip, rec.dip))
```

# Alternatives to PySiLK

- SiLK tools
  - Not as flexible criteria as Python.
  - Could use tuple files
    - Must be maintained
    - Aren’t self-contained with logic
    - Large tuple files run slower than Python.
- Text processing with Perl, C, or Java
  - Create text with rwcut delimited without titles
  - Convert ports back to integers
  - Dealing with dates, times, or addresses difficult

# Modified example of PySilk

- Summarize the selection as a count by port
- Just keep a Python dictionary
  - Key = port number
  - Value = count

# PySiLK advantages

- Speeds both programming and processing
  - Keeps data in binary, unlike Perl & C
    - No parsing text
  - Built-in conversions of objects to strings
  - Full power of Python
- Good for:
  - Stateful filters and output options
  - Integrate SiLK with other data types
  - Complex or branching filter rules
  - Custom key fields and aggregators for rwcut, rwsort

# Outline

Introduction: SiLK

Network flow

Basic SiLK tools

Advanced SiLK tools

**Summary**

# Furthering Your SiLK Analysis Skills (1)

Each tool has a **`--help`** option.

SiLK Reference Guide

SiLK Analysts' Handbook

- Both available at the SiLK tools website http://tools.netsa.cert.org

Email support

- silk-help@cert.org

# Furthering Your SiLK Analysis Skills (2)

Tool tips

- SiLK Tooltips link on http://tools.netsa.cert.org

Flow analysis research and advanced techniques

- http://www.cert.org/flocon
- http://www.cert.org/netsa

# Questions?

# Contact Information

Ron Bandes — **rbandes@cert.org**
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA