

Notices

Copyright 2015 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon[®], CERT[®], CERT Coordination Center[®] and FloCon[®] are registered marks of Carnegie Mellon University.

DM-0003146



Housekeeping

Restrooms on past registration desk

Breaks and lunch in same location

Follow exit signs in case of emergency

Ask questions any time, don't be shy

Course 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.
- *Analyze traffic and profile a network using basic SiLK commands.*

Agenda



I. Network flow

- I. What is network flow
- II. Interpreting flow records
- III. SiLK commands

II. Basic SiLK tools

- I. SiLK Records, Files, and the Repository
- II. Analysis Tools and Categorization
- III. IP Sets

Carnegie Mellon University



Schedule

9:00 AM	SiLK Part 1 of 4	Basics of Network Flow and Unix Commands
10:45 AM	Break	
11:00 AM	SiLK Part 2 of 4	Basics of SiLK
12:30 PM	Lunch	
1:30 PM	SiLK Part 3 of 4	Network flow analysis with SiLK
3:15 PM	Break	
3:30 PM	SiLK Part 4 of 4	More network flow analysis with SiLK
5:00 PM	Adjourn	
6:00 PM	Welcome Reception	Near reception

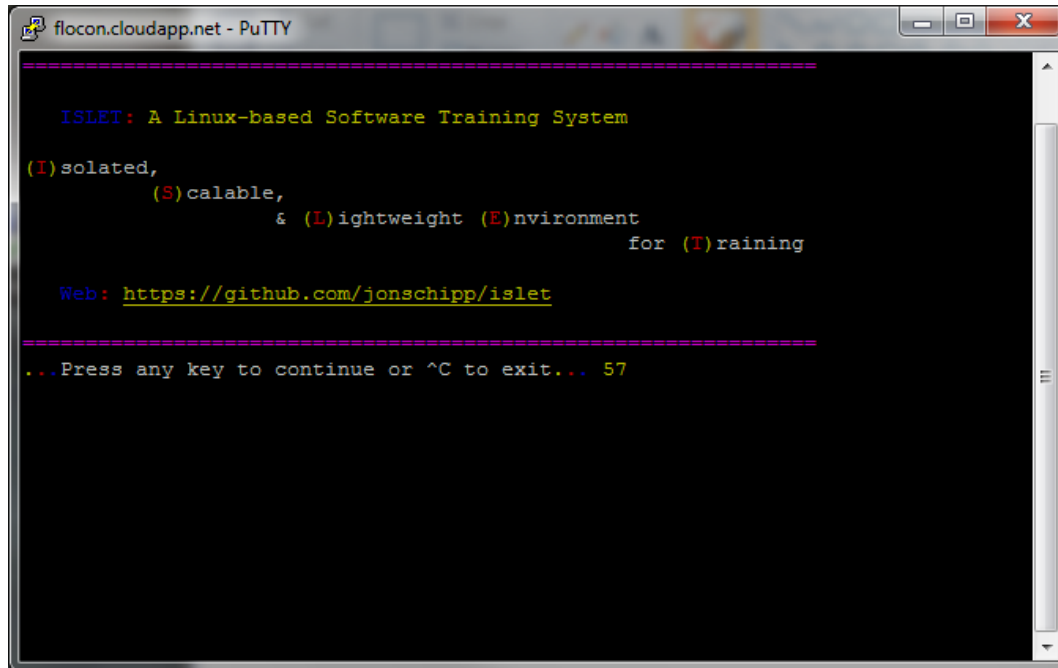
Setting up your analysis environment

- SSH to

flocon.cloudapp.net

- Username: **demo**
- Password: **flocon2016**

Analysis environment continued



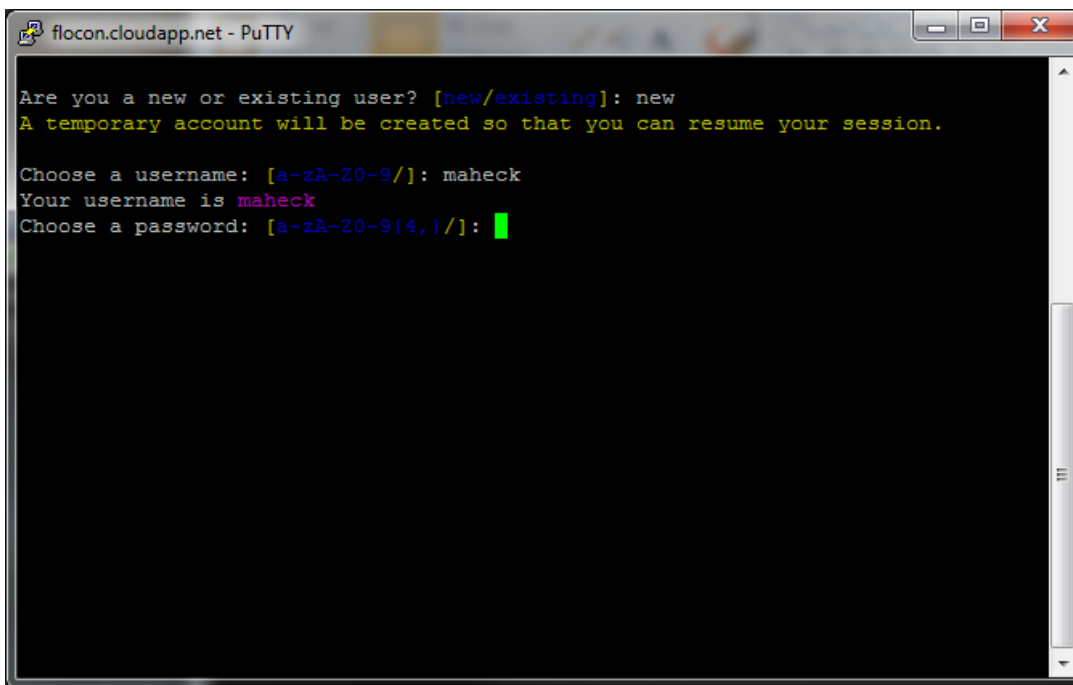
```
flocon.cloudapp.net - PuTTY

=====
ISLET: A Linux-based Software Training System
(I)solated,
    (S)calable,
        & (L)ightweight (E)nvironment
                                for (T)raining

Web: https://github.com/jonschipp/islet
=====
...Press any key to continue or ^C to exit... 57
```


Analysis environment – Account creation

- Create your own application account
- Remember your information!



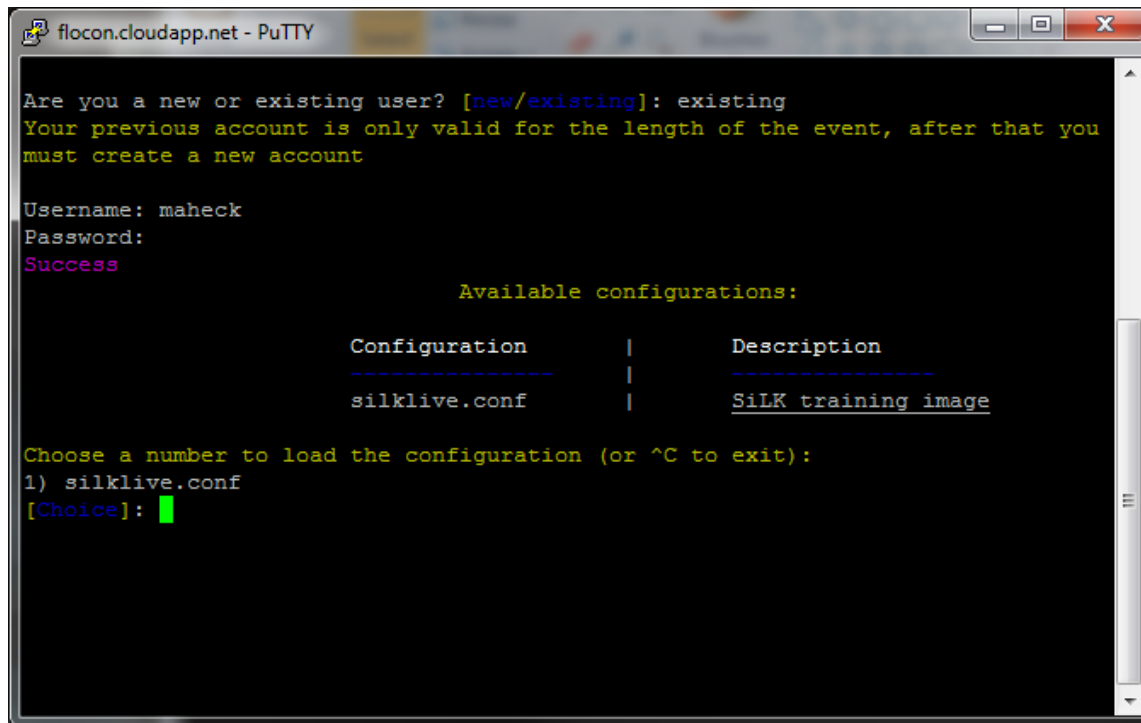
```
flocon.cloudapp.net - PuTTY
Are you a new or existing user? [new/existing]: new
A temporary account will be created so that you can resume your session.

Choose a username: [a-zA-Z0-9/]: maheck
Your username is maheck
Choose a password: [a-zA-Z0-9{4,}/]: █
```

Analysis environment – Account information

- Accounts last for 14 days
 - The service will be shutdown on 1/25/2016
- You are limited to 2 GB of Hard drive space
 - Exceeding this limit will cause your account to be wiped
- Do not store anything of value on the server!
 - All information will be wiped

Analysis environment – SiLK Training Image



```
flocon.cloudapp.net - PuTTY
Are you a new or existing user? [new/existing]: existing
Your previous account is only valid for the length of the event, after that you
must create a new account

Username: maheck
Password:
Success

          Available configurations:

Configuration | Description
-----|-----
silklive.conf | SiLK training image

Choose a number to load the configuration (or ^C to exit):
1) silklive.conf
[Choice]: █
```

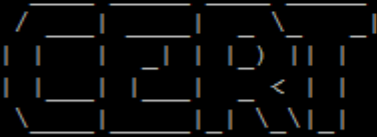
Analysis environment – At the Prompt

```
demo@silK-live: ~  
=====
```

Welcome to SiLK Configuration!

SiLK, the System for Internet-Level Knowledge, is a collection of traffic analysis tools developed by the CERT Network Situational Awareness Team (CERT NetSA) to facilitate security analysis of large networks. The SiLK tool suite supports the efficient collection, storage, and analysis of network flow data, enabling network security analysts to rapidly query large historical traffic data sets.

Web: <http://tools.netsa.cert.org>



A place to try out SiLK

```
=====
```

Enjoy yourself!

Run `rwsiteinfo` for information on repository
e.g. `$ rwsiteinfo --fields=sensor,describe-sensor,type:list`

```
demo@silK-live:~$ █
```

Exercise 0: *NIX

```
PS1='\W \!> ' # this is not permanent
export SILK_IPV6_POLICY=asv4
cd /data/bluered
ls -l silk.conf
less silk.conf # type "q" to exit from less
cd
```



Part I: Network Flow



Part I Lessons: Network Flow

1. What is Network Flow?
2. Interpreting Flow Records
3. Issuing SiLK Commands

FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics



Lesson I.1

What is Network Flow?

Lesson I.1 Learning Objective

Given a sequence of packets and some basic knowledge of packets, the learner will be able to identify the uniflows comprising the packets.



<http://www.iccwbo.org/News/Articles/2012/ICC-defends-freedom-of-expression-and-the-free-flow-of-information-online/>

What is Network Flow?

A log of all network activity; not a recording of all packets

A record of metadata from related packets

- similar to a phone bill (call detail record)

Content of messages is *not* recorded

- much, much more compact
 - longer retention
 - less processing
- increased privacy
- less impact from encryption

LATITUDE	LONGITUDE	DATE	TIME	NUMBER	NAME	DURATION
44.50880 N	73.18223 W	1/28/2008	0917	802-555-1234	Chittenden Bank	0:10:17
44.50880 N	73.18223 W	1/28/2008	0942	802-555-8673	Poopsie LaRue	0:01:03
44.50880 N	73.18223 W	1/28/2008	0945	802-555-9201	Hanley Strappman	0:05:32
44.27834 N	73.21263 W	1/29/2008	2205	802-555-7758	Verizon Voice Mail	0:01:13
44.27834 N	73.21263 W	1/29/2008	1532	802-555-4492	Widgets LLC	0:03:47
44.27834 N	73.21263 W	1/29/2008	2209	802-555-7758	Verizon Voice Mail	0:00:36
44.50880 N	73.18223 W	1/30/2008	0830	202-555-1818	British Embassy	0:18:12
44.27834 N	73.21263 W	1/30/2008	2208	802-555-7758	Verizon Voice Mail	0:00:53
44.27834 N	73.21263 W	1/30/2008	2211	802-555-8673	Poopsie LaRue	0:06:18
44.50880 N	73.18223 W	1/31/2008	0903	202-555-1843	British Embassy	0:03:21
44.50880 N	73.18223 W	1/31/2008	0908	416-555-9834	British Embassy	0:22:04
44.4143 N	73.03561 W	1/31/2008	1047	802-555-9201	Hanley Strappman	0:01:02
44.4143 N	73.03561 W	1/31/2008	1050	213-555-2761	M. Fendell	0:09:06
44.25295 N	72.58229 W	1/31/2008	1127	802-555-9201	Hanley Strappman	0:05:38

What SiLK Does

Investigation analysis

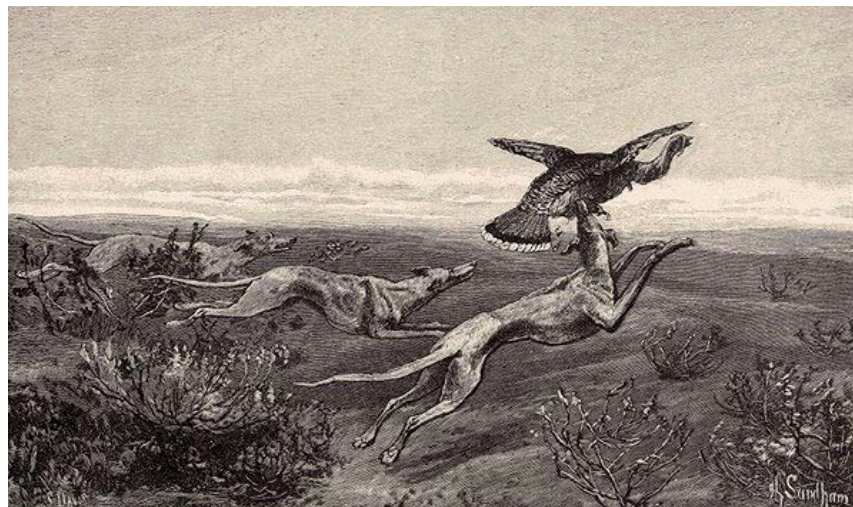
- most useful for analysing past network events
- may feed an automated report generator
- good for forensics (what happened **before** the incident?)

Descriptive analysis – profiling/categorizing

Directed analysis (hunt) – looking for specific malicious behavior

Exploratory analysis – looking for the unusual

Predictive Analysis



Did you ever wonder...

http://www.clipartpanda.com/clipart_images/pondering-clipart-9915834

What's on my network?

What happened before the event?

Where are policy violations occurring?

What are the most popular web servers?

By how much would volume be reduced with a blacklist?

Do my users browse to known infected web servers?

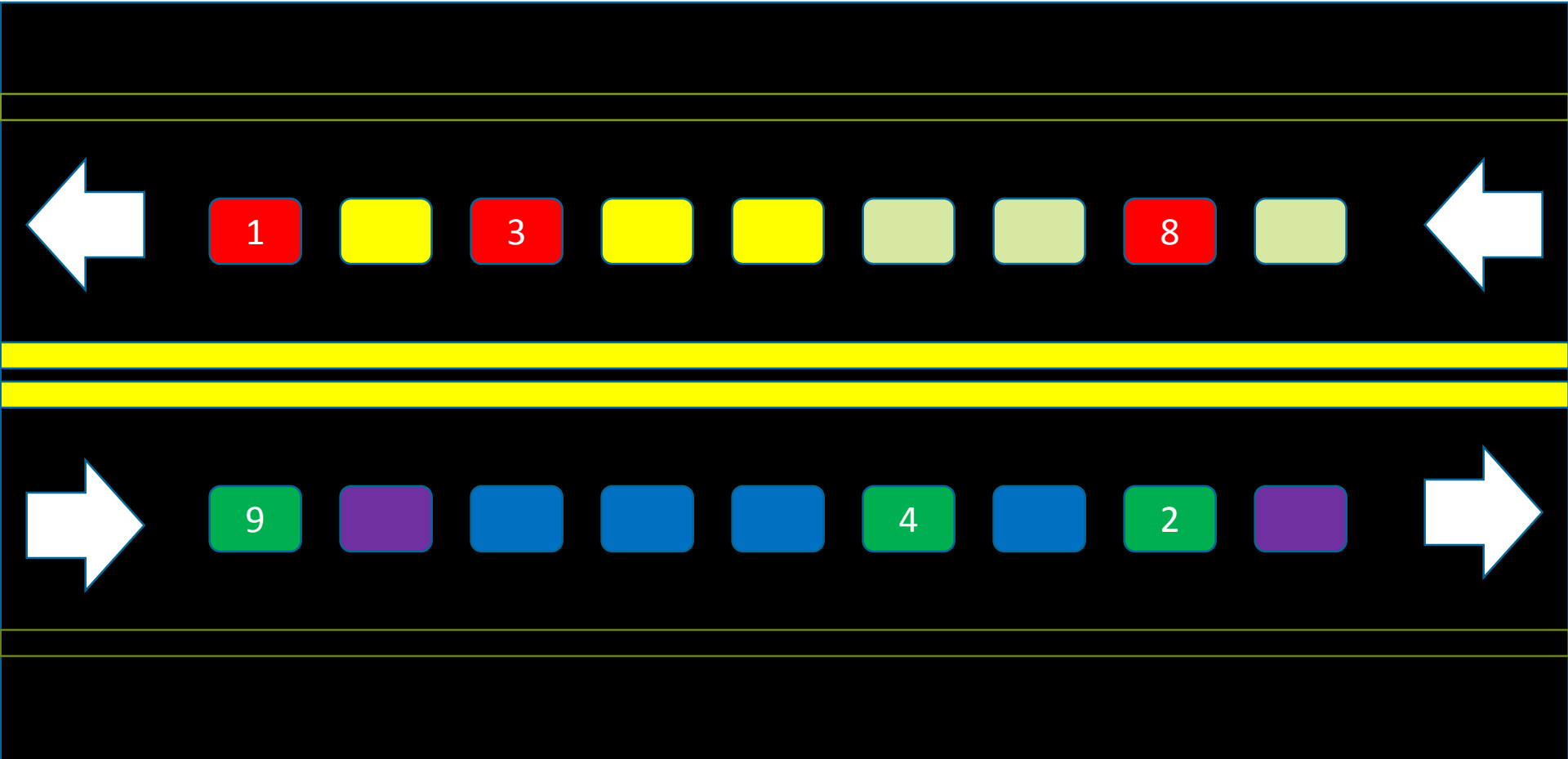
Do I have a spammer on my network?

