Intrusion Detection - Snort

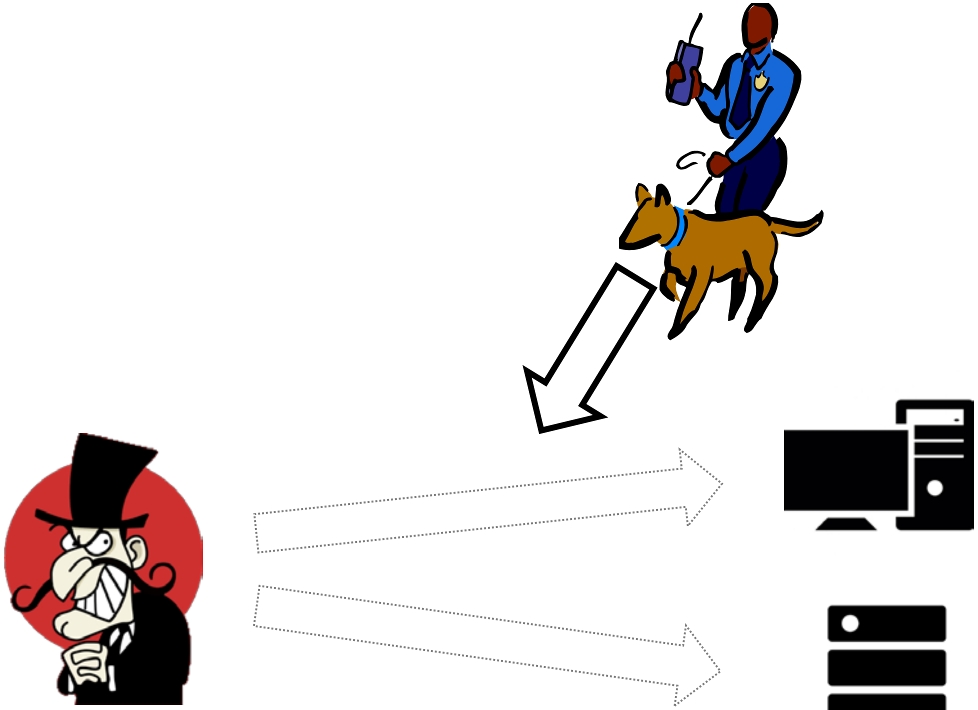
Sometimes - Defenses Fail

- Our defenses aren’t perfect
  - Patches aren’t applied promptly enough
  - AV signatures not up to date
  - 0-days get through
  - Someone brings in an infected USB drive
  - An insider misbehaves

- Most penetrations are never detected
  - This allows continuing abuse, and helps the attackers spread elsewhere

Additional Monitoring

- Prevention is ideal, but DETECTION is a must!
  - Offense leads defense!



What can IDS possibly do?

- Detect strange patterns or behaviors
- Detect things that should not be there
  - abnormalities
- Help contain attacks before they spread
- Match activities against known attacks
- Classify good or bad traffic
  - tuning

What IDS cannot do?