When did my web server stop responding to queries?

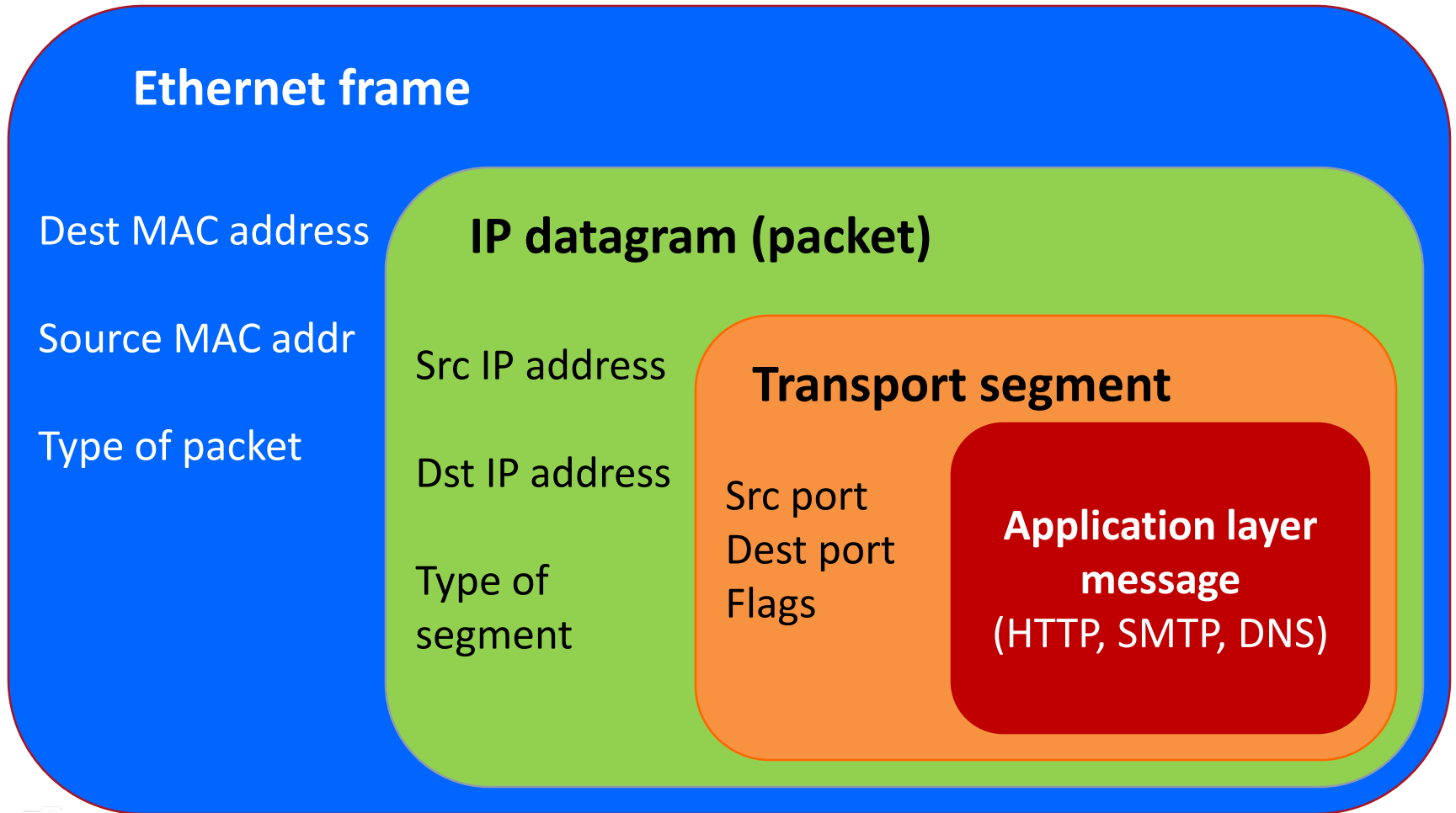
Who uses my public servers?



Unidirectional Flows (Uniflows)



Packet Encapsulation



Two TCP/IP Sockets Make a Connection

TCP/IP SOCKET

IP address: 10.0.0.1

L4 protocol: TCP

Ephemeral port #



**Client
Process**

TCP/IP SOCKET

IP address: 203.0.113.1

L4 protocol: TCP

Well-Known Port #



**Server
Process**



Connection

Network Flow versus NetFlow

Network Flow—a generic term for the summarization of packets related to the same flow or connection into a single record

NetFlowTM—a Cisco trademarked set of format specifications for storing network flow information in a digital record. Most common are versions 5 and 9.

IPFIX—a format specification from the IETF for flow records, an extension of Cisco NetFlow v9

SiLK—another set of format specifications for flow records and other related data, plus the tool suite to process that data

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

The screenshot shows the Wireshark interface with a packet capture of a DNS query and response. The packet list pane shows two packets:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.105	10.1.10.1	DNS	78	Standard query A www.mudynamics.com
2	0.348077	10.1.10.1	192.168.1.105	DNS	94	Standard query response A 69.55.232.156

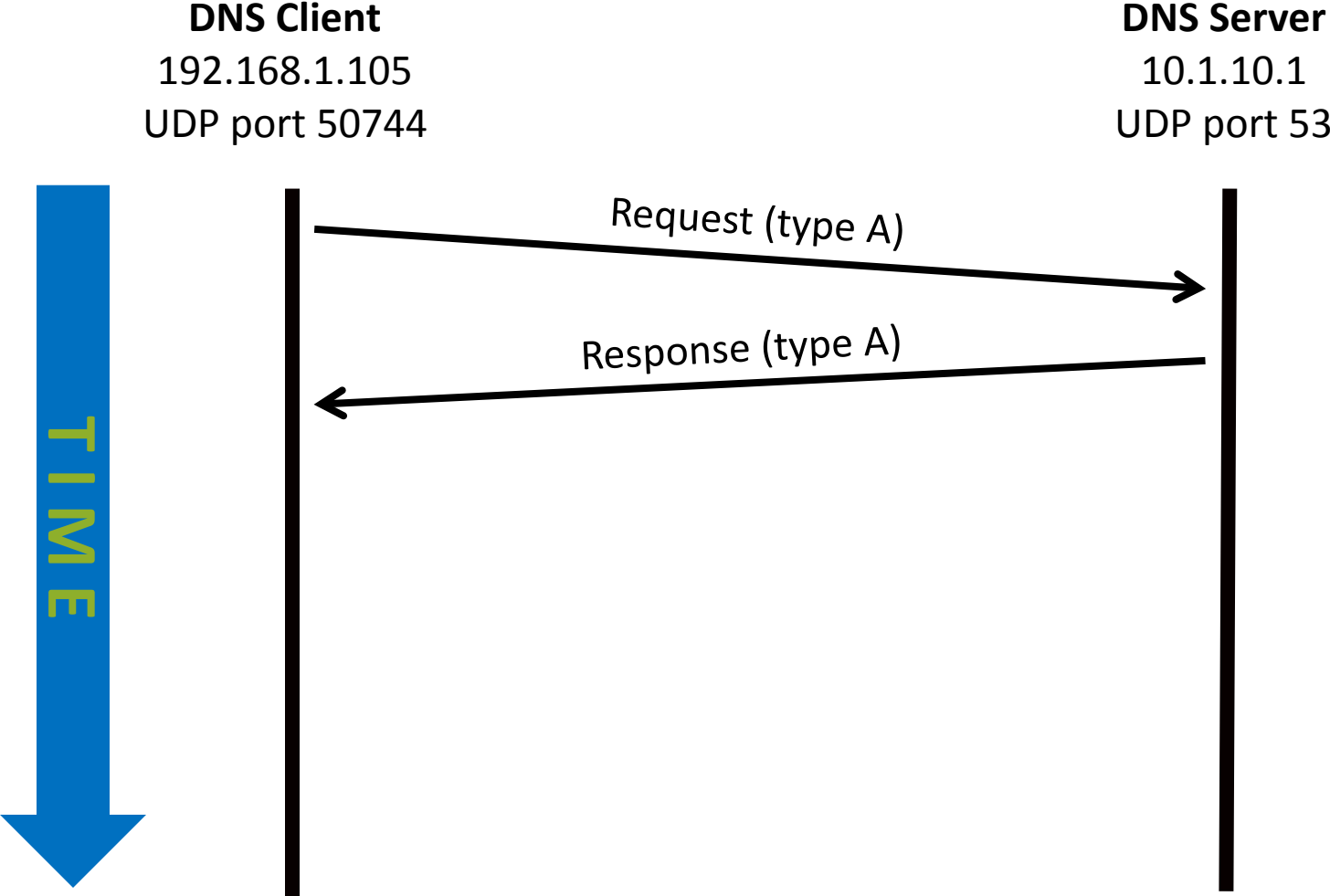
The packet details pane for the selected packet (Frame 2) shows the following layers:

- Ethernet II, Src: Cisco-Li_66:ae:1c (00:1a:70:66:ae:1c), Dst: AppleCom_d3:9a:b8 (00:19:e3:d3:9a:b8)
- Internet Protocol Version 4, Src: 10.1.10.1 (10.1.10.1), Dst: 192.168.1.105 (192.168.1.105)
- User Datagram Protocol, Src Port: domain (53), Dst Port: 50744 (50744)
- Domain Name System (response)

The packet bytes pane shows the raw data in hexadecimal and ASCII:

Offset	Hex	ASCII
0000	00 19 e3 d3 9a b8 00 1a 70 66 ae 1c 08 00 45 00 pf....E.
0010	00 50 05 91 00 00 3f 11 9f f9 0a 01 0a 01 c0 a8	.P....?.
0020	01 69 00 35 c6 38 00 3c 78 0d ea f9 81 80 00 01	.i.5.8.<x.....
0030	00 01 00 00 00 00 03 77 77 77 0a 6d 75 64 79 6ew ww.mudyn
0040	61 6d 69 63 73 03 63 6f 6d 00 00 01 00 01 c0 0c	amics.co m.....
0050	00 01 00 01 00 00 0e 10 00 04 45 37 e8 9cE7...

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

Lesson I.1 Summary



Flow records constitute a log of network activity.

Flow analysis can answer many questions without storing content.

Flow records are extremely compact.
Benefits are

- long retention
- faster processing
- reduced privacy concerns
- encryption is not an obstacle

SiLK uses unidirectional flows—uniflows.

Next Lesson

In lesson I.2, you will learn to interpret SiLK flow records and understand the nature of the associated network activity.



FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics



Lesson 1.2

Interpreting Flow Records

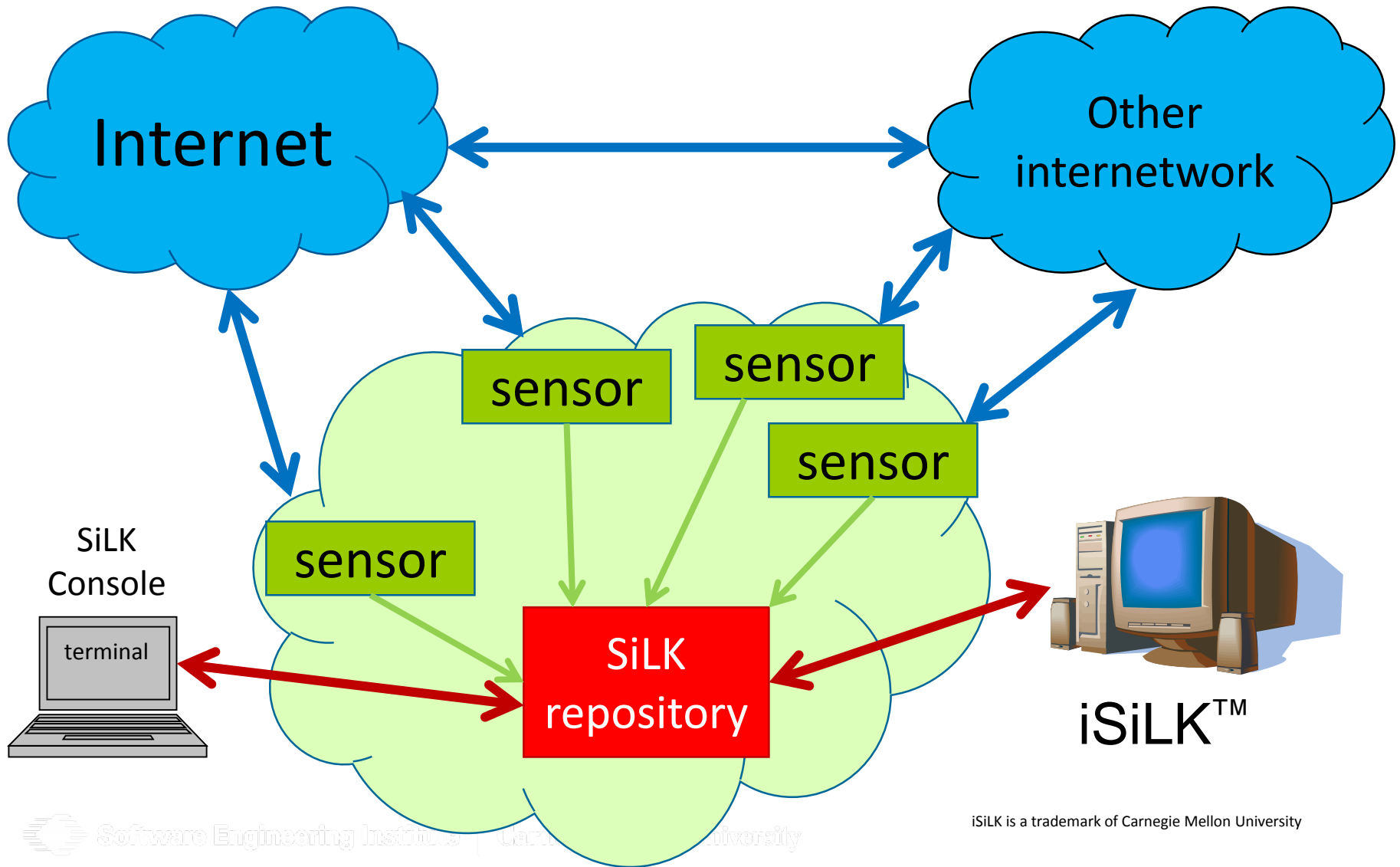


Lesson 1.2 Learning Objective

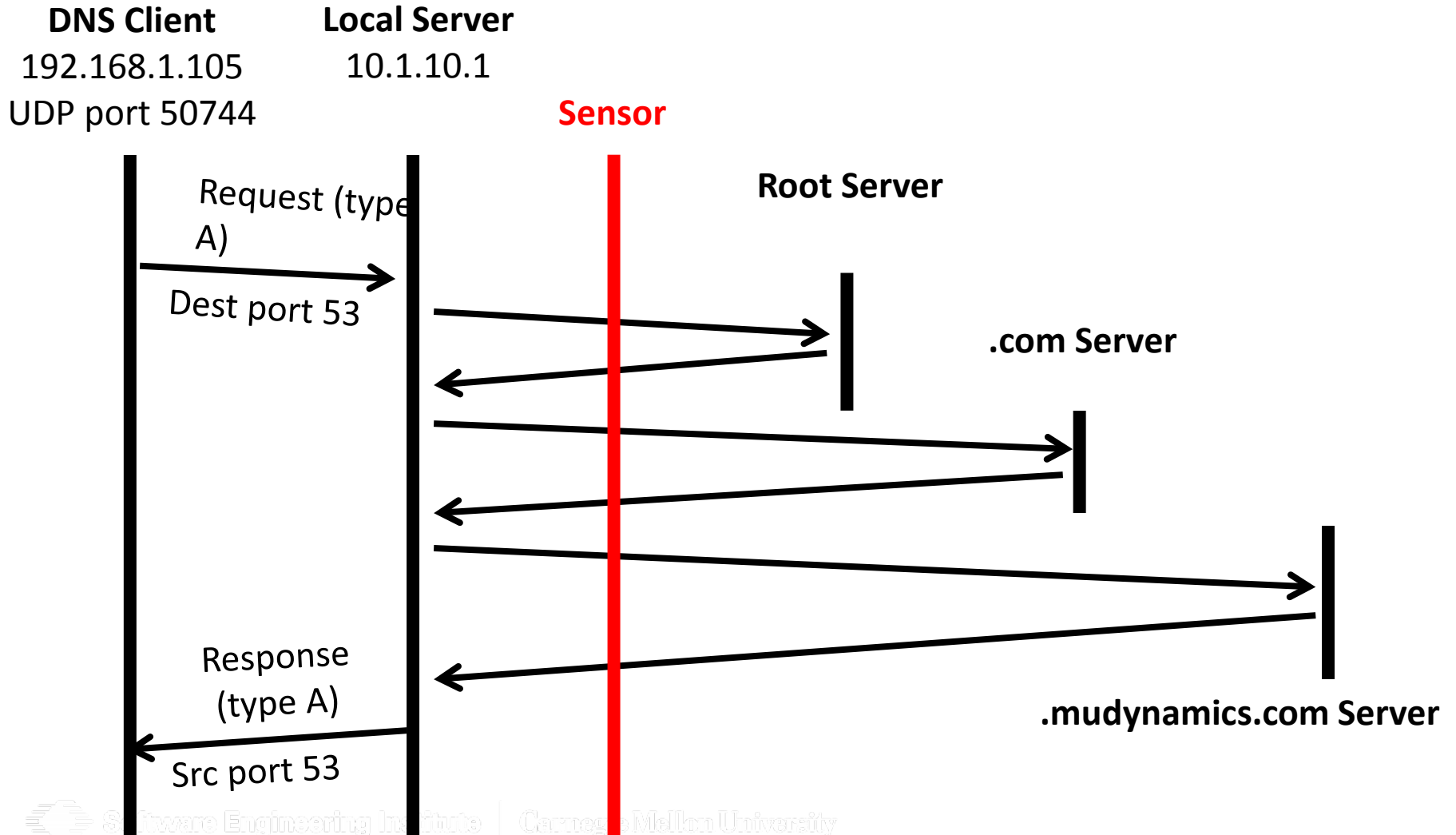
Given a series of uniflows and general knowledge of TCP/IP, the learner will be able to deduce and infer the nature of the network activity.