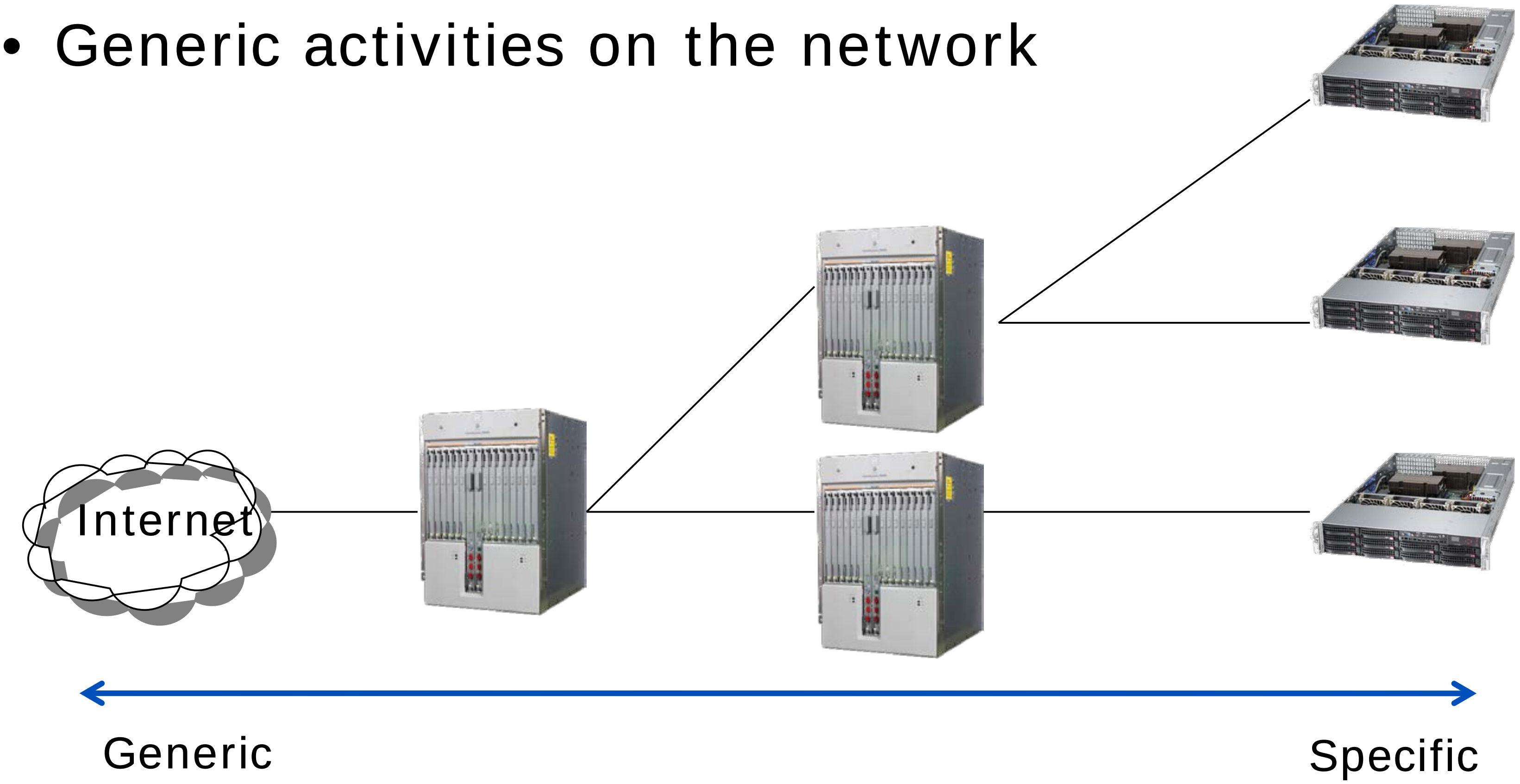
- Compensate for
  - weak authentication and identification mechanisms
  - weakness in network protocols or configuration errors

- Investigate attack patterns without human intervention

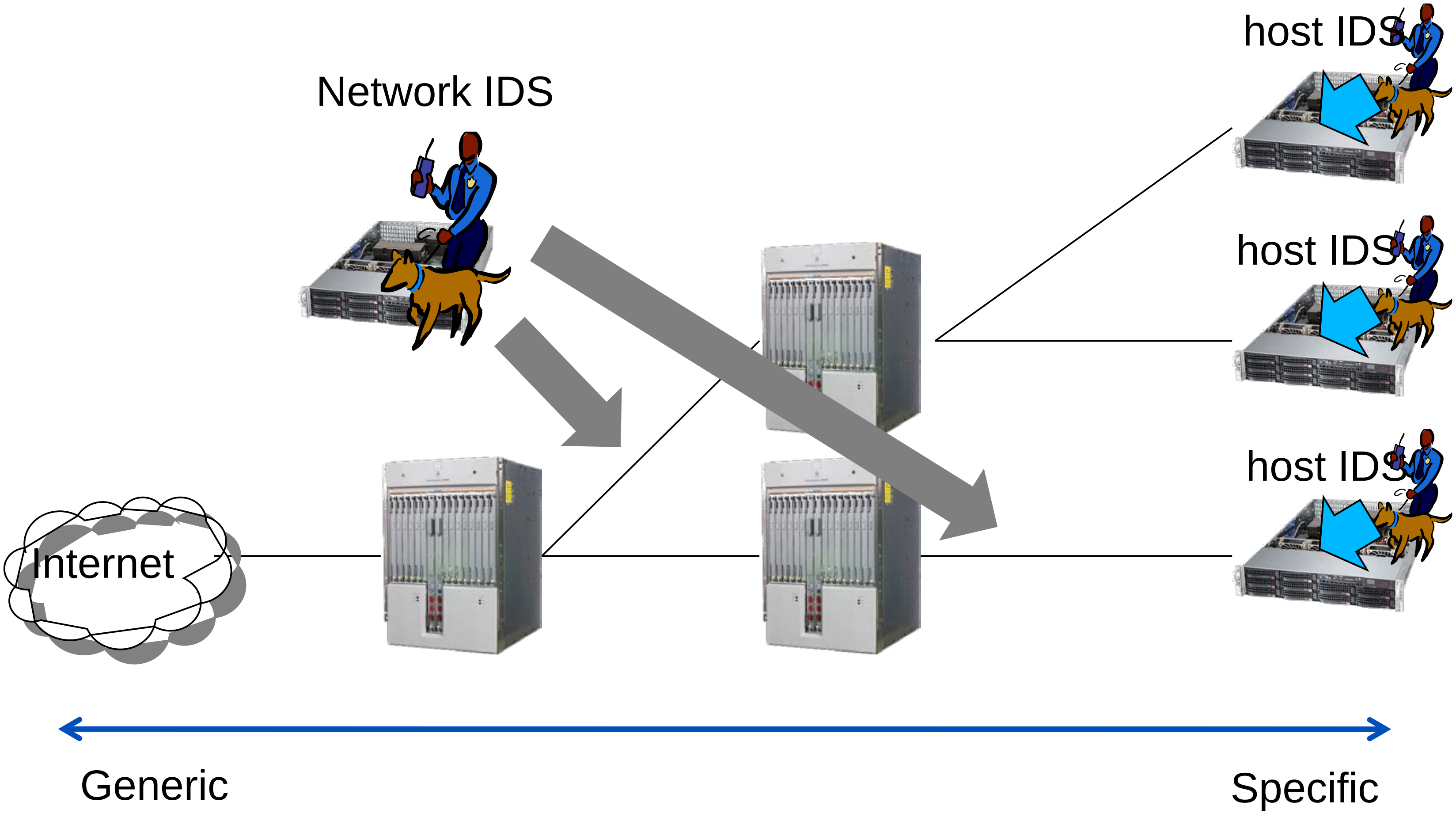
- Guess your organization’s security policy

Monitoring Point

- Specific rules closer to the end hosts/nodes
- Generic activities on the network