Network Monitoring



Realistic Sequence Diagram

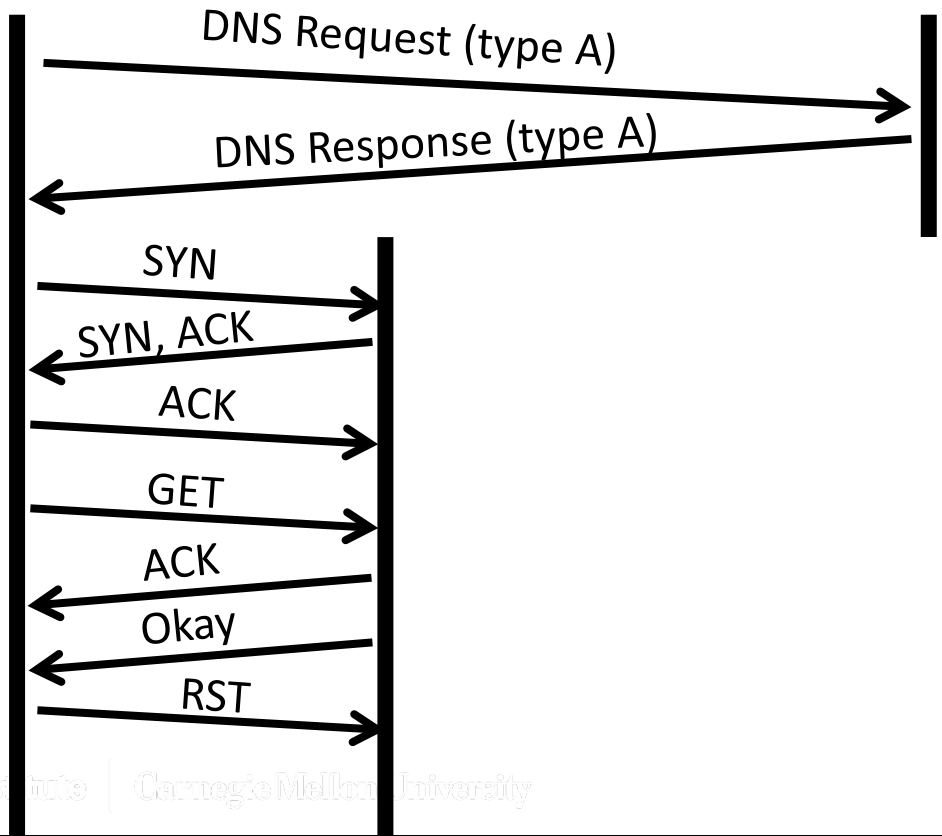


What is this? — 1

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

HTTP Client HTTP Server DNS Server
192.168.1.105 198.51.100.6 10.1.10.1



What Is This? — 2

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

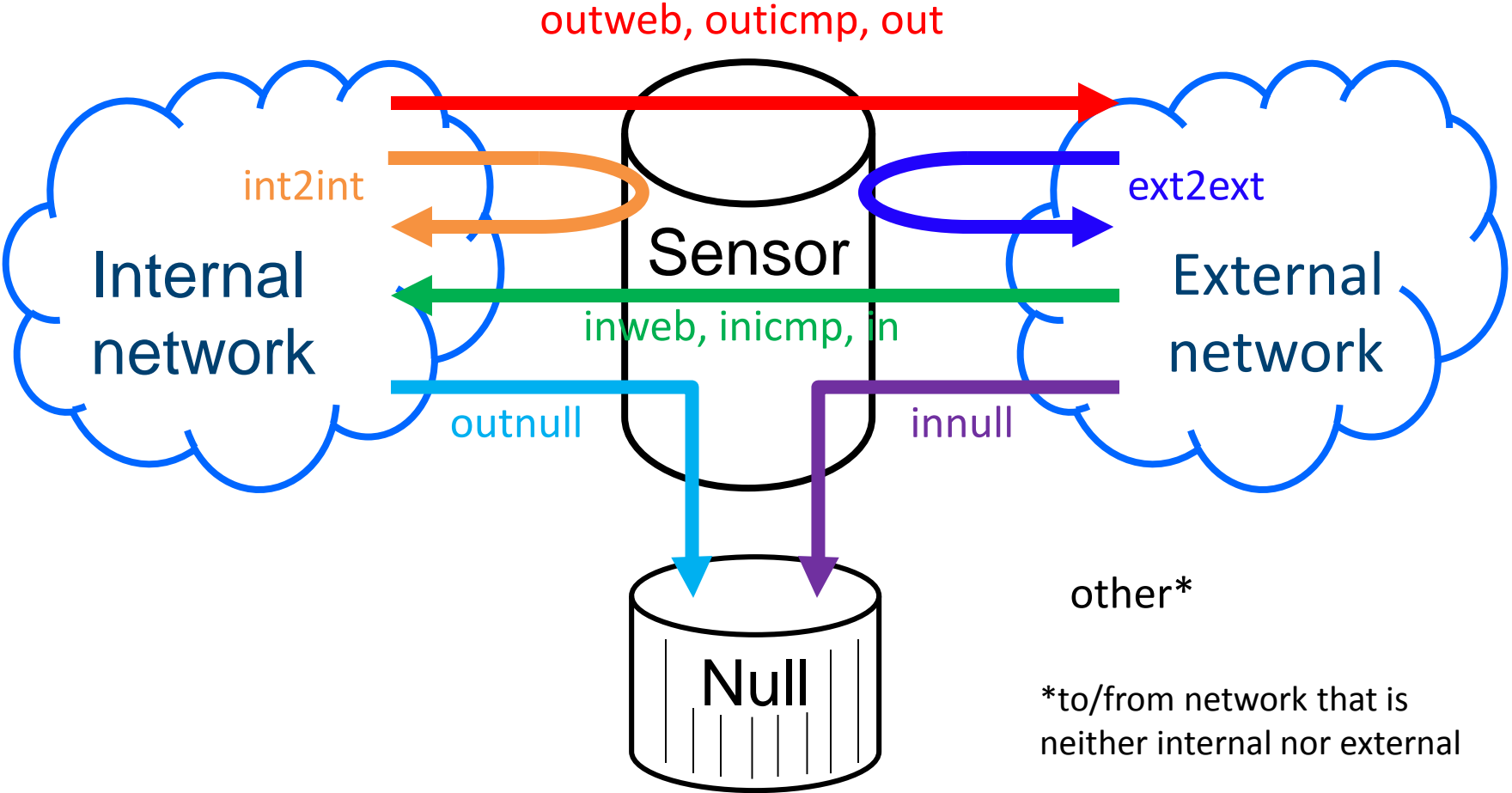
What Is This? — 3

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

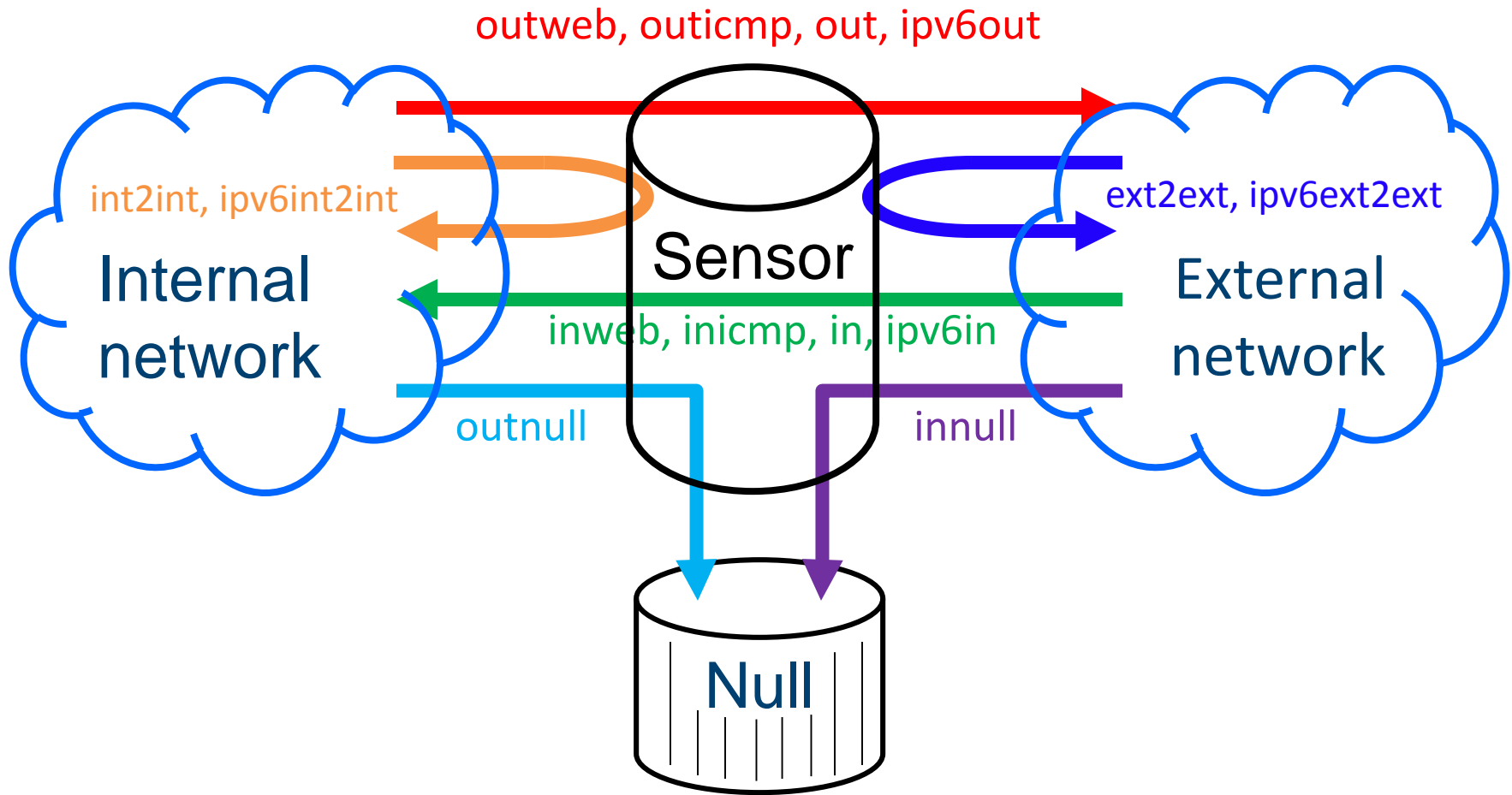
What Is This? — 4

sIP	dIP	sPort	dPort	pkts	bytes	flags	sTime
88.187.13.78	71.55.40.204	40936	80	83	3512	FS PA	2010/12/08T11:00:01
71.55.40.204	88.187.13.78	80	40936	84	104630	FS PA	2010/12/08T11:00:01
88.187.13.78	71.55.40.204	40938	80	120	4973	FS PA	2010/12/08T11:00:04
71.55.40.204	88.187.13.78	80	40938	123	155795	FS PA	2010/12/08T11:00:05
88.187.13.78	71.55.40.204	56172	80	84	3553	FS PA	2010/12/08T12:00:02
71.55.40.204	88.187.13.78	80	56172	83	103309	FS PA	2010/12/08T12:00:02
88.187.13.78	71.55.40.204	56177	80	123	5093	FS PA	2010/12/08T12:00:05
71.55.40.204	88.187.13.78	80	56177	124	157116	FS PA	2010/12/08T12:00:05

Standard SiLK Types



More complete 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 often default types

Lesson 1.2 Summary



Sensor placement affects what is seen or not seen in flow records.

We learned to interpret network activity from flow records.

A class of SiLK sensors uses a particular set of record types.

Next Lesson

In lesson I.3 we will learn how to issue *NIX commands, how to obtain online help for SiLK commands, and how to obtain information about the SiLK repository using a SiLK command.



FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics

<http://dylanbeattie.blogspot.com/2011/06/just-do-it-command-query-segregation.html>



Lesson I.3 Issuing SiLK Commands



Lesson I.3 Learning Objectives

- The learner will be able to issue simple SiLK commands correctly.
- The learner will be able to obtain online help text for SiLK commands and other *NIX commands.
- The learner will be able to obtain information about a SiLK sensor network and a SiLK flow-record repository.



*NIX commands

System prompt

Info + prompt character

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

User command

command name

rwfilter (case sensitive)

options

-h --help -k2 --key=2

arguments

results.rw

redirections

> >> < <<

pipe

|

For example:

```
rwcut --all-fields results.rw >results.txt
```

```
rwcut --fields=1-6 results.rw | more
```



<https://en.wikipedia.org/wiki/Linux>

Some standard *NIX commands

ls – list name and attributes of files and directories

cd – change the current working directory

cat – output the contents of a file

head – output the first lines of a file

echo – output the argument

more and **less** – display a file one page at a time

cut – output only selected fields of a file

sort – reorder the records (lines) of a file

wc – word count (optionally, character and line count) of a file

exit – logout and terminate a terminal window

*NIX Standard Streams

Standard In (**stdin**) – where normal (especially interactive) input comes from

Standard Out (**stdout**) – where normal/expected (especially interactive) output goes to

Standard Error (**stderr**) – where messages (especially unexpected) go to

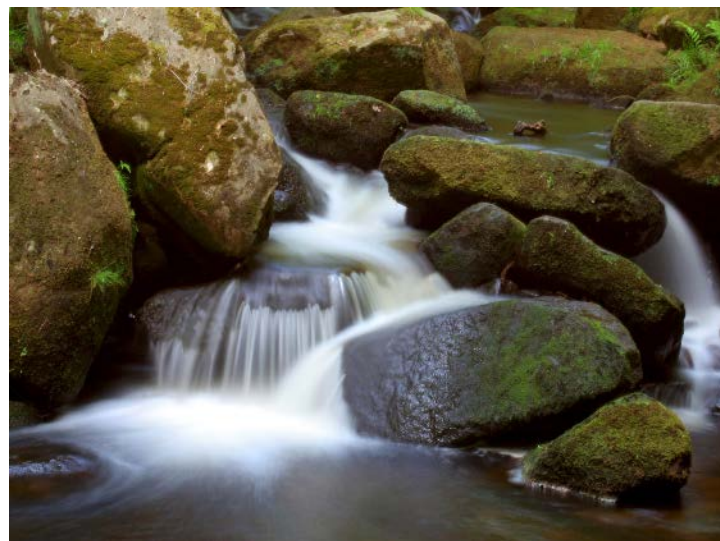
Defaults:

stdin – keyboard

stdout – screen/window

stderr – screen/window

Defaults are overridden by redirections and pipes



Shell Scripts



<http://clipgid.com/play-script-clipart.html>

Put a complicated command, pipeline, or sequence of pipelines into a script file.

- It saves your commands for reuse or learning.
- It eases making changes.

Create your script with `nano` or `vi` (`vim`). `vi` or `vim` can be found on every Linux/UNIX (*NIX) system.

Name your shell script something like `dothis.sh`

Add execute permission `chmod +x dothis.sh`

Execute (run) your script: `./dothis.sh`

 Software Engineering Institute | Carnegie Mellon University

SSH is the registered trademark of SSH Communications Security Corp

Exercise 1: Use a few relevant Linux commands

Create a new directory, change to it and use the `echo` command with

redirect “>”

and

append “>>”

to create a file.

Then examine it with `ls`, `cat` and `wc`

```
mkdir ex1
cd ex1
echo 10.1.60.25 > adr1.txt
echo 10.2.190.254 >> adr1.txt
ls adr1.txt
ls -l adr1.txt
cat adr1.txt
wc adr1.txt
cd
```

Software Engineering Institute | Carnegie Mellon University

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

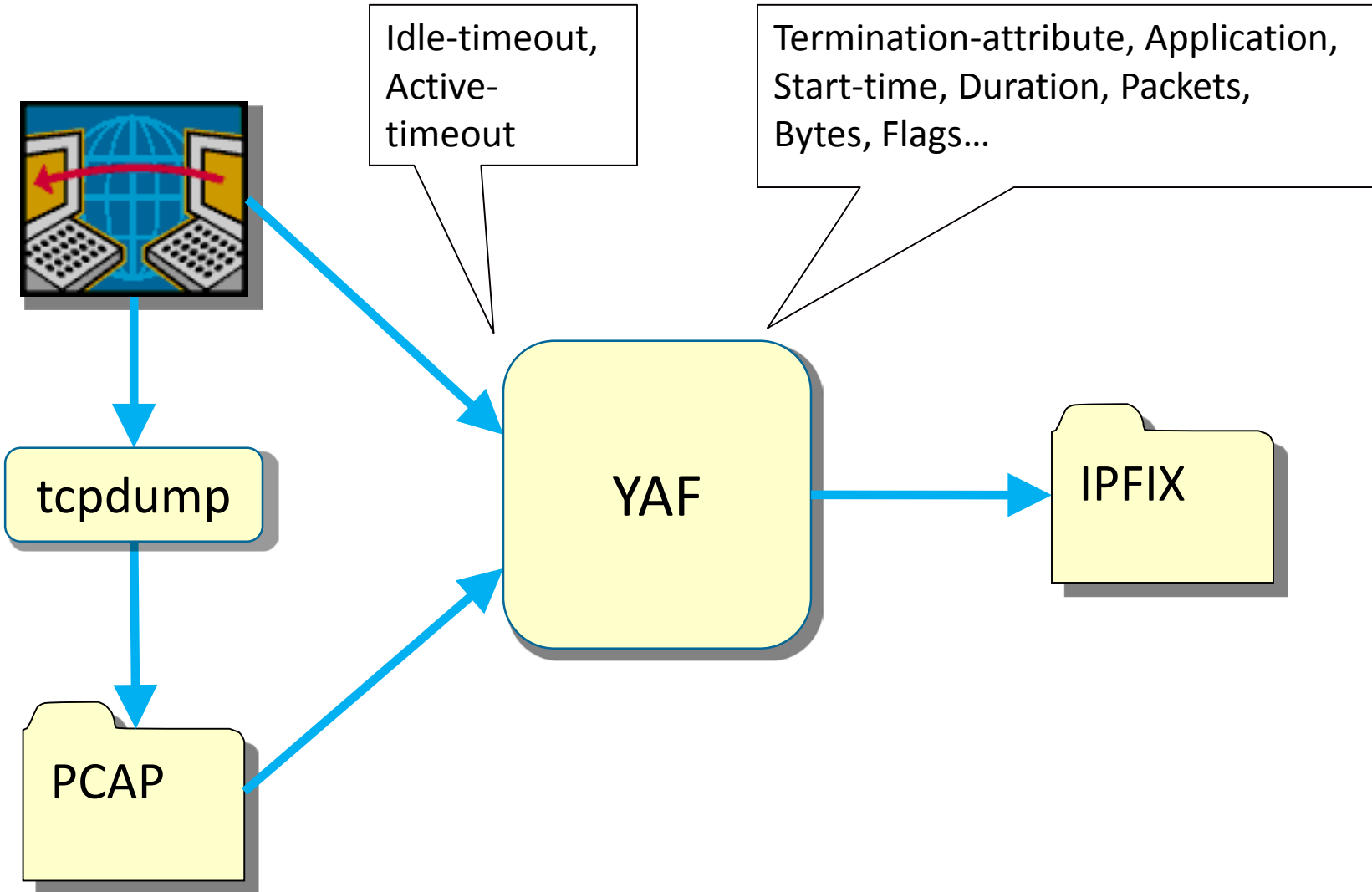
Packing stores flow records in a scheme optimized for space and ease of analysis