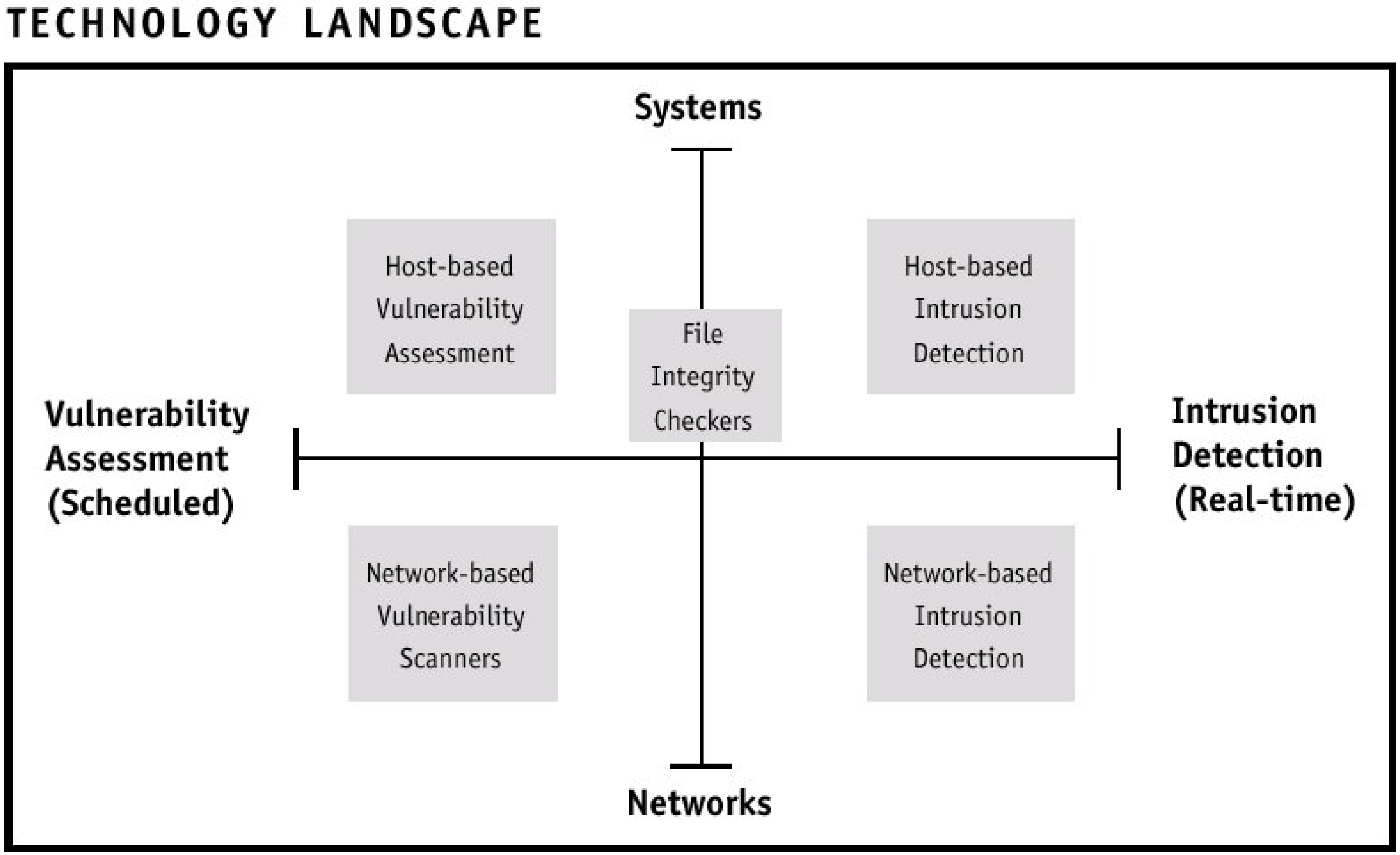
Network and Host IDS



Types of Detection

- Signature based
  - Match patterns/characteristics of known attacks
    - Signatures need to be updated and only known issues detected
- Anomaly based
  - Look for any unusual behaviour
    - Anything that deviates from what is considered normal
- Darknet
  - Monitor inbound traffic to unlit (dark) IPs
    - Why?
- Honeypot
  - Set a trap!
  - Its value lies in being being compromised
    - Log any activity and setup triggers/notifications
    - Helps understand an attacker's methodology, identify vulnerabilities

IDS Technology landscape



TECHNOLOGY LANDSCAPE

Systems

Host-based Vulnerability Assessment

File Integrity Checkers

Host-based Intrusion Detection

Vulnerability Assessment (Scheduled)

Intrusion Detection (Real-time)

Network-based Vulnerability Scanners

Network-based Intrusion Detection

Networks

Preventive

Real Time

Alert

- Depending on how you tune your detection engine/rules
  - You may receive millions of alerts (too strict)
  - You may miss out on critical events (too loose)

Alert

- False-positive
  - System raising an incorrect alert
  - Incorrect rejection of a true null hypothesis
- False-negative
  - Does not detect an attack
  - Failure to reject a false null hypothesis

Intrusion Detection for ISPs

- Monitor your own network
- Monitor your customer networks
  - Good:
    - you can help them detect problems and prevent malicious traffic clogging your network infra
  - Bad:
    - privacy-invasive

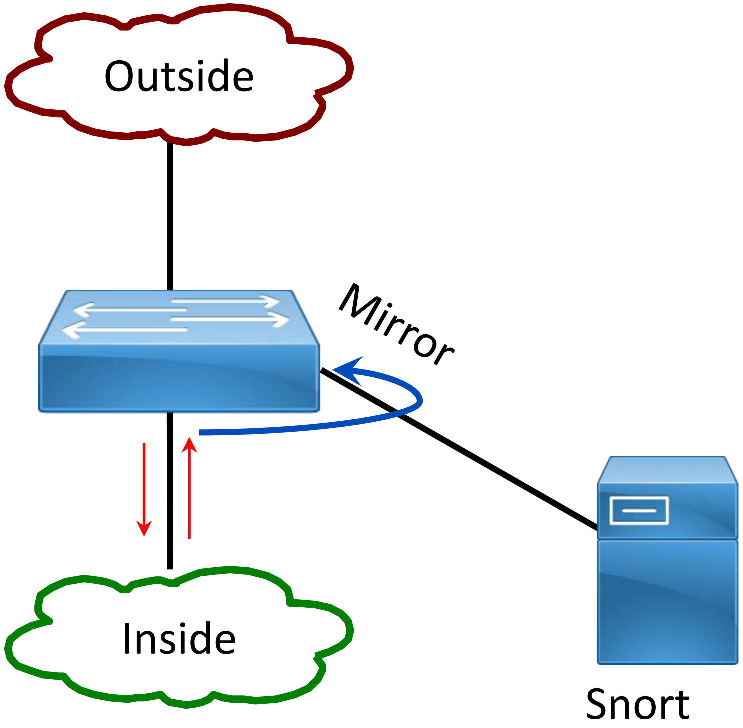
SNORT

- Open source IDS (one of the oldest ones)
  - Hundreds of thousands of users
- Active development of rules by the community
  - Upto date (often more than commercial alternatives)
- It is fast
  - With the right HW and proper tuning

Getting Snort to see the network

- You can run Snort in multiple ways
  - In-line (behind firewalls)
    - Could help test your firewall rules
    - But, one more element that could fail (single point?)
  - In-line (in front of firewalls)
    - Too many alerts!
  - SPAN/mirror traffic to Snort
  - Tap on the physical link (optical splitter)

Port Mirroring



Getting Snort to see the network

- Be careful not to overload switch port
  - You do not want to mirror multiple gigabit ports to a single GE port
  - Could drop packets if the traffic exceeds 1Gbps

Port Mirroring

- You can mirror
  - one port to another,
  - a group ports to one port
  - An entire VLAN to a port

Example: Cisco Catalyst

```
(config)#monitor session <sess#> source <int-ID/VLAN-id>
(config)#monitor session <sess#> destination <int-ID/VLAN-id>
```

Snort configuration file

- By default: /etc/snort/snort.conf
  - A long file (900+ lines of code)
  - Many **pre-processor** entries
    - Snort pre-processors help examine packets for suspicious activities, or
    - Modify them to be interpreted correctly by the detection rules (processor codes are run before detection engine is called)

SNORT Rules

- Snort rules are plain-text files
- Adding new rules is as easy as dropping the files to /etc/snort/rules directory
- Rules can be loaded from **snort.conf** with the "*include*" statement
- Rules can match anything
  - Technical: port scans, web attacks, buffer overflow, etc.
  - Policies: URL filters, keywords, etc.

Tailoring the rules

- Not all rules (default) will be applicable to your network
  - You customise/pick which rules you want to run
  - Else, to many false positives or to many alerts
    - Might tempt you to ignore the alerts or even turn it off

- You can suppress/disable rules you don’t need

Updating Snort rules

- Commercially maintained (Cisco) Snort rules are available for free after 30 days delay
  - http://www.snort.org/start/rules

- Volunteers also maintain rule sets
  - http://rules.emergingthreats.net/open/

- You can automate updating of rules using "Pulled Pork"
  - http://code.google.com/p/pulledpork/

Snort rules

- Snort rules have two sections
  - Rule Header and Rule options
- Rule header contains
  - the rule's action, protocol, src/dst addresses, and src/dst ports information
- Rule options contain
  - alert messages and information on which parts of the packet should be inspected for the action to be taken
  - http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node28.html

Snort rules

```
action protocol ip-addr port -> ip-addr port (rule
option1; option2)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22
(msg: "SSH Detected"; sid:10; rev:1;)
```

- The text up to first parenthesis – rule header
- Enclosed in parenthesis – rule options
  - words before colons in the options are called ”option keywords”

Snort Rule actions

- **alert** - generate an alert using the selected alert method, and then log the packet
- **log** – log the packet
- **pass** – ignore the packet
- **drop** – block and log the packet
- **reject** – block the packet, log it, and send TCP reset if protocol is TCP, or an ICMP port unreachable if it is UDP
- **sdrop** – block the packet without logging

Snort rules : direction

•The direction operator **->** indicates the orientation, or direction, of the traffic that the rule applies to.

•There is no **<-** operator.