Analysis of flow data

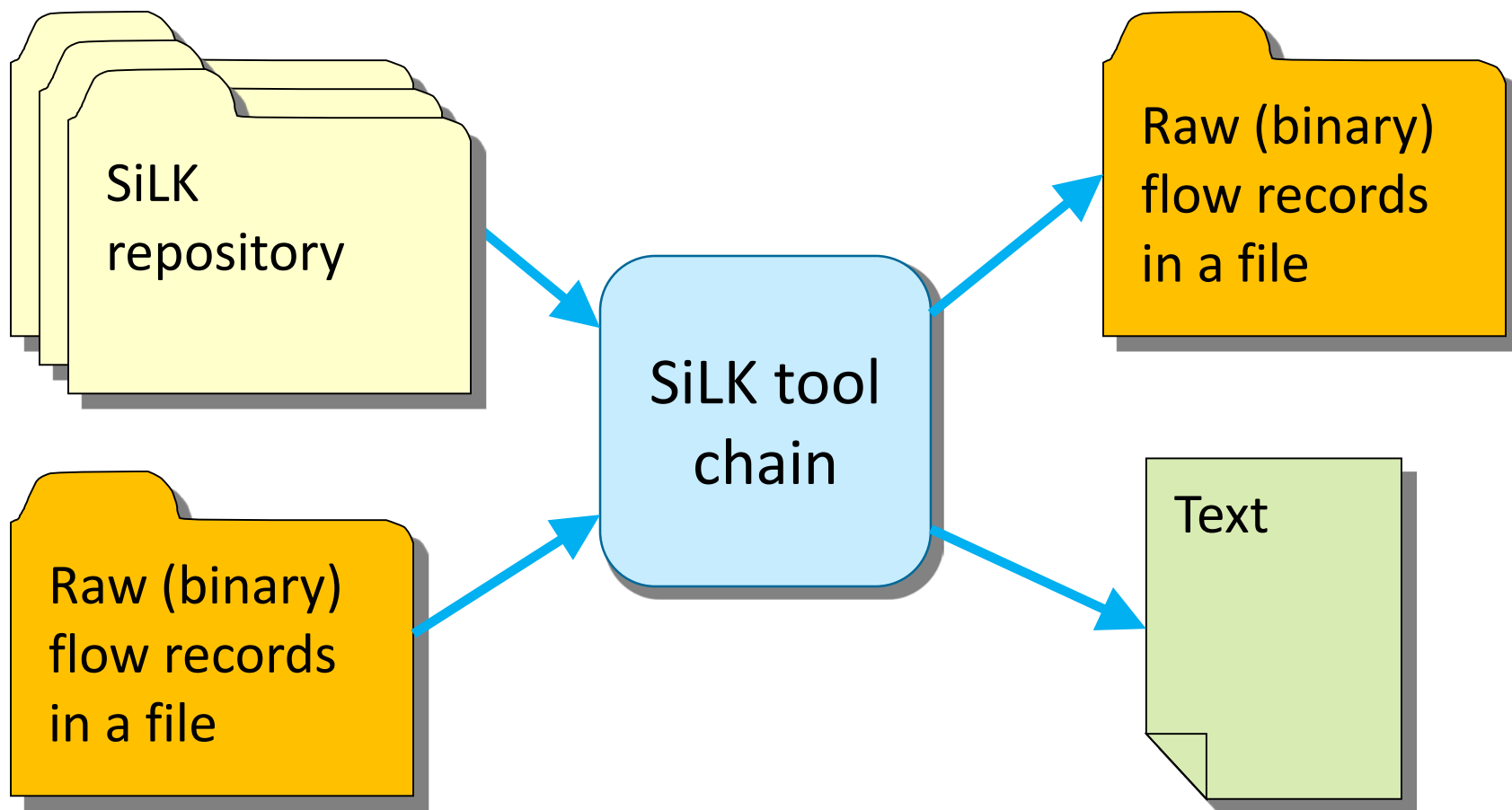
- Investigation of flow records using SiLK tools



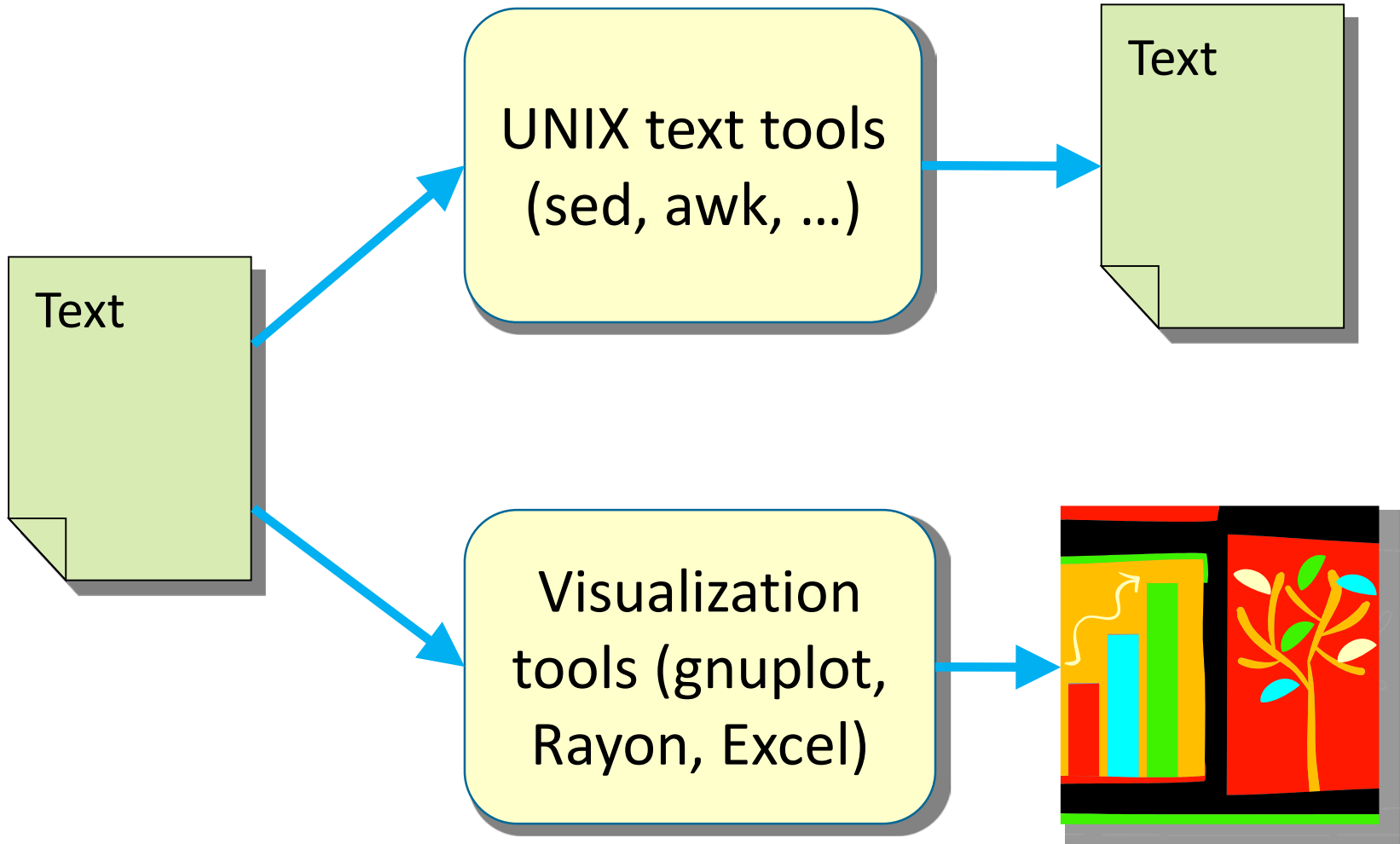
Collection



Analysis



Reporting



Exercise 2: Which sensors are defined?

```
rwsiteinfo --fields=id-sensor,sensor
```

```
rwsiteinfo --fields=id-sensor,sensor | less
```

```
rwsiteinfo --fields=id-sensor,sensor,describe-sensor \  
| less
```



 Software Engineering Institute <http://www.rainbird.com/landscape/products/central/flowsensors.htm>

Which record-types are defined?

rwsiteinfo --fields=type,mark-defaults



while this is true for most wines, there are always exceptions to the rule.

<http://winefolly.com/review/types-dessert-wine/>



Where can I get more information?

We can't discuss all parameters for every tool.

Resources

- Analyst's Handbook
- SiLK Reference Guide (collected man pages)
- `--help` option
- `man` command
- <http://tools.netsa.cert.org>



Answers to questions you haven't asked yet

At this point you probably have dozens of questions. Typical answers are:

- Yes, it does, and here is how to do it
- Yes, read about it in <reference>
- Yes, but it will take too long to describe right now
- Yes, but it is not a good idea because <some lame excuse>
- Because <long silence>
- No, it doesn't because <really good reason>
- No, it doesn't <long silence>
- No, but that's a really good idea, please email it to me. Thanks!

Lesson I.3 Summary



We learned the parts of *NIX commands.

Data should be kept in binary form as long as possible.

We learned where to get more information about commands.

We learned to obtain information about the SiLK repository using the rwsiteinfo command.

Next Lesson

In lesson II.1 we will learn how to choose just the flow records that are applicable to our inquiry.



Part II: Basic SiLK Tools



Part II Lessons: Network Flow

1. SiLK Records, Files, and the Repository
2. Analysis Tools and Categorization
3. IP Sets

FloCon 2016

12th Annual Open Forum for Large-Scale Network Analytics

<https://clearwatercompliance.com/hipaa-hitech-news/hipaa-and-it-security/hipaa-security-reminder-transporting-medical-records/>



Lesson II.1

SiLK Records, Files, and the Repository



Lesson II.1 Learning Objectives

- The learner will be able to display selected fields from a sequence of flow records.
- The learner will be able to determine which flow-record fields will be useful for a given analysis.
- The learner will be able to identify which rfilter keywords are selection options.
- The learner will be able to pull flow records from a SiLK repository.

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` option.



<http://www.ifrick.ch/2011/09/apple-senkt-preise-in-der-schweiz-teilweise-massiv/price-cut/>

sip	packets	type	flags
dip	bytes	in	initialflags
sport	sensor	out	sessionflags
dport	<i>scc</i>	<i>dur</i>	<i>application</i>
protocol	<i>dcc</i>	<i>stime</i>	<i>attributes</i>
class	<i>nhip</i>	<i>etime</i>	<i>itype & icode</i>

rwcut Default Display

By default

- sIP (1), sPort (3)
- dIP (2), dPort (4)
- Protocol (5)
- packets, bytes
- flags
- sTime, eTime, duration
- sensor

`--all-fields` # way too much info

`--fields=1-5,sTime` # just right

Create the ex3records.rw file

```
# rfilter will not overwrite a file
rm ex3records.rw
```

```
rwfilter --type=all \
        --sensor=S0 \
        --start=2009/04/20T11 \
        --proto=0- \
        --pass=stdout \
| rwsort --fields=stime \
| rfilter --input-pipe=stdin \
        --max-pass=30 \
        --proto=0- \
        --pass=ex3records.rw
```

Exercise 3: What do the data look like?

```
rwcut ex3records.rw --fields=1-5,packets
```

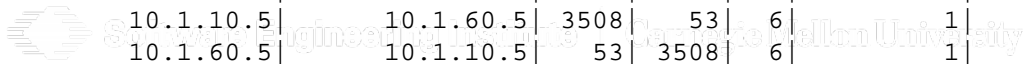
Try other values for `--fields`.

Try omitting the `--fields` option.

Exercise 3: What do the data look like?

```
rwcut --fields=1-5,packets ex3records.rw
```

sIP	dIP	sPort	dPort	pro	packets
10.1.60.203	10.1.60.187	50398	80	6	5
10.1.60.187	10.1.60.203	80	50398	6	5
10.1.60.203	10.1.60.73	50189	5222	6	6
10.1.60.73	10.1.60.203	5222	50189	6	5
10.1.60.203	10.1.60.187	49592	443	6	10
10.1.60.187	10.1.60.203	443	49592	6	10
10.1.60.203	10.1.60.187	0	2048	1	276
10.1.60.187	10.1.60.203	0	0	1	276
10.1.60.203	10.1.60.5	0	2048	1	279
10.1.60.5	10.1.60.203	0	0	1	279
10.1.60.203	10.1.60.5	56515	53	17	1
10.1.60.5	10.1.60.203	53	56515	17	1
10.1.60.203	10.1.60.73	0	2048	1	276
10.1.60.73	10.1.60.203	0	0	1	276
10.1.60.203	10.1.60.25	0	2048	1	356
10.1.60.25	10.1.60.203	0	0	1	356
10.1.60.203	10.1.60.25	60515	25	6	4
10.1.10.5	10.1.60.5	1031	53	17	32
10.1.60.5	10.1.10.5	53	1031	17	32
10.1.10.5	10.1.60.5	3507	53	6	1
10.1.60.5	10.1.10.5	53	3507	6	1
10.1.10.5	10.1.60.5	3508	53	6	1
10.1.60.5	10.1.10.5	53	3508	6	1
10.1.10.5	10.1.60.5	3507	53	6	1
10.1.10.5	10.1.60.5	3508	53	6	1
10.1.60.5	10.1.10.5	53	3507	6	1
10.1.60.5	10.1.10.5	53	3508	6	1
10.1.10.5	10.1.60.5	3507	53	6	1
10.1.10.5	10.1.60.5	3508	53	6	1
10.1.60.5	10.1.10.5	53	3508	6	1



Exercise 3: What do the data look like?

rwcut ex3records.rw

eTime sen	sIP	dIP sPort dPort pro	packets	bytes	flags	sTime	duration
0.006	10.1.60.203	10.1.60.187 50398 80 6	5	383	FS PA	2009/04/20T11:35:19.439	
0.005	10.1.60.187	10.1.60.203 80 50398 6	5	674	FS PA	2009/04/20T11:35:19.440	
0.009	10.1.60.203	10.1.60.73 50189 5222 6	6	433	FS PA	2009/04/20T11:35:19.446	
0.008	10.1.60.73	10.1.60.203 5222 50189 6	5	446	FS PA	2009/04/20T11:35:19.447	
0.056	10.1.60.203	10.1.60.187 49592 443 6	10	1085	FS PA	2009/04/20T11:35:19.463	
0.055	10.1.60.187	10.1.60.203 443 49592 6	10	4162	FS PA	2009/04/20T11:35:19.464	
1799.899	10.1.60.203	10.1.60.187 0 2048 1	276	23184		2009/04/20T11:35:19.490	
1799.899	10.1.60.187	10.1.60.203 0 0 1	276	23184		2009/04/20T11:35:19.491	
1799.900	10.1.60.203	10.1.60.5 0 2048 1	279	23436		2009/04/20T11:35:19.494	
1799.899	10.1.60.5	10.1.60.203 0 0 1	279	23436		2009/04/20T11:35:19.495	
0.002	10.1.60.203	10.1.60.5 56515 53 17	1	65		2009/04/20T11:35:19.500	
0.000	10.1.60.5	10.1.60.203 53 56515 17	1	81		2009/04/20T11:35:19.502	
1799.902	10.1.60.203	10.1.60.73 0 2048 1	276	23184		2009/04/20T11:35:19.514	
	10.1.60.73	10.1.60.203 0 0 1	276	23184		S0	

Exercise 3: What do the data look like?

```
rwcut --fields=1-5,packets,flags,stime ex3records.rw
```

sIP	dIP	sPort	dPort	pro	packets	flags	sTime
10.1.60.203	10.1.60.187	50398	80	6	5	FS PA	2009/04/20T11:35:19.439
10.1.60.187	10.1.60.203	80	50398	6	5	FS PA	2009/04/20T11:35:19.440
10.1.60.203	10.1.60.73	50189	5222	6	6	FS PA	2009/04/20T11:35:19.446
10.1.60.73	10.1.60.203	5222	50189	6	5	FS PA	2009/04/20T11:35:19.447
10.1.60.203	10.1.60.187	49592	443	6	10	FS PA	2009/04/20T11:35:19.463
10.1.60.187	10.1.60.203	443	49592	6	10	FS PA	2009/04/20T11:35:19.464
10.1.60.203	10.1.60.187	0	2048	1	276		2009/04/20T11:35:19.490
10.1.60.187	10.1.60.203	0	0	1	276		2009/04/20T11:35:19.491
10.1.60.203	10.1.60.5	0	2048	1	279		2009/04/20T11:35:19.494
10.1.60.5	10.1.60.203	0	0	1	279		2009/04/20T11:35:19.495
10.1.60.203	10.1.60.5	56515	53	17	1		2009/04/20T11:35:19.500
10.1.60.5	10.1.60.203	53	56515	17	1		2009/04/20T11:35:19.502
10.1.60.203	10.1.60.73	0	2048	1	276		2009/04/20T11:35:19.514
10.1.60.73	10.1.60.203	0	0	1	276		2009/04/20T11:35:19.516
10.1.60.203	10.1.60.25	0	2048	1	356		2009/04/20T11:35:19.528
10.1.60.25	10.1.60.203	0	0	1	356		2009/04/20T11:35:19.529
10.1.60.203	10.1.60.25	60515	25	6	4	S	2009/04/20T11:35:19.529
10.1.10.5	10.1.60.5	1031	53	17	32		2009/04/20T11:35:23.415
10.1.60.5	10.1.10.5	53	1031	17	32		2009/04/20T11:35:23.417
10.1.10.5	10.1.60.5	3507	53	6	1	S	2009/04/20T11:35:23.443
10.1.60.5	10.1.10.5	53	3507	6	1	R A	2009/04/20T11:35:23.444
10.1.10.5	10.1.60.5	3508	53	6	1	S	2009/04/20T11:35:23.445
10.1.60.5	10.1.10.5	53	3508	6	1	R A	2009/04/20T11:35:23.446
10.1.10.5	10.1.60.5	3507	53	6	1	S	2009/04/20T11:35:24.084
10.1.10.5	10.1.60.5	3508	53	6	1	S	2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3507	6	1	R A	2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3508	6	1	R A	2009/04/20T11:35:24.086
10.1.10.5	10.1.60.5	3507	53	6	1	S	2009/04/20T11:35:24.632
10.1.10.5	10.1.60.5	3508	53	6	1	S	2009/04/20T11:35:24.632
10.1.60.5	10.1.10.5	53	3508	6	1	R A	2009/04/20T11:35:24.633