•Bidirectional operator **<>**

Snort rules : sid

- The Snort ID (sid):
  - Uniquely identifies snort rules (similar to ACL numbers)
    - 0-99 reserved for future use
    - 100-1,000,000 reserved for rules in Snort distribution
    - >1,000,000 can be used to define local rules

Snort rules : rev

- The revision number (rev)
  - Allows rules to be refined and updated

Snort rules : classtype

- Rules can be classified and assigned priority numbers
  - to group and distinguish them (low and high priority alerts)
  - Priorities 1-4 (High, Medium, Low, very low)

- Attack classifications defined by Snort resides in `/etc/snort/classification.config`

```
config classification: DoS, Denial of Service Attack, 2
```

Class Name   Class Description   Priority

Sample rules

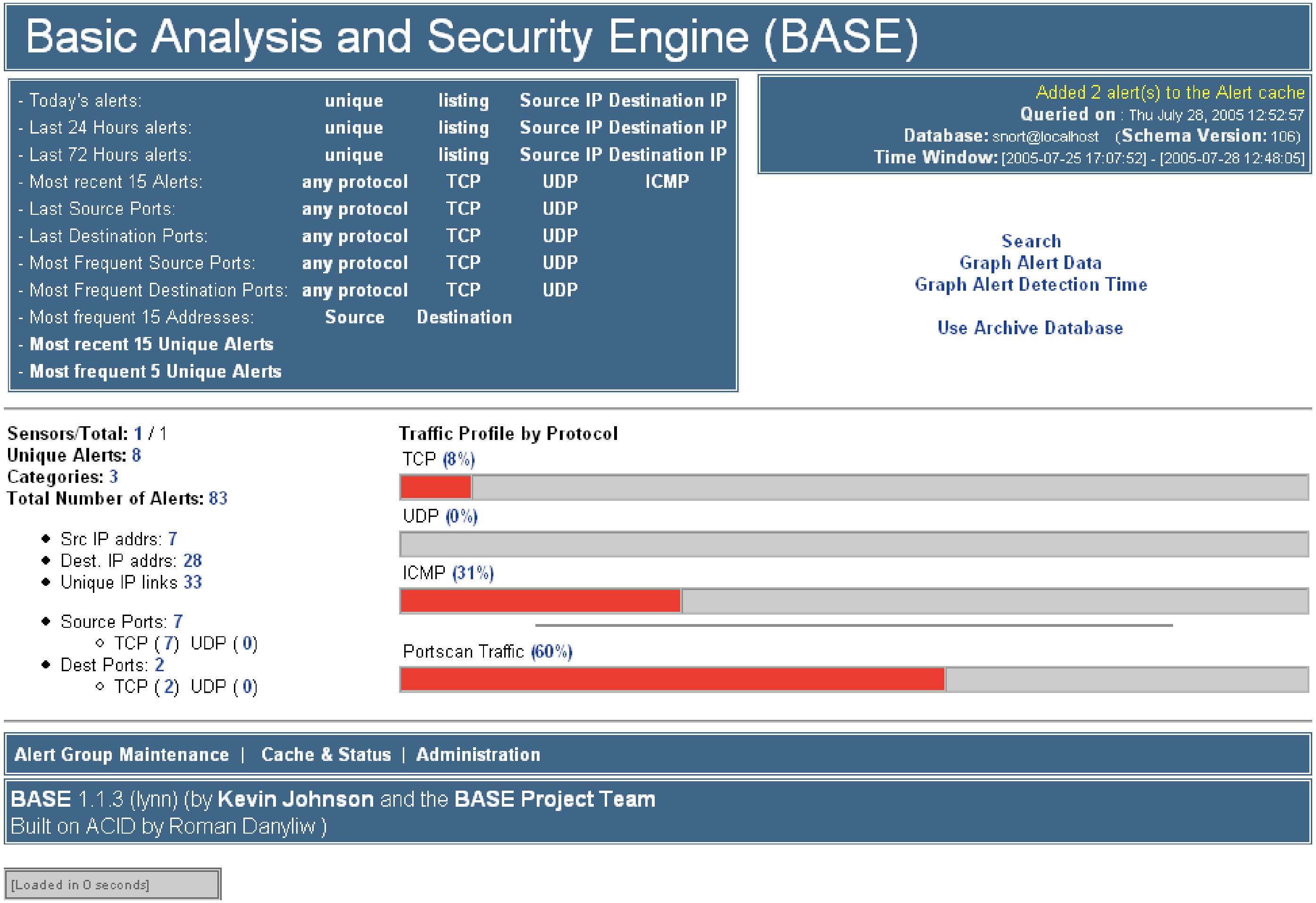
```
alert tcp msg:"MYSQL root login attempt";
flow:to_server,established; content:"|0A 00 00 01 85 04 00 00
80|root|00|"; classtype:protocol-command-decode; sid:1775;
rev:2;)

alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 3306 (msg:"MYSQL
show databases attempt"; flow:to_server,established;
content:"|0F 00 00 00 03|show databases"; classtype:protocol-
command-decode; sid:1776; rev:2;)

alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 3306 (msg:"MYSQL
4.0 root login attempt"; flow:to_server,established;
content:"|01|"; within:1; distance:3; content:"root|00|";
within:5; distance:5; nocase; classtype:protocol-command-
decode; sid:3456; rev:2;)
```

Reporting and logging

- Snort can be made to log alerts to an SQL database, for easier searching

- A web front-end for Snort, **BASE**, allows one to browse security alerts graphically

BASE - Basic Analysis and Security Engine



Basic Analysis and Security Engine (BASE)

- Today's alerts:	unique	listing	Source IP	Destination IP
- Last 24 Hours alerts:	unique	listing	Source IP	Destination IP
- Last 72 Hours alerts:	unique	listing	Source IP	Destination IP
- Most recent 15 Alerts:	any protocol	TCP	UDP	ICMP
- Last Source Ports:	any protocol	TCP	UDP	
- Last Destination Ports:	any protocol	TCP	UDP	
- Most Frequent Source Ports:	any protocol	TCP	UDP	
- Most Frequent Destination Ports:	any protocol	TCP	UDP	
- Most frequent 15 Addresses:	Source	Destination		
- **Most recent 15 Unique Alerts**				
- **Most frequent 5 Unique Alerts**				

Added 2 alert(s) to the Alert cache
Queried on : Thu July 28, 2005 12:52:57
Database: snort@localhost (**Schema Version:** 106)
Time Window: [2005-07-25 17:07:52] - [2005-07-28 12:48:05]

Search
Graph Alert Data
Graph Alert Detection Time

Use Archive Database

Sensors/Total: 1 / 1
Unique Alerts: 8
Categories: 3
Total Number of Alerts: 83

- Src IP addrs: 7
- Dest. IP addrs: 28
- Unique IP links 33

- Source Ports: 7
  - TCP (7) UDP (0)
- Dest Ports: 2
  - TCP (2) UDP (0)

Traffic Profile by Protocol

TCP (8%)

UDP (0%)

ICMP (31%)

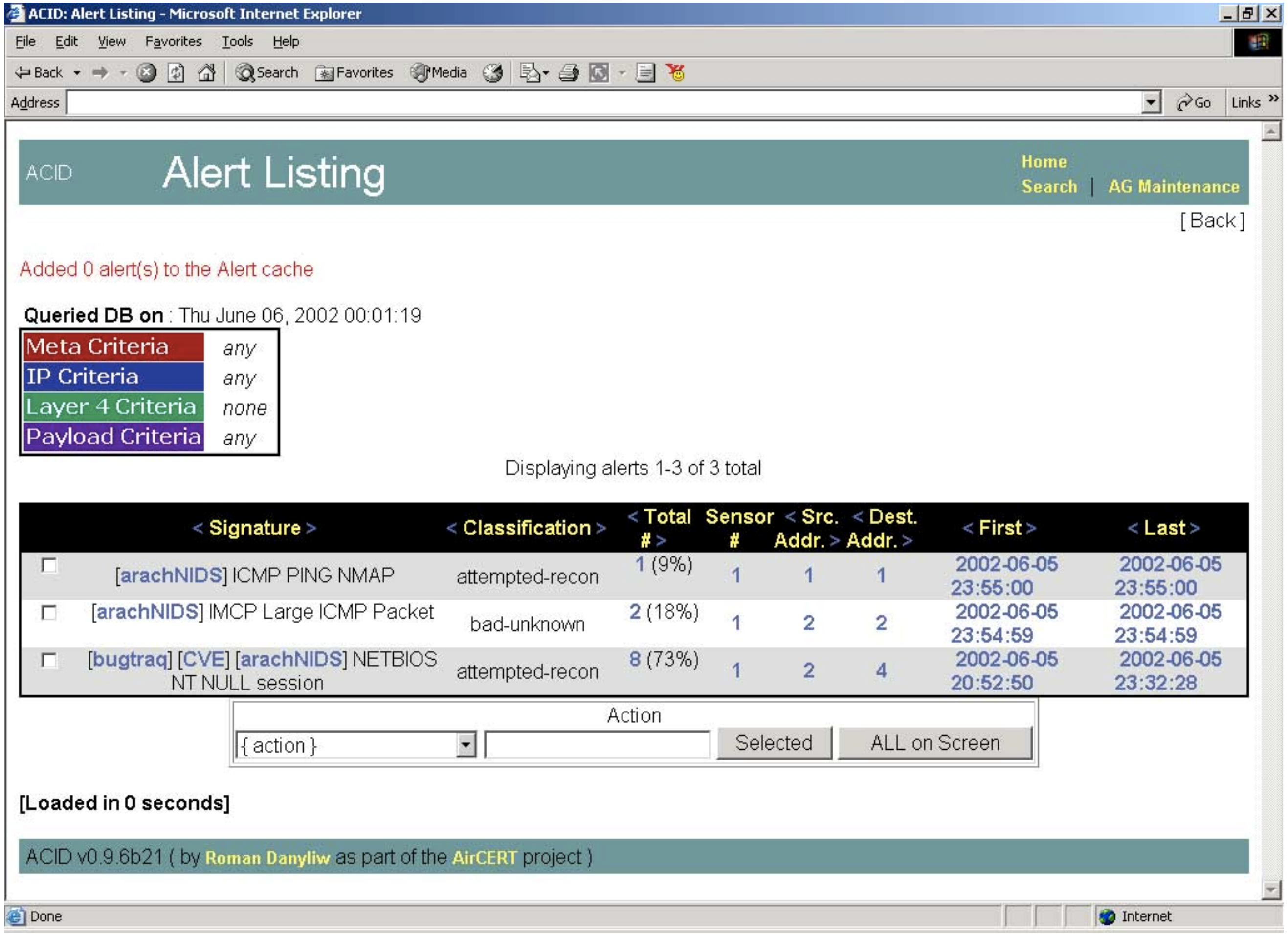
Portscan Traffic (60%)