Exercise 3: What do the data look like?

```
rwcut --fields=1-5,iType,iCode,packets,flags,stime ex3records.rw
```

sIP	dIP	sPort	dPort	pro	iTy	iCo	packets	flags	sTime
10.1.60.203	10.1.60.187	50398	80	6			5	FS PA	2009/04/20T11:35:19.439
10.1.60.187	10.1.60.203	80	50398	6			5	FS PA	2009/04/20T11:35:19.440
10.1.60.203	10.1.60.73	50189	5222	6			6	FS PA	2009/04/20T11:35:19.446
10.1.60.73	10.1.60.203	5222	50189	6			5	FS PA	2009/04/20T11:35:19.447
10.1.60.203	10.1.60.187	49592	443	6			10	FS PA	2009/04/20T11:35:19.463
10.1.60.187	10.1.60.203	443	49592	6			10	FS PA	2009/04/20T11:35:19.464
10.1.60.203	10.1.60.187	0	2048	1	8	0	276		2009/04/20T11:35:19.490
10.1.60.187	10.1.60.203	0	0	1	0	0	276		2009/04/20T11:35:19.491
10.1.60.203	10.1.60.5	0	2048	1	8	0	279		2009/04/20T11:35:19.494
10.1.60.5	10.1.60.203	0	0	1	0	0	279		2009/04/20T11:35:19.495
10.1.60.203	10.1.60.5	56515	53	17			1		2009/04/20T11:35:19.500
10.1.60.5	10.1.60.203	53	56515	17			1		2009/04/20T11:35:19.502
10.1.60.203	10.1.60.73	0	2048	1	8	0	276		2009/04/20T11:35:19.514
10.1.60.73	10.1.60.203	0	0	1	0	0	276		2009/04/20T11:35:19.516
10.1.60.203	10.1.60.25	0	2048	1	8	0	356		2009/04/20T11:35:19.528
10.1.60.25	10.1.60.203	0	0	1	0	0	356		2009/04/20T11:35:19.529
10.1.60.203	10.1.60.25	60515	25	6			4	S	2009/04/20T11:35:19.529
10.1.10.5	10.1.60.5	1031	53	17			32		2009/04/20T11:35:23.415
10.1.60.5	10.1.10.5	53	1031	17			32		2009/04/20T11:35:23.417
10.1.10.5	10.1.60.5	3507	53	6			1	S	2009/04/20T11:35:23.443
10.1.60.5	10.1.10.5	53	3507	6			1	R A	2009/04/20T11:35:23.444
10.1.10.5	10.1.60.5	3508	53	6			1	S	2009/04/20T11:35:23.445
10.1.60.5	10.1.10.5	53	3508	6			1	R A	2009/04/20T11:35:23.446
10.1.10.5	10.1.60.5	3507	53	6			1	S	2009/04/20T11:35:24.084
10.1.10.5	10.1.60.5	3508	53	6			1	S	2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3507	6			1	R A	2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3508	6			1	R A	2009/04/20T11:35:24.086
10.1.10.5	10.1.60.5	3507	53	6			1	S	2009/04/20T11:35:24.632
10.1.10.5	10.1.60.5	3508	53	6			1	S	2009/04/20T11:35:24.632
10.1.60.5	10.1.10.5	53	3508	6			1	R A	2009/04/20T11:35:24.633

Exercise 3: What do the data look like?

```

rwcut --fields=1-5,packets,flags,initialFlags,sessionFlags,stime ex3records.rw

```

sIP	dIP	sPort	dPort	pro	packets	flags	initialF	sessionF	sTime
10.1.60.203	10.1.60.187	50398	80	6	5	FS PA	S	F PA	2009/04/20T11:35:19.439
10.1.60.187	10.1.60.203	80	50398	6	5	FS PA	S A	F PA	2009/04/20T11:35:19.440
10.1.60.203	10.1.60.73	50189	5222	6	6	FS PA	S	F PA	2009/04/20T11:35:19.446
10.1.60.73	10.1.60.203	5222	50189	6	5	FS PA	S A	F PA	2009/04/20T11:35:19.447
10.1.60.203	10.1.60.187	49592	443	6	10	FS PA	S	F PA	2009/04/20T11:35:19.463
10.1.60.187	10.1.60.203	443	49592	6	10	FS PA	S A	F PA	2009/04/20T11:35:19.464
10.1.60.203	10.1.60.187	0	2048	1	276				2009/04/20T11:35:19.490
10.1.60.187	10.1.60.203	0	0	1	276				2009/04/20T11:35:19.491
10.1.60.203	10.1.60.5	0	2048	1	279				2009/04/20T11:35:19.494
10.1.60.5	10.1.60.203	0	0	1	279				2009/04/20T11:35:19.495
10.1.60.203	10.1.60.5	56515	53	17	1				2009/04/20T11:35:19.500
10.1.60.5	10.1.60.203	53	56515	17	1				2009/04/20T11:35:19.502
10.1.60.203	10.1.60.73	0	2048	1	276				2009/04/20T11:35:19.514
10.1.60.73	10.1.60.203	0	0	1	276				2009/04/20T11:35:19.516
10.1.60.203	10.1.60.25	0	2048	1	356				2009/04/20T11:35:19.528
10.1.60.25	10.1.60.203	0	0	1	356				2009/04/20T11:35:19.529
10.1.60.203	10.1.60.25	60515	25	6	4	S	S	S	2009/04/20T11:35:19.529
10.1.10.5	10.1.60.5	1031	53	17	32				2009/04/20T11:35:23.415
10.1.60.5	10.1.10.5	53	1031	17	32				2009/04/20T11:35:23.417
10.1.10.5	10.1.60.5	3507	53	6	1	S	S		2009/04/20T11:35:23.443
10.1.60.5	10.1.10.5	53	3507	6	1	R A	R A		2009/04/20T11:35:23.444
10.1.10.5	10.1.60.5	3508	53	6	1	S	S		2009/04/20T11:35:23.445
10.1.60.5	10.1.10.5	53	3508	6	1	R A	R A		2009/04/20T11:35:23.446
10.1.10.5	10.1.60.5	3507	53	6	1	S	S		2009/04/20T11:35:24.084
10.1.10.5	10.1.60.5	3508	53	6	1	S	S		2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3507	6	1	R A	R A		2009/04/20T11:35:24.085
10.1.60.5	10.1.10.5	53	3508	6	1	R A	R A		2009/04/20T11:35:24.086
10.1.10.5	10.1.60.5	3507	53	6	1	S	S		2009/04/20T11:35:24.632
10.1.10.5	10.1.60.5	3508	53	6	1	S	S		2009/04/20T11:35:24.632
10.1.60.5	10.1.10.5	53	3508	6	1	R A	R A		2009/04/20T11:35:24.633

Exercise 3a: I wonder what a raw file looks like?

```
cd # make home directory the working directory
rm -f ex3arecords.raw # remove file; ok if not there
rfilter --type=in \
  --start-date=2009/4/20:14 --protocol=0- \
  --compress=none \
  --max-pass=1 \
  --pass=ex3arecords.raw
ls -l ex3arecords.raw
rfileinfo ex3arecords.raw
rwcut --fields=1-5,packets ex3arecords.raw
rwcut --all-fields ex3arecords.raw
hexdump -C ex3arecords.raw # any readable text?
```

Exercise 3a Output

```
ex3a.sh
```

```
-rw-r--r--. 1 pnk pnk 264 Jan  7 21:10 ex3arecords.rw
  rwfileinfo ex3arecords.rw
```

```
ex3arecords.rw:
```

```
format(id)          FT_RWIPV6ROUTING(0x0c)
version             16
byte-order          littleEndian
compression(id)     none(0)
header-length       176
record-length       88
record-version      1
silk-version        3.10.0
count-records       1
file-size           264
command-lines
```

```
1  rwfilter --type=in --start-date=2009/4/20:11 --protocol=0- --compress=none --max-
pass=1 --pass=ex3arecords.rw
```

```
rwcut --fields=1-5,packets ex3arecords.rw
```

sIP	dIP	sPort	dPort	pro	packets
10.1.10.5	10.1.60.5	3507	53	6	1

```
rwcut --all-fields ex3arecords.rw
```

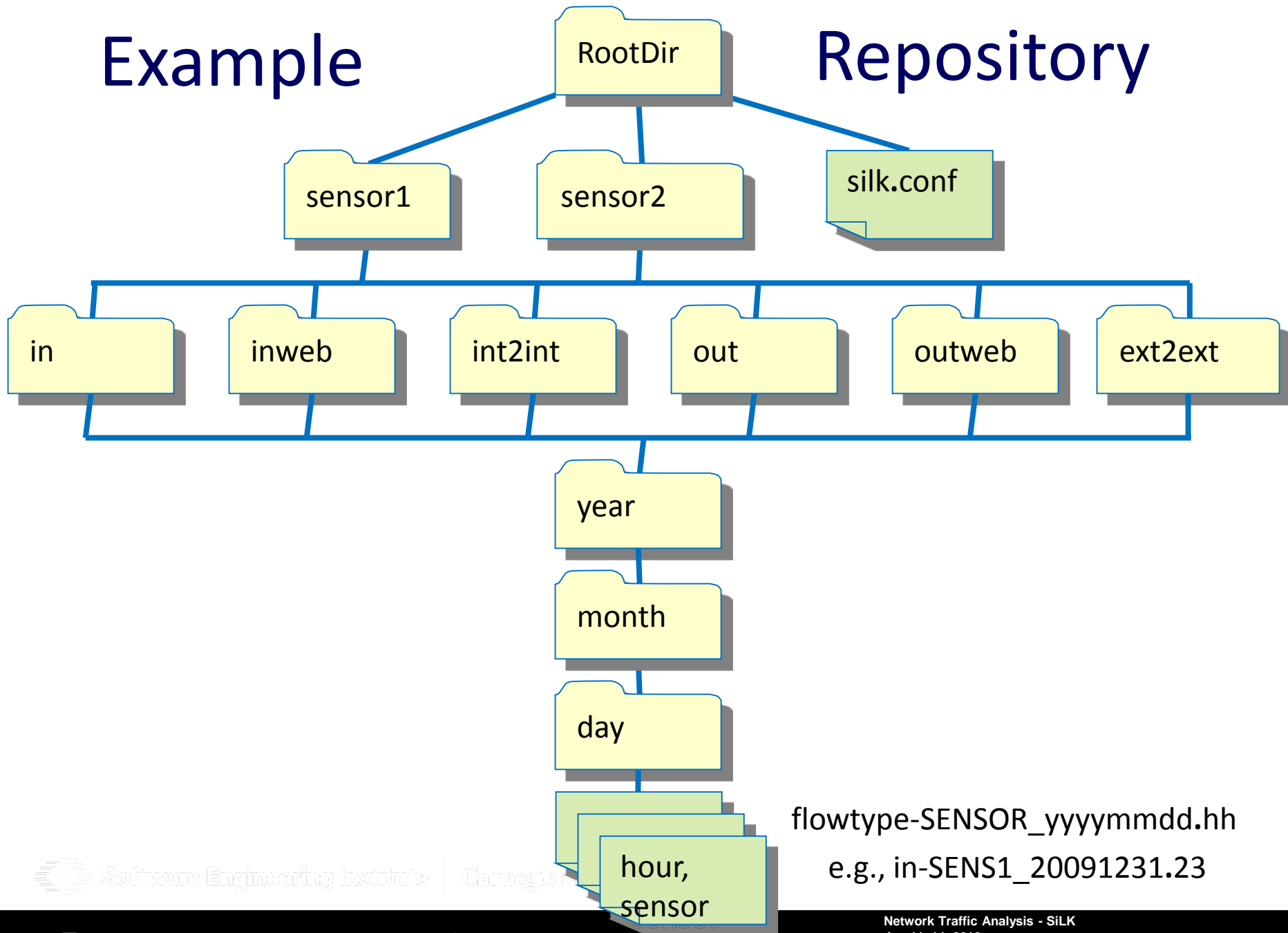
duration	sIP	eTime	sen	in	out	packets	bytes	flags	sTime			
type	sTime+msec	eTime+msec	eTime+msec	dur+msec	iTy	iCo	nhIP	initialF	sessionF	attribut	appli	cla
0.001	10.1.10.5	2009/04/20T11:35:23.444	10.1.60.5	3507	53	6	1	48	S			2009/04/20T11:35:23.443
in	2009/04/20T11:35:23.443	2009/04/20T11:35:23.444	2009/04/20T11:35:23.444	0.001							0	all

Exercise 3a Output

```
hexdump -C ex3arecords.rw # any readable text?
00000000  de ad be ef 00 0c 10 00 00 2d ed d0 00 58 00 01 |.....-...X..|
00000010  00 00 00 02 00 00 00 76 72 77 66 69 6c 74 65 72 |.....vrwfilter|
00000020  20 2d 2d 74 79 70 65 3d 69 6e 20 2d 2d 73 74 61 | --type=in --sta|
00000030  72 74 2d 64 61 74 65 3d 32 30 30 39 2f 34 2f 32 |rt-date=2009/4/2|
00000040  30 3a 31 31 20 2d 2d 70 72 6f 74 6f 63 6f 6c 3d |0:11 --protocol=|
00000050  30 2d 20 2d 2d 63 6f 6d 70 72 65 73 73 3d 6e 6f |0- --compress=no|
00000060  6e 65 20 2d 2d 6d 61 78 2d 70 61 73 73 3d 31 20 |ne --max-pass=1|
00000070  2d 2d 70 61 73 73 3d 65 78 33 61 72 65 63 6f 72 |--pass=ex3arecor|
00000080  64 73 2e 72 77 00 00 00 00 00 00 00 00 2a 00 00 |ds.rw.....*..|
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
000000b0  33 1e 4f c3 20 01 00 00 01 00 00 00 b3 0d 35 00 |3.O. ....5..|
000000c0  06 00 00 00 02 02 00 01 00 00 00 00 00 00 00 00 |.....|
000000d0  01 00 00 00 30 00 00 00 00 00 00 00 00 00 00 00 |....0.....|
000000e0  00 00 ff ff 0a 01 0a 05 00 00 00 00 00 00 00 00 |.....|
000000f0  00 00 ff ff 0a 01 3c 05 00 00 00 00 00 00 00 00 |.....<.....|
00000100  00 00 ff ff 00 00 00 00 |.....|
00000108
```

Example

Repository



flowtype-SENSOR_yyyymmdd.hh
e.g., in-SENS1_20091231.23

Basic SiLK Tools: `rwfilter`

Pick files from the repository

Compression

Plug in additional tools

Basic statistics

Direct flow output



Advanced flow-by-flow filtering

rwfilter Syntax

General form

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

Example call

```
rwfilter --sensor=S0 --type=in \  
--start-date=2015/8/5T13 \  
--end-date=2015/8/5T20 \  
--protocol=0- --pass=workday-5.rw
```

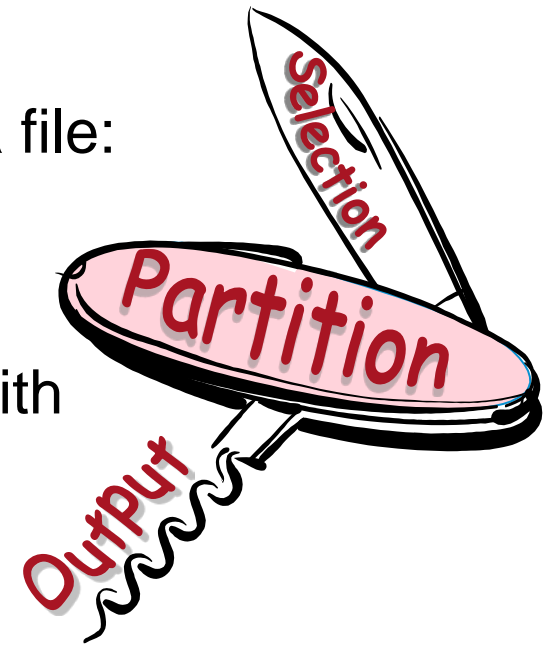

Selection and Input Criteria

Selection options control access to repository files:

- `--start-date=2009/4/21`
- `--end-date=2009/4/21T03`
- `--sensor=S0`
- `--type=in,inweb`

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

- `myfile.rw`
- `stdin`
- useful for chaining filters through a pipe with `stdin/stdout`



Basic Partitioning Options

- Simple numeric fields: ports, protocol, ICMP Type
- Specified IP addresses, CIDR blocks
- Sets of IP addresses
- Combinations of key fields – Tuple files

Simple Numeric Key Fields

--protocol=
--sport= --dport= --aport= # source, dest, any

--protocol=6,17 # TCP or UDP
--protocol=0-5,7-16,18- # not TCP or UDP
--protocol=0- # all protocols
--dport=80,443 # HTTP or HTTPS
--sport=6000-6063,9100-9107 # X11 or JetDirect
--aport=20,21 # FTP
--sport=0-1023 # Well-Known Ports

Specified IP address or CIDR block

--saddress= --daddress= --any-address=
--not-saddress= --not-daddress= --not-any-address=

May specify a single:

IP address	192.0.2.1
CIDR block	192.0.2.0/24

Specified IP addresses or CIDR blocks

--**s**cidr= --**d**cidr= --any-cidr=
--not-**s**cidr= --not-**d**cidr= --not-any-cidr=

May specify multiple:

IP addresses	192.0.2.1,198.51.100.3
CIDR blocks	192.0.2.0/24,198.51.100.0/24
mixture	192.0.2.1,192.0.2.8/29

Sets of arbitrary addresses

--sipset= **--dipset=** **--anyset=**
--not-sipset= **--not-dipset=** **--not-anyset=**

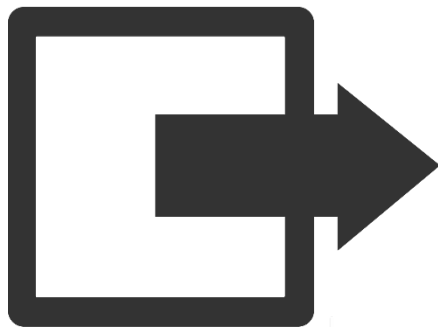
Specifies the name of a file storing the IP set:

--sipset=internalservers.set
--dipset=RussianBizNtwk.set
--anyset=TorNodes.set
--not-dipset=whitelist.set



rwfilter output options

<code>--pass-destination=</code>	<code># file to get records that pass</code>
<code>--fail-destination=</code>	<code># file to get records that fail</code>
<code>--all-destination=</code>	<code># file to get all records</code>
<code>--print-statistics</code>	<code># report recs read/pass/fail</code>
<code>--print-volume-statistics</code>	<code># report how many</code>
	<code># recs/pkts/bytes pass/fail</code>



What Is Going on Here? — 5

```
rwfilter --sensor=S0 --type=in \  
  --start=2009/4/21T00 --end=2009/4/21T07 \  
  --daddress=10.1.0.0/16 --print-volume-stat
```

	Recs	Packets	Bytes	Files
Total	1436	2615	158084	8
Pass	1436	2615	158084	
Fail	0	0	0	

Exercise 4: `rwfilter`

- 1) Find all traffic going outbound to external HTTPS servers on April 20, 2009. Save these flows in file `https0420.rw`. Only pull records captured by sensor S0.
- 2) How many flow records matched the criteria?

Exercise 4: `rwfilter`

- 1) Find all traffic going outbound to external HTTPS servers on April 20, 2009. Save these flows in file `https0420.rw`. Only pull records captured by sensor S0.
- 2) How many flow records matched the criteria?

Hint

HTTPS normally uses port 443

Exercise 4: `rwfilter` solution

```
rwfilter --sensor=S0 --type=outweb \  
  --start=2009/4/20 --dport=443 \  
  --pass=https0420.rw --print-volume-statistics
```

	Recs	Packets	Bytes	Files
Total	1308	37588	39354028	13
Pass	174	2413	223465	
Fail	1134	35175	39130563	

```
rwfileinfo https0420.rw --fields=count
```

```
https0420.rw:
```

```
count-records 174
```

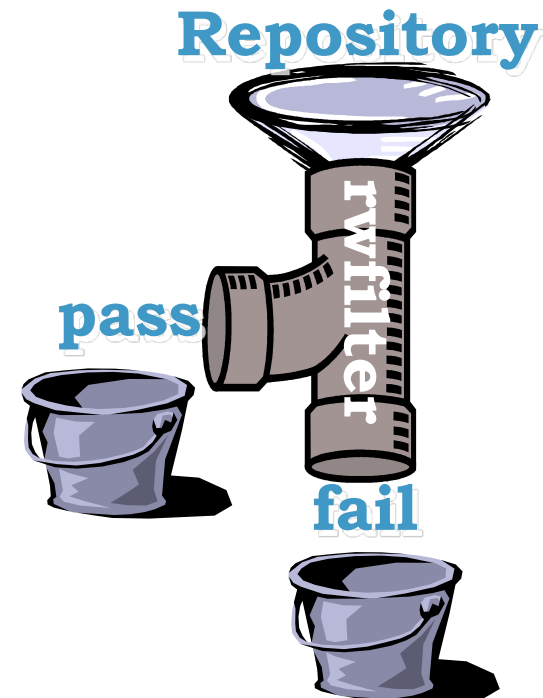
Output Criteria

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 (screen not allowed for non-text data).

Other useful output

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



What Is This? — 8

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

One day's outgoing web, but only if 100,000 or more bytes per flow

Chain two rwfilter calls

One minute or more -> long
Less than one minute -> short

Answer: Classifies 100,000+-byte web output flows by fast or slow transfer. Bursty vs. Persistent?

Example Typos

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

SiLK Commandments

1. Thou shalt use Sets instead of using several rfilter 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 rfilter 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.

Lesson II.1 Summary



We learned how to display the fields of interest from flow records.

Files are chosen from the repository with selection options. Records are chosen from those files with partitioning options.

There are lots of ways to partition on IP addresses.

Next Lesson

In lesson II.2 we will learn to reduce large numbers of flow records to meaningful information and statistics.



FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics



Lesson II.2 Analysis Tools and Categorization



Lesson II.2 Learning Objectives

- The learner will be able to create a time series of given flow records.
- The learner will be able to determine all the different values of a given field for given flow records and determine the traffic volumes for those field values.
- The learner will be able to display the top/bottom n values of a given field as measured by some measure of volume.

Basic SiLK Counting Tools: `rwcount`, `rwstats`, `rwuniq`

“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 all the pairs.”
- potential scanners:
 - “Count unique dIPs by sIP and print the sources that contacted more than 100 destinations.”

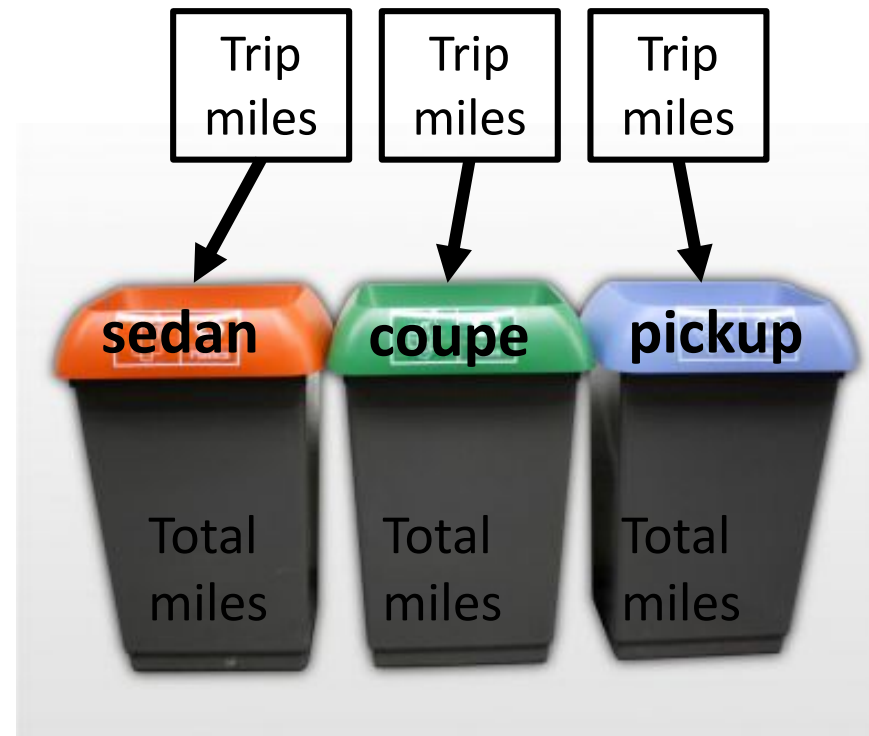
Categorization—Bins

For motor vehicle trips we could bin trip records by

- vehicle style – sedan, coupe, SUV, pickup, van
- highway or city trip
- personal or business trip

We could measure the trips and aggregate in bins

- total miles
- fuel consumption
- oil consumption
- pollutant emission



Bins

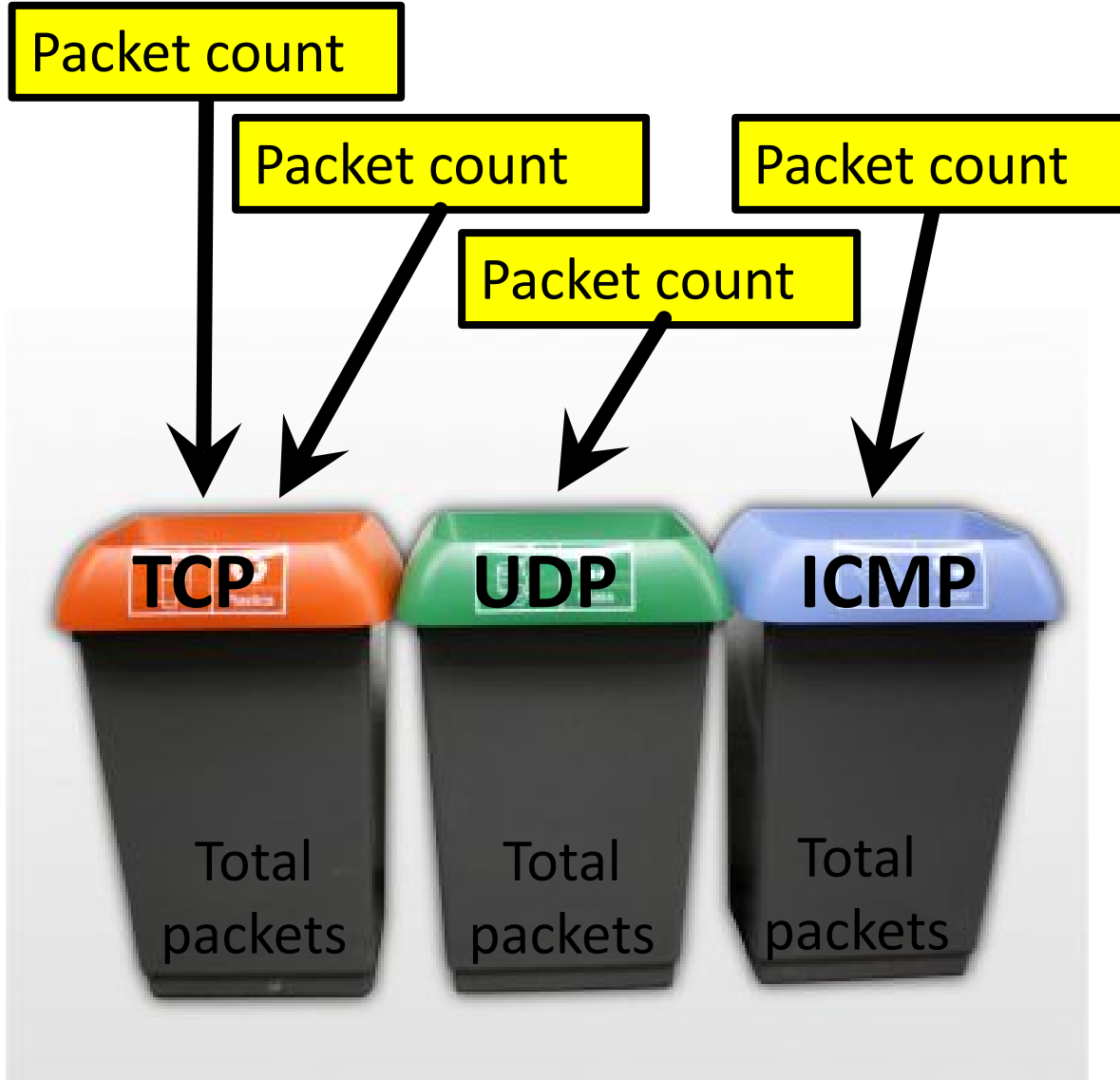
For flows we could bin by

- address or address block
- port
- protocol
- time period

We could measure the flows and aggregate in bins

- count of flow records, packets, bytes
- count of distinct values of other fields, e.g., addr
- earliest sTime, latest eTime

Bins



**Value
from flow
record**
e.g., packets

Bin key field
e.g., protocol

**Aggregate
Value**

Basic SiLK Counting Tools: `rwcount`, `rwstats`, `rwuniq`

`rwcount`: count volume across time periods

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

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

“Key field” = SiLK fields defining bins

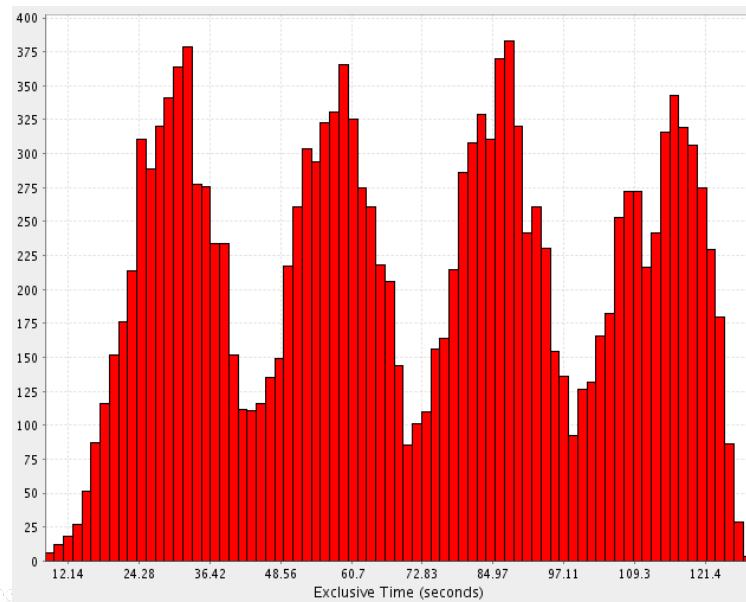
“Volume” = {Records, Bytes, Packets} and a few others
measure

aggregate value

Each tool reads raw binary flow records as input.

rwcount

- count records, bytes, and packets by time and display results
`rwcount --bin-size=300`
- fast, easy way of summarizing volumes as a time series
- great for simple bandwidth studies
- easy to take output and make a graph with graphing S/W

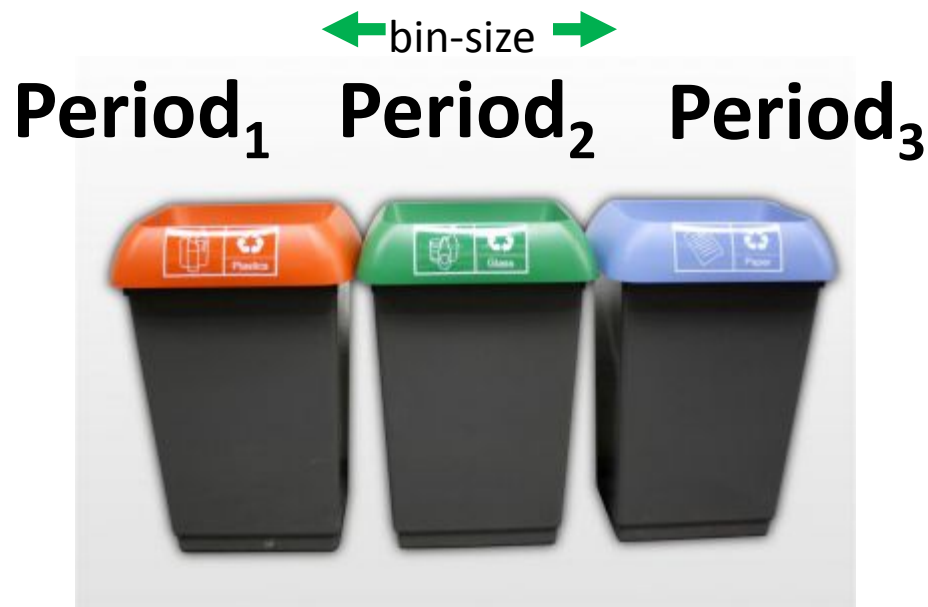


<http://www.cs.uoregon.edu/research/tau/docs/paraprof/ch05s02.html>

Time Bins

When binning by time, you must specify the period of time for each bin. This is called the **bin-size**.

It's the size of the bin's opening, not the volume of the container.



rwcount

The bin key is always time. You choose the period.

The aggregate measures are chosen for you. They are flows (records), bytes, packets.

```
rwfilter --sensor=S0 --start=2009/4/21 \  
        --type=in --proto=1 --pass=stdout \  
| rwcount --bin-size=3600
```

	Date	Records	Bytes	Packets
...				
	2009/04/21T13:00:00	10.00	2460.00	41.00
	2009/04/21T14:00:00	29.00	8036.00	107.00
	2009/04/21T15:00:00	22.00	2214.00	47.00
	2009/04/21T16:00:00	10.00	1586.00	23.00

What Is This? — 9

```
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: `rwcount`

The shell can help with the arithmetic: `$((24*60*60))`

You also can find common periods in the Quick Reference Guide.

Time series for all outgoing traffic on sensor S0:

```
rwfilter --sensor=S0 --type=out,outweb \  
  --start=2009/04/21 --end=2009/04/23 \  
  --proto=0- --pass=stdout \  
| rwcount --bin-size=$((24*60*60))
```

Exercise 5: rwcoun

Produce a time-series with 30-minute intervals, analyzing incoming ICMP traffic collected at sensor S0 on April 20, 2009.

Exercise 5: rwcoun

Produce a time-series with 30-minute intervals, analyzing incoming ICMP traffic collected at sensor S0 on April 20, 2009.

HINT

ICMP is Protocol 1

Exercise 5: rwcoun solution

```
rwfilter --sensor=S0 --type=in \  
  --start=2009/4/20 --protocol=1 --pass=stdout \  
| rwcoun --bin-size=1800
```

Date	Records	Bytes	Packets
...			
2009/04/20T13:30:00	5.05	1588.92	26.48
2009/04/20T14:00:00	21.92	5480.87	91.35
2009/04/20T14:30:00	8.03	3610.21	60.17
2009/04/20T15:00:00	14.58	5432.54	90.54
2009/04/20T15:30:00	17.33	6519.74	108.66
2009/04/20T16:00:00	13.69	5702.65	95.04
2009/04/20T16:30:00	12.89	5105.11	85.09
2009/04/20T17:00:00	11.50	5135.57	85.59
2009/04/20T17:30:00	7.00	2704.40	45.07

rwuniq

rwuniq will display all bins for a particular field or fields.

Output is normally unsorted.

--sort-output causes sorting by the key (bin).

Calling `rwuniq`

`rwuniq --fields=KEY --value=VOLUME`

- Choose one or several key fields.
- Aggregate volume count: records, bytes, or packets.
- standard output formatting options (see “man `rwuniq`”)

Apply thresholds to bins before outputting:

- `--bytes`, `--packets`, `--flows`, `--sip-distinct`,
`--dip-distinct`
- Specify minimum aggregate value or a range

`--sort-output` by key (`rwstats` sorts by value)

What Is This? – 10

```
rwfilter outtraffic.rw \  
  --stime=2010/12/08:18:00:00-2010/12/08:18:59:59.999 \  
  --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

```
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

Exercise 6: `rwuniq`

For outgoing flows from S0 on 2009/04/20, write and execute the `rwfilter` piped to `rwuniq` commands to list how many TCP flows (records) there were with each different number of packets. Display sorted by the number of packets.

Are there any odd results you can explain?

Exercise 6: `rwuniq`

For outgoing flows from S0 on 2009/04/20, write and execute the `rwfilter` piped to `rwuniq` commands to list how many TCP flows (records) there were with each different number of packets. Display sorted by the number of packets.

Are there any odd results you can explain?

HINT

TCP is protocol 6

Exercise 6: `rwuniq`

For outgoing flows from S0 on 2009/04/20, write and execute the `rwfilter` piped to `rwuniq` commands to list how many TCP flows (records) there were with each different number of packets. Display sorted by the number of packets.

Are there any odd results you can explain?

HINT

TCP is protocol 6 (`proto=6`)

The TCP 3-way handshake
requires 3 packets

Exercise 6: `rwuniq` Solution

```
rwfilter --type=out,outweb \
  --sensor=S0 \
  --start=2009/4/20 \
  --proto=6 \
  --pass=stdout \
| rwuniq --fields=packets --sort-output
```

packets	Records
1	2573
2	129
3	133
4	25
5	271
6	289
7	182
8	74
9	61
10	20
11	20
12	8
13	16
14	7
15	1
16	6
17	1
18	3
19	2

What can you say about flows with 1, 2 and 3 packets?

It seems as though 4 packets is an oddity.

Do you have an explanation? What can be accomplished with 4 TCP packets?

There are, of course, exceptions

rwstats

Like rwuniq, rwstats displays bins for a field or fields, but only displays the top N or bottom N bins.

The top/bottom N is determined by some traffic volume measurement, such as flows, packets, or bytes.

The bins are displayed sorted by the measurement.

It also provides percentages.

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`”)

What Is This? – 12

```
rwfilter outtraffic.rw \  
  --stime=2010/12/08T18:00:00-2010/12/08T18:59:59 \  
  --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 7: `rwstats`

What are the top 10 incoming protocols on April 20, 2009, collected on sensor S0?

Exercise 7: rwstats

What are the top 10 incoming protocols on April 20, 2009, collected on sensor S0?

HINT

Incoming flows have type **in** or **inweb**

Exercise 7: rstats solution

```
rwfilter --sensor=S0 --type=in,inweb \  
  --start=2009/4/20 --prot=0- --pass=stdout \  
| rstats --fields=protocol --value=records --count=10
```

INPUT: 5512 Records for 3 Bins and 5512 Total Records

OUTPUT: Top 10 Bins by Records

pro	Records	%Records	cumul_%
6	4476	81.204644	81.204644
17	896	16.255443	97.460087
1	140	2.539913	100.000000

Exercise 8: `rwstats`

Top 9 inside hosts according to how many outside hosts they communicate with on April 20, 2009, collected on sensor S0?

.

Exercise 8: rwstats

Top 9 inside hosts according to how many outside hosts they communicate with on April 20, 2009, collected on sensor S0?

HINT

Use

`--value=distinct:dip`

Exercise 8: rstats solution

```
rwfilter --sensor=S0 --type=out,outweb \  
  --proto=0- --start-date=2009/4/20 --pass=stdout \  
 | rstats --fields=sip --value=distinct:dip --count=9
```

INPUT: 5001 Records for 14 Bins

OUTPUT: Top 9 Bins by dIP-Distinct

sIP	dIP-Distin	%dIP-Disti	cumul_%
10.1.60.187	17	?	?
10.1.60.5	11	?	?
10.1.60.25	11	?	?
10.1.60.191	9	?	?
10.1.60.73	5	?	?
10.1.60.253	3	?	?
10.1.60.251	3	?	?
212.117.116.35	3	?	?
10.1.60.4	2	?	?