Alert Group Maintenance | **Cache & Status** | **Administration**

BASE 1.1.3 (lynn) (by **Kevin Johnson** and the **BASE Project Team**
Built on ACID by Roman Danyliw)

[Loaded in 0 seconds]

BASE - Basic Analysis and Security Engine


ACID: Alert Listing - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go Links
Done
Internet

ACID **Alert Listing**

Home
Search | AG Maintenance

[Back]

Added 0 alert(s) to the Alert cache

Queried DB on : Thu June 06, 2002 00:01:19

Meta Criteria	*any*
IP Criteria	*any*
Layer 4 Criteria	*none*
Payload Criteria	*any*

Displaying alerts 1-3 of 3 total

	< Signature >	< Classification >	< Total # >	Sensor #	< Src. Addr. >	< Dest. Addr. >	< First >	< Last >
□	[arachNIDS] ICMP PING NMAP	attempted-recon	1 (9%)	1	1	1	2002-06-05 23:55:00	2002-06-05 23:55:00
□	[arachNIDS] IMCP Large ICMP Packet	bad-unknown	2 (18%)	1	2	2	2002-06-05 23:54:59	2002-06-05 23:54:59
□	[bugtraq] [CVE] [arachNIDS] NETBIOS NT NULL session	attempted-recon	8 (73%)	1	2	4	2002-06-05 20:52:50	2002-06-05 23:32:28

Action

{ action } ▾ | Selected | ALL on Screen

[Loaded in 0 seconds]

ACID v0.9.6b21 (by Roman Danyliw as part of the AirCERT project)

References and documentation

- Snort preprocessors:
  - http://www.informit.com/articles/article.aspx?p=101148&seqNum=2
- Snort documentation
  - https://www.snort.org/documents#OfficialDocumentation
- Writing SNORT Rules
  - http://manual.snort.org/node27.html

Lab Exercise

Setup

- Follow lab manual to install SNORT and check the basic SNORT rules.

Exercise : 1

- Write a rule to detect XMAS scans against your server
  - XMAS scan sets the FIN, PSH, URG flags
  - Check the rules with nmap

```
nmap -sX <SERVER_IP>
```

RFC 793 - any TCP segment with an out-of-state Flag sent to an open port is discarded, whereas segments with out-of-state flags sent to closed ports should be handled with a RST in response.
*> **Allows an attacker to scan for closed ports by sending certain types of rule-breaking packets and detect closed ports via RST packets**"* – https://capec.mitre.org/data/definitions/303.html

Exercise : 2

- Write a rule to detect any attempt from outside (external) your network to access your webserver’s **/admin** pages

Content Matching

Exercise : 3

- Write a rule to check SSH brute force attack and log the IP (more than 3 times in 60 seconds)

```
detection_filter:track by_src, count 3, seconds 60;
```