--no-percents will clean up the question marks.

rwuniq VS. rwstats - 1

rwuniq	both	rwstats in top/bottom mode
all bins except per thresholds	Bin by key	--top or --bottom bins
	Default aggregate value is flows (records).	
--sort-output by key otherwise unsorted		Sorted by primary aggregate value
Thresholds or ranges: --bytes , --packets , --flows , --sip-distinct , --dip-distinct	Choose which bins have aggregate values significant enough to output.	--count , --threshold , --percentage

rwuniq vs. rwstats - 2

rwuniq	both	rwstats in top/bottom mode
--all-counts (bytes, pkts, flows, earliest sTime, and latest eTime)	Show volume aggregate value[s].	--no-percents (good when primary aggregate isn't Bytes, Packets, or Records)
	--bin-time to adjust sTime and eTime	
	--presorted-input (omit when value includes Distinct fields, even if input is sorted)	
--values= sTime-Earliest, eTime-Latest	--values= Records, Packets, Bytes, sIP-Distinct, dIP-Distinct, Distinct:KEY-FIELD (KEY-FIELD can't also be key field in --fields)	

Lesson II.2 Summary



We learned how to categorize flow records by time or some other field.

Display a time series of flows with `rwcount`.

Display all categories (bins) with `rwuniq`.

Display the top or bottom bins, according to some measurement, with `rwstats`.

FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics

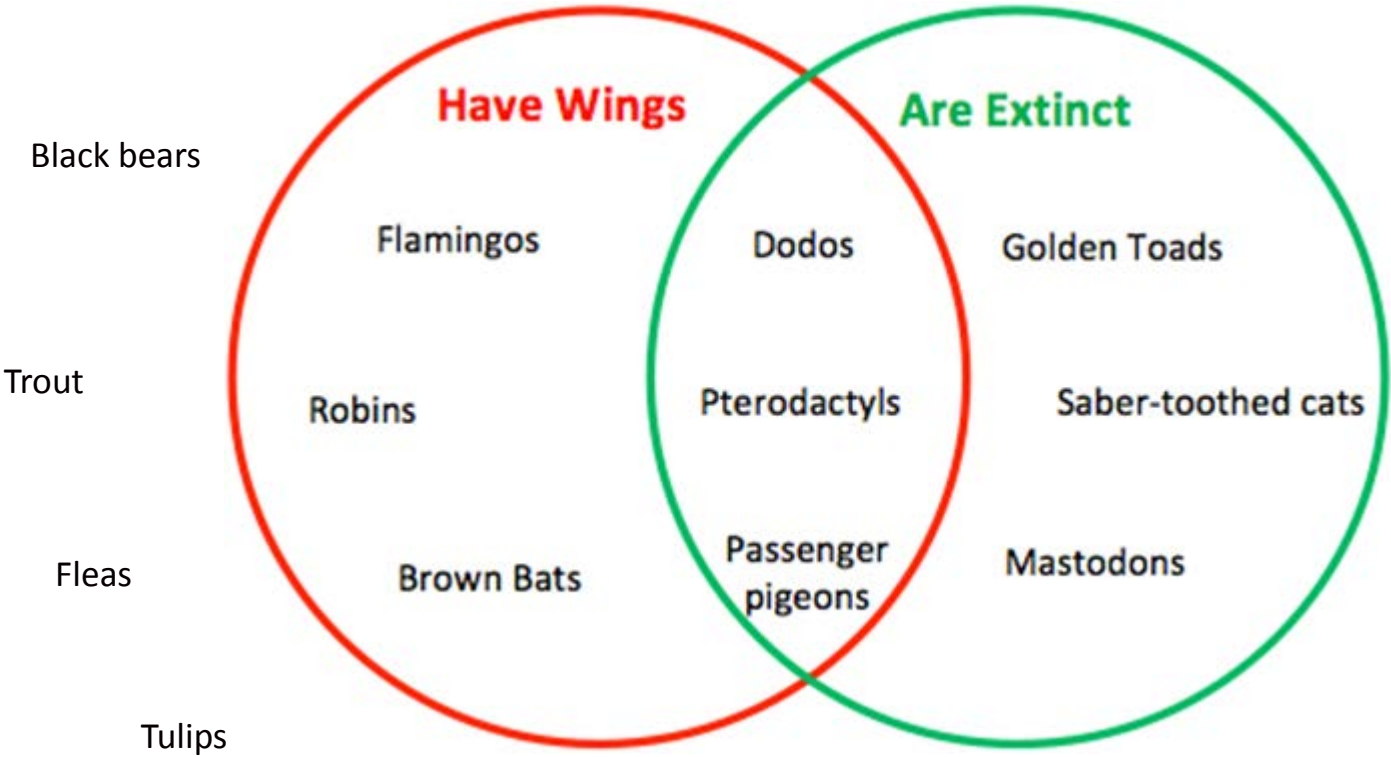
Lesson II.3

IP Sets

Lesson II.3 Learning Objectives

- Given a collection of IP addresses and CIDR blocks, the learner will be able to create an IP Set SiLK-file.
- Given an IP Set, the learner will be able to display the contents and characteristics of the set.
- Given an IP Set, the learner will be able to partition flow records based on the presence/absence of IP addresses in the set.
- Given a sequence of flow records, the learner will be able to extract IP addresses from the records and create an IP Set.

Sets



<http://www.mywordtemplates.org/diagram/template1501.html>

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!

- From individual IP address in dotted decimal or integer format
- From CIDR blocks, e.g., 192.168.0.0/16
- From flow records

Use it directly within your rfilter commands.

- `--sipset`, `--dipset`, `--anyset`
- `--not-sipset`, `--not-dipset`, `--not-anyset`

Set Tools

rwsetbuild: Create a set from text.

rwsetcat: Display an IP set as text.

rwset: Create sets from binary flow records.

rwsetmember: Test if an address is in given IP sets.

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

Creating a Set from a text file

Start with a text file containing IP addresses

IPv4 in dotted quad notation

IPv6 in canonical format (e.g. 2001:db8::f00)

Run **rwsetbuild** to make the conversion from text to set

```
$ cat sample.set.txt
192.168.1.1
172.16.0.1
10.1.2.3
$
$ rwsetbuild sample.set.txt sample.set
$ ls -l sample*
-rw-r--r--. 1 pnk pnk 124 Jan  7 17:22 sample.set
-rw-r--r--. 1 pnk pnk  32 Jan  7 17:21 sample.set.txt
$ rwsetcat sample.set
10.1.2.3
172.16.0.1
192.168.1.1
$
```

Exercise 9: Create a set file

In Exercise 1 you created the text file **adr1.txt**

It should contain two IPv4 addresses in dotted quad notation

Create a set file from it

Exercise 9: Create a set file

In Exercise 1 you created the text file **adr1.txt**

It should contain two IPv4 addresses in dotted quad notation

Create a set file from it

HINT

Use

```
Rwsetbuild <text file> <set file>
```

Exercise 9: Create a set file

In Exercise 1 you created the text file **adr1.txt**

It should contain two IPv4 addresses in dotted quad notation

Create a set file from it

HINT

If you run it twice, `rwsetbuild` will not overwrite the set file

You'll have to delete it first.

Create a Set of IP CIDR Blocks

```
$ cp ~rbandes/public/private_example.set.txt . # copy file
$ cat private_example.set.txt # display file
10.0.0.0/8 # RFC 1918 private
172.16.0.0/12 # RFC 1918 private
192.0.2.0/24 # documentation (example.com or example.net)
192.168.0.0/16 # RFC 1918 private
198.51.100.0/24 # documentation (example.com or example.net)
203.0.113.0/24 # documentation (example.com or example.net)
$ rwsbuild private_example.set.txt private_example.set
$ rwssetcat private_example.set | head -n 5
10.0.0.0
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
$ rwssetcat --count-ips private_example.set
17892096
```

Use IP Set as Partitioning Criterion

```
$ rfilter --type=in,inweb --start=2009/4/20 --end=2009/4/24 \  
  --sipset=private_example.set --print-volume-statistics
```

	Recs	Packets	Bytes	Files
Total	2563253	9609775	5501740288	369
Pass	2557016	9603538	5501284187	
Fail	6237	6237	456101	

Find Addresses from Traffic NOT in the IP Set

```
$ rfilter --type=in,inweb --start=2009/4/20 --end=2009/4/24 \  
  --not-sipset=private_example.set --pass=stdout \  
  | rset --sip-file=outside_not_private.set  
  
$ rsetcat --count-ips outside_not_private.set  
6237
```


Examine the IP Set

```
$ rwsocat outside_not_private.set | less
$ rwsocat --cidr-blocks outside_not_private.set | less
$ rwsocat --network-structure=8 outside_not_private.set
  100.0.0.0/8 | 6237
$ rwsocat --network-structure=16 outside_not_private.set \
  | wc -l
2
$ rwsocat --network-structure=16 outside_not_private.set
  100.1.0.0/16 | 5932
  100.2.0.0/16 | 305
$ rwsocat --network-structure=24 outside_not_private.set \
  | wc -l
264
```

Exercise 10 Sets

1) For April 21, 2009 on sensor S0, make a set-file of addresses of all actual inside hosts.

Should we examine incoming or outgoing traffic?

2) Make a set-file of all outside addresses.

Can you make both sets with one command?

Exercise 10 Sets

1) For April 21, 2009 on sensor S0, make a set-file of addresses of all actual inside hosts.

Should we examine incoming or outgoing traffic?

2) Make a set-file of all outside addresses.

Can you make both sets with one command?

HINT

Pipe `rwfilter` to `rwset`

Set Exercise 10 solution

```
rwfilter --sensor=S0 --type=out,outweb \  
        --start-date=2009/4/21 \  
        --proto=0- --pass=stdout \  
| rwset --sip-file=insidehosts.set \  
        --dip-file=outsidehosts.set
```

Exercise 11 Sets

Examine the two set-files from Exercise 9.

Exercise 11 Sets

Examine the two set-files from Exercise 9.

HINT

How big are the set files?

What can you say about the files?

How many addresses in each set?

What are they?

Set Exercise 11 solution

```
ls -l insidehosts.set
```

```
rwfileinfo insidehosts.set
```

```
rwsetcat insidehosts.set --count
```

```
rwsetcat insidehosts.set | less
```

```
ls -l outsidehosts.set
```

```
rwfileinfo outsidehosts.set
```

```
rwsetcat outsidehosts.set --count
```

```
rwsetcat outsidehosts.set | less
```

Exercise 12 Sets

Which /16 networks are on the inside?

Which /8 networks are on the outside?

Bonus question

How many /24 networks are on the outside?

Exercise 12 Sets

Which /16 networks are on the inside?

Which /8 networks are on the outside?

Bonus question

How many /24 networks are on the outside?

HINT

Use `--network-struct=N`

Exercise 12 Sets

Which /16 networks are on the inside?

Which /8 networks are on the outside?

Bonus question

How many /24 networks are on the outside?

HINT

Use `--network-struct=N`

Where N comes from

CIDR notation /N

Exercise 12 solution

```
rwsetcat --network-struct=16 insidehosts.set
```

```
rwsetcat --network-struct=8 outsidehosts.set
```

Bonus question

```
rwsetcat --network-struct=24 outsidehosts.set \  
| wc -l
```

Set-like Files: Bags

Wouldn't it be nice to count something per address and associate the two?

Yes, it would, it exists and it is called a Bag

- rwbag
- rwbagbuild
- rwbagcat
- rwbagtool

Bag Example

```
rm -f sf.bag
rfilter --type=out,outweb \
        --sensor=S0 \
        --start=2009/4/20 \
        --proto=0- \
        --pass=stdout \
| rwbag --sip-flows=sf.bag
```

```
$ rwbagcat sf.bag
 10.1.60.4|          20|
 10.1.60.5|        3155|
 10.1.60.25|       182|
 10.1.60.53|         1|
 10.1.60.73|       171|
 10.1.60.74|         1|
 10.1.60.153|        11|
 10.1.60.187|      1045|
 10.1.60.191|       250|
 10.1.60.251|      115|
 10.1.60.253|        12|
212.117.116.35|         8|
212.117.116.36|        28|
212.117.116.38|         2|
```

Set-like Files: Prefix Map (PMap)

How do I work with, say, service names like HTTP and HTTPS rather than 80 and 443?

Use a PMap, short for Prefix Map

- `rwpmapbuild`
- `rwpmapcat`
- `rwpmaplookup`

Pmap Example

```
rwfilter --sensor=S0 \
        --start=2009/4/21 \
        --proto=0- \
        --pass=stdout \
| rwuniq --pmap-file=pname:/data/bluered/protocols.pmap \
        --fields=src-pname,proto \
        --values=bytes --sort-out
```

src-pname	pro	Bytes
ICMP	1	17228
TCP	6	53954032
UDP	17	1175172

Set-like Files: Tuple

Is there a way to search for multiple, independent field values without resorting to multiple `rwfilter` commands?

Yes, it is called a Tuple and it can be used in addition to or instead of other partitioning parameters (use it instead of, say, `proto=6 dport=25,58,143,158,209,366,465,587`)

```
rwfilter ... -tuple-file=email-ports.txt ...
```


Comparison of IP Set, Bag, Tuple, PMap

	IP Set	Bag	Tuple	PMap
Semantics	presence	volume	conditionals	categories
Columns	1	2	1–5	2
1 st Column	IP Addr	various	Flow-Label Field	IP Addr or Proto/Port
2 nd Column	—	measure	Flow-Label Field or none	Label
Used for Partitioning	yes	no	yes	yes
Used for Field Output	no	no	no	yes
Binary/Text	binary	binary	text	binary
Combine	set algebra	arithmetic	no	no
Usual Role	input, interim, output	interim, output	input, interim	input



Lesson II.3 Summary



An IP Set is a collection of IP addresses. There are no duplicates in a set. An IP address is either in a given set or it is not.

`rwsetbuild` creates an IP Set from a text file. `rwset` creates an IP Set from flow records.

IP sets can be used for partitioning flow records with `rwfilter`.

`rwsetcat` displays the contents or summaries of an IP Set.

Part II Summary—Basic SiLK

- rwsiteinfo
- rwcut
- rwfilter
- rwcount
- rwstats
- rwuniq
- rwsetbuild
- rwsetcat
- rwset
- rwsetmember

A look ahead

There is more to the SiLK Analysis Tool Suite than the above

- TCP Flags
- Application Label
- PySiLK
- Plug-ins
- Rayon

The Analyst's Handbook is a great resource for learning more

<http://tools.netsa.cert.org/silk/analysis-handbook.pdf>

As is the report: Network Profiling Using Flow

<http://www.sei.cmu.edu/reports/12tr006.pdf>

 Software Engineering Institute | Carnegie Mellon University

Resources

<http://tools.netsa.cert.org/silk/docs.html>

  Software Engineering Institute | Carnegie Mellon University

CERT NetSA Security Suite

Monitoring for Large-Scale Networks

SILK

Documentation

SILK

- Documentation
- Downloads
- Release Notes
- FAQ
- License
- Credits
- Reference Data

Contents

- **Analysis Suite** - overview of the analysis tools
 - Filtering, displaying, and sorting
 - SILK Python extension (PySILK)
 - Counting, grouping, and mating
 - IPset, Bag, and Prefix Map manipulation
 - IP and port labeling files
 - Run-time plug-ins
 - Packet and IPFIX processing
 - Scan detection
 - Flow file utilities
 - Utilities
- **Packing System** - summary of the SILK packing tools
- **Configuration and Overview** - brief description of SILK's configuration files
- **Analysis Handbooks and References** - documents and additional references describing the analysis tools
- **Installation Information** - information for installing and configuring SILK
- **Alphabetized Index of Manuals** - links to all of the SILK manual pages

Analysis Suite

The SILK analysis suite is a collection of command-line tools for processing SILK Flow records created by the [SILK packing system](#). These tools read binary files containing SILK Flow records and partition, sort, and count these records. The most important analysis tool is [rfilter](#), an application for querying the central data repository for SILK Flow records that satisfy a set of filtering options. The tools are intended to be combined in various ways to perform an analysis task. A typical analysis uses UNIX pipes and intermediate data files to share data between invocations of the tools.

The tools and plug-in modules that make up the analysis tools are listed below, roughly grouped by functionality.

Filtering, displaying, and sorting


rfilter	Select SILK Flow records from the data repository and partition the records into one or more 'pass' and/or 'fail' output streams.
rwcut	Print the attributes of SILK Flow records in a delimited, columnar, human-readable format. Users can define new printable attributes using plug-ins written in C or PySILK.
rwsort	Sort SILK Flow records using a user-specified key comprised of record attributes, and write the records to the named output path or to the standard output. Users can define new key fields using plug-ins written in C or PySILK.

SILK Python Extension (PySILK)

PySILK: SILK in	Read, manipulate, and write SILK Flow records, IPsets, and Bags from within Python.
---------------------------------	---

Resources

<http://tools.netsa.cert.org/>



Software Engineering Institute | Carnegie Mellon University

CERT NetSA Security Suite

Monitoring for Large-Scale Networks


Projects

- [Analysis Pipeline 4.4.1](#)
- [Analysis Pipeline 5.3](#)
- [fixbuf 1.7.1](#)
- [IPA 0.5.2](#)
- [iSILK 0.6.2](#)
- [netsa-python 1.4.3](#)
- [Orcus 1.0.3](#)
- [pyfixbuf 0.2.0](#)
- [Rayon 1.4.3](#)
- [schemaTools 1.2](#)
- [SiLK 3.11.0.1](#)
- [SiLK IPset 3.11.0](#)
- [snarf 0.2.4](#)
- [super_mediator 1.1.3](#)
- [YAF 2.7.1](#)

Scripts

- [Prism 1.2](#)
- [Cif2Stix 1.0](#)
- [Stix2Cif 1.0](#)

Links

- [Public Mailing List](#)
- [Wiki](#)
- [Tooltips](#)
- [Live DVD](#)
- [Defunct Projects](#)
-  [Releases RSS Feed](#)

The Network Situational Awareness (NetSA) group at CERT has developed and maintains a suite of open source tools for monitoring large-scale networks using flow data. These tools have grown out of the work of the AirCERT project, the SiLK project and the effort to integrate this work into a unified, standards-compliant flow collection and analysis platform.

CERT is a part of the [Software Engineering Institute \(SEI\)](#), a federally funded research and development center (FFRDC) operated by [Carnegie Mellon University](#).

Featured Projects

SiLK 3.11.0.1

[Download Now](#)

The System for Internet Level Knowledge (SiLK) is an efficient network flow collection and storage infrastructure that will accept flow data from a variety of sensors. SiLK also provides a suite of efficient command-line tools for analysis.

YAF 2.7.1

[Download Now](#)

Yet Another Flow Sensor (YAF) processes packet data into bidirectional flow records that can be used as input to an IPFIX Collecting Process. YAF's output can be used with the NetSA Aggregated Flow (NAF) toolchain and the SiLK tools.

Analysis Pipeline 5.3

[Download Now](#)

The Analysis Pipeline 5.3 is a streaming analysis tool than can process more than just SiLK flows as done in version 4.x. It can now process YAF records and raw IPFIX records. It can do all of the analyses available in version 4.x. A notable enhancement is expansive DNS record processing. This includes fast flux detection and domain name watchlisting.

super_mediator 1.1.3

[Download Now](#)

super_mediator is an IPFIX mediator for use with the YAF and SiLK tools. It collects and filters YAF output data to various IPFIX collecting processes and/or csv files. super_mediator can be configured to perform de-duplication of DNS resource records as exported by YAF.

© 2006–2015 Carnegie Mellon University netsa-help@cert.org

Questions?



Contact Information

Paul Krystosek — `pnk@cert.org`

Matt Heckathorn — `maheckathorn@cert.org`

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA

FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics

Extra slides

How well does compression work?

```
FILE=compress-none.rw
for RECS in 1 2 3 4 5 6 7 8 9 10 11
do
    rm -r $FILE
    rfilter --type=all \
            --start-date=2009/4/20:11 \
            --protocol=0- \
            --compress=none \
            --max-pass=$RECS \
            --pass=$FILE
    ls -l $FILE
done
```

No compression vs compression

```
-rw-r--r--. 1 pnk pnk 264 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 352 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 440 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 528 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 616 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 704 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 792 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 880 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 968 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 1056 Jan 5 22:02 compress-none.rw  
-rw-r--r--. 1 pnk pnk 1144 Jan 5 22:02 compress-none.rw
```

```
-rw-r--r--. 1 pnk pnk 256 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 272 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 282 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 290 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 298 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 303 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 324 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 333 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 344 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 355 Jan 5 22:02 compress-best.rw  
-rw-r--r--. 1 pnk pnk 364 Jan 5 22:02 compress-best.rw
```

FloCon 2016

12th Annual Open Forum for
Large-Scale Network Analytics

Visualization



Flow Visualization

Visualization has many uses

- Analysis
- Explanation
- Discovery

One of the best results of visualization
is to speed up whatever you are doing

Popular types of visualization:

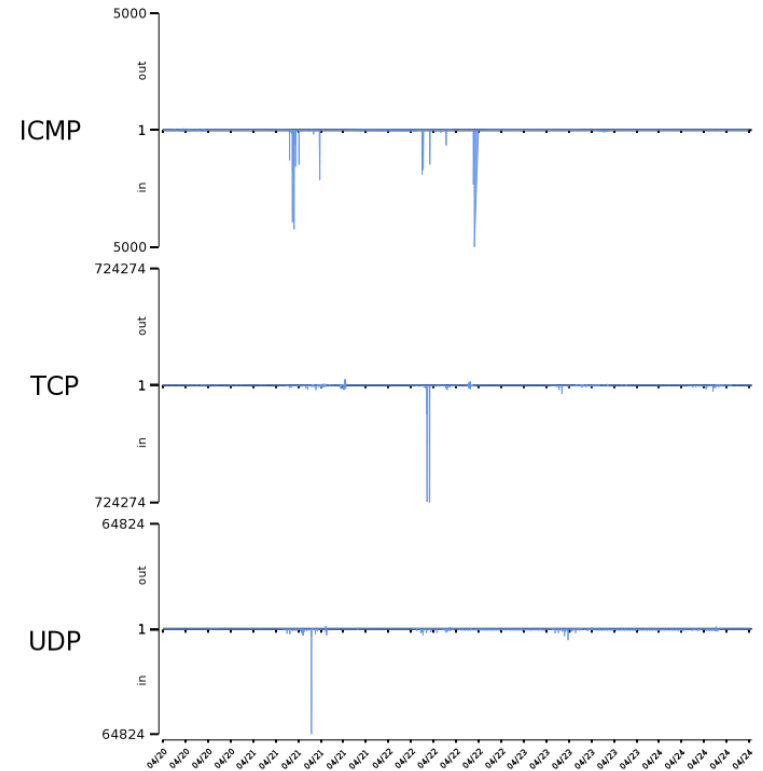
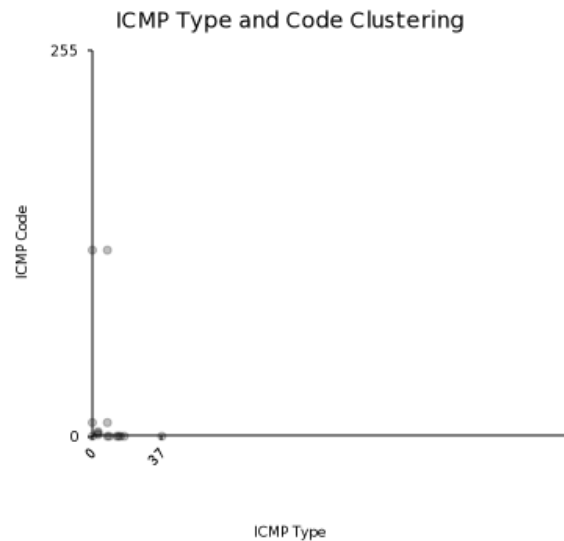
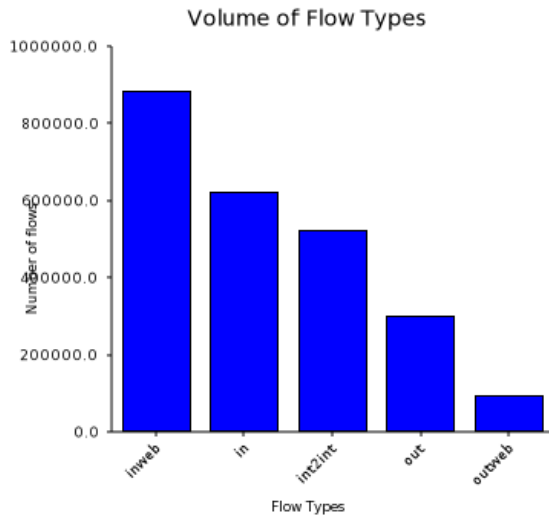
- Bar Chart
- Time Series
- Scatter Plot
- Histogram
- Link diagram (directed graph)
- Heat Map
- Other
 - Timelines
 - Geographic maps
 - Pie charts

Software to do visualization: Rayon

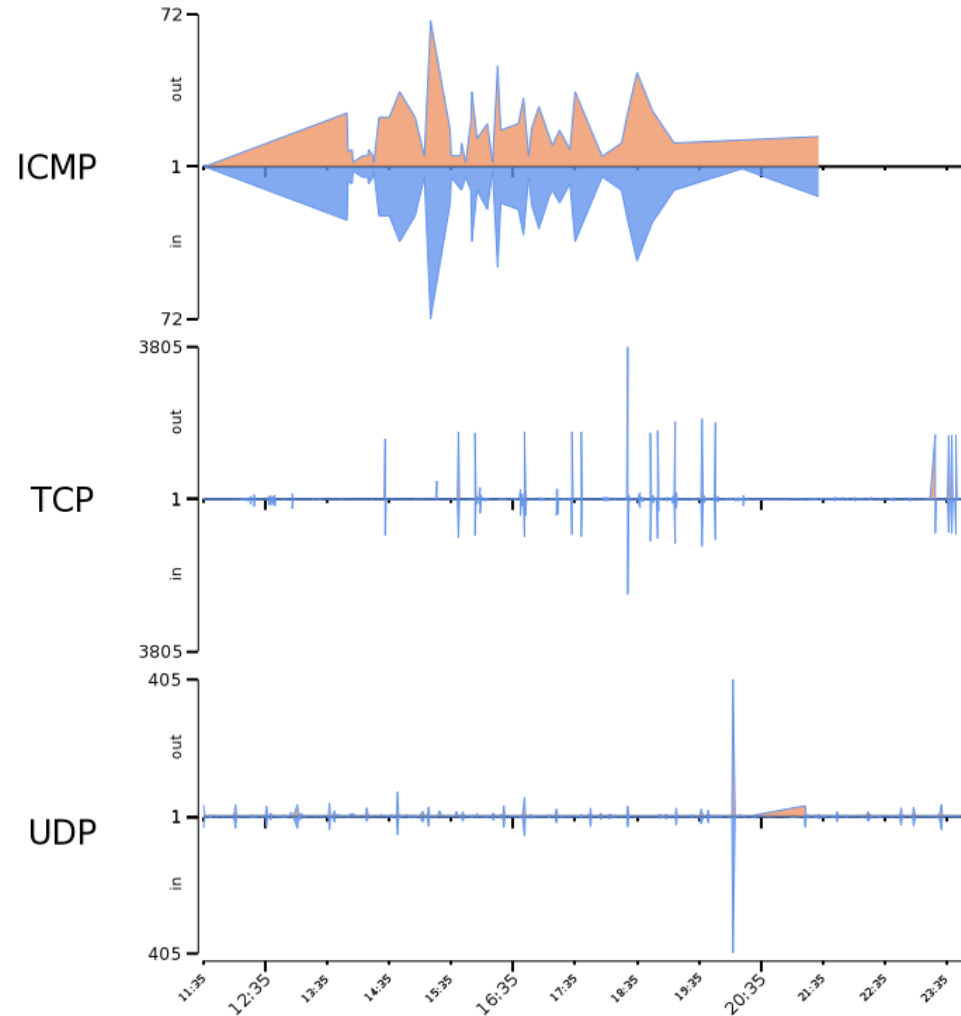
- Rayon was written to work and play well with SiLK and Python
- It fits in with the Unix pipe mode of scripting
- It doesn't (yet?) handle everything we want to do

Viz Type	Application
Bar Chart	rycategories
Time Series	rytimeseries
Scatter Plot	ryscatterplot
Histogram	rycategories
Link diagram	GraphViz
Heat Map	ryhilbert

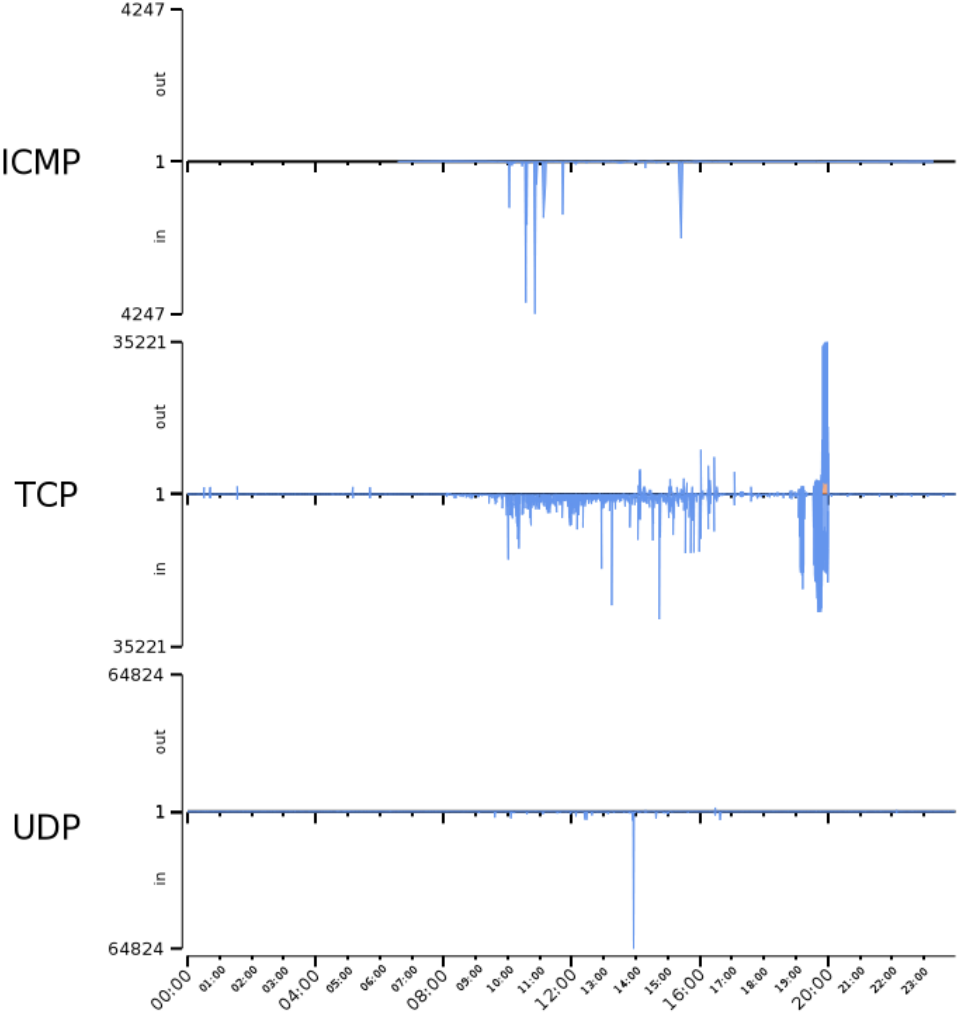
What does our data look like?



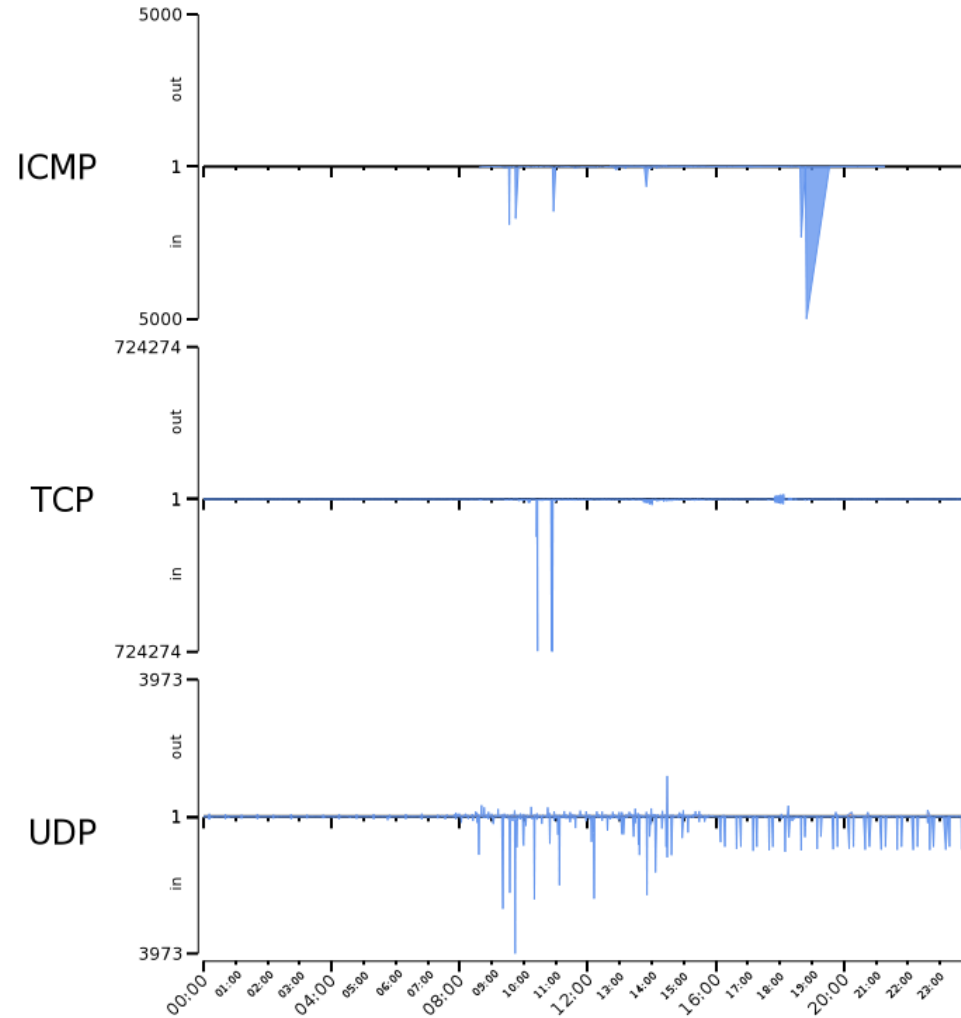
Lets take a closer look 2009/04/20



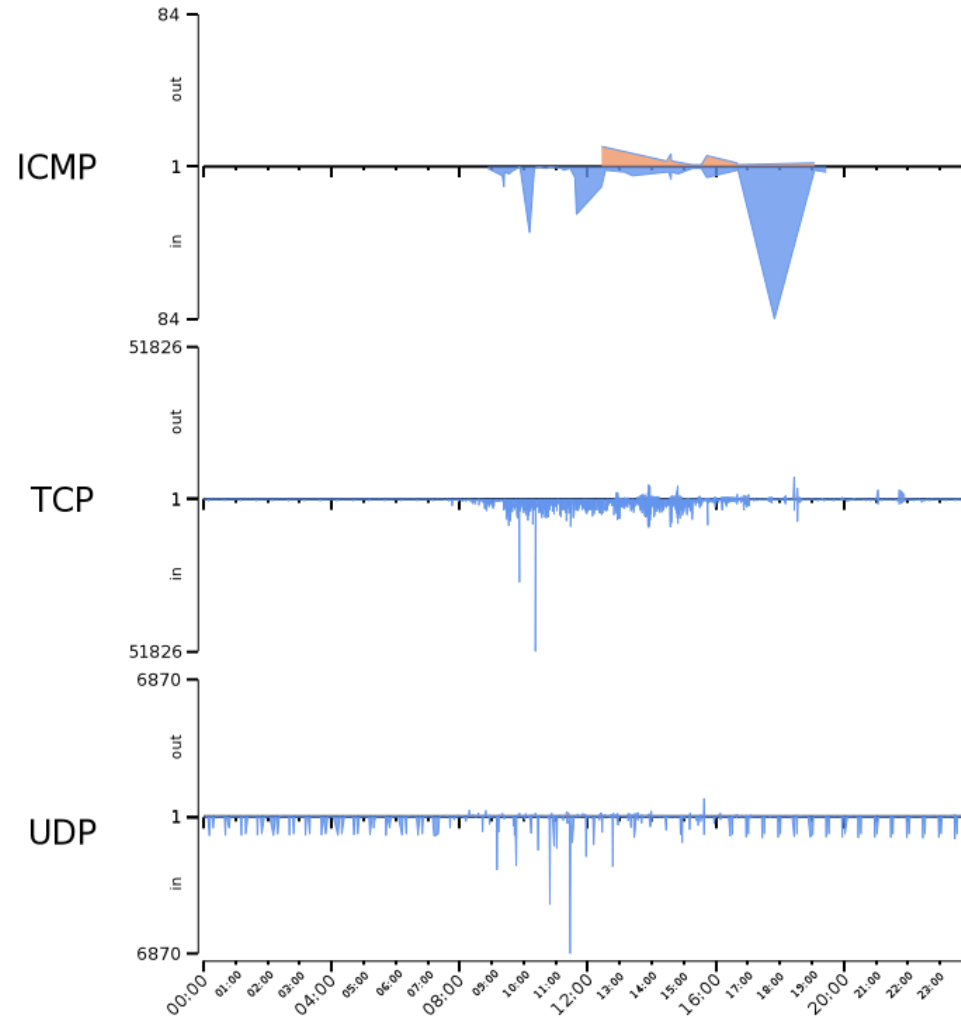
Lets take a closer look 2009/04/21



Lets take a closer look 2009/04/22



Lets take a closer look 2009/04/23



Lets take a closer look 2009/04/24

