

# Intrusion Detection

March 9, 2012

## Administrative – submittal instructions

- answer the lab assignment's questions in written report form, as a text, pdf, or Word document file (no obscure formats please)
- email to [csci530l@usc.edu](mailto:csci530l@usc.edu)
- exact subject title must be "idslab"
- deadline is start of your lab session the following week
- reports not accepted (zero for lab) if
  - late
  - you did not attend the lab (except DEN or prior arrangement)
  - email subject title deviates

## Spring break due date adjustments

- firewalls

- lecture 3/2

- due ~~3/16~~ 3/23 instead

- intrusion detection

- lecture 3/9

- lab ~~3/16~~ 3/23 instead

- due ~~3/23~~ 3/30 instead

## IDS – intrusion detection systems

- **monitor** activities and circumstances
- **respond**/react when “wrong”

## IDS vs firewall

- firewall – fence/gate/door/lock
  - does not react, static
  - preventative intent
  - you lock your door to prevent burglary, you find joe **in the living room!!**<sup>1</sup> stealing<sup>2</sup>
- i.d.s. – burglar alarm / motion detector
  - reacts, dynamic
  - curative intent
  - you admit joe as a party guest, you find joe in the living room<sup>2</sup> **stealing!!**<sup>1</sup>

<sup>1</sup>this is what's wrong

<sup>2</sup>this is not what's wrong

## What's available to monitor?

- log files
  - independently pre-existing
  - rich information source
- system state & behavior pattern
  - must observe & record to define norm first
- network traffic

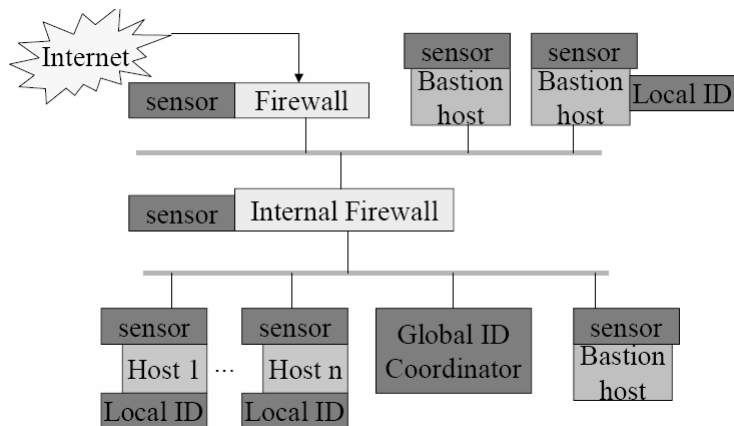
## What are monitor products called?

- log files – log monitors
- state/behavior pattern – integrity checkers
- network traffic – sniffers
  
- i.d.s. combines these functions

## IDS – example responses

- write message to a log destination
- alert system administrator
- add rule to fortify firewall
- disable user account

## Running IDS – where and why



“sensor” = “collector”

## Locus of detection

- distributed detection (at hosts)
  - host analyzes local information
    - local logs/state/traffic
  - host concludes there’s a problem
  - host tells global coordinator there’s a problem
- centralized detection (at global coordinator)
  - hosts send local information to global coordinator
  - global coordinator analyzes received information
    - received logs/state/traffic
  - global coordinator concludes there’s a problem

## Focus-of-detection categorizations

- host-based i.d.s. deals with
  - log files
  - system state and behavior pattern
- network based i.d.s. deals with
  - network traffic

## Network based ids' scope

- may be host aware only – about the host
  - examines network traffic of the single host
  - like snort
- may be network aware – about the net
  - examines network traffic of multiple hosts
    - collected from a single host, or
    - collected from multiple hosts and gathered

## Detection technique categorizations

- rule based
  - uses pre-set rules
  - uses pattern matching (packets against rules)
- anomaly based
  - uses pre-set historical state & behavior profiles
  - uses statistical analysis (current state & behavior against profiles)

## Pre-operational groundwork requirements

- rule based
  - identify behavior patterns of known attacks (attack signatures)
  - write and store rules expressing the patterns
  - yardsticks of the abnormal
    - behavior *like* the pattern is “*wrong*”
- anomaly based
  - identify state and behavior patterns of the system
  - write and store profiles expressing the patterns
  - yardsticks of the normal
    - behavior *unlike* the pattern is “*wrong*”

## Rule vs anomaly based detection

	rule-based	anomaly-based
known attacks	good, precise	limited
unknown/un-foreseen attacks	no role	good

## State/behavior criteria examples

- locations of user's login
- times of user's login
- size of command history file

## Corresponding anomalies

- user logs in from unusual place
- user logs in at unusual time
- command history file truncated/shrunk (?falsified?!!)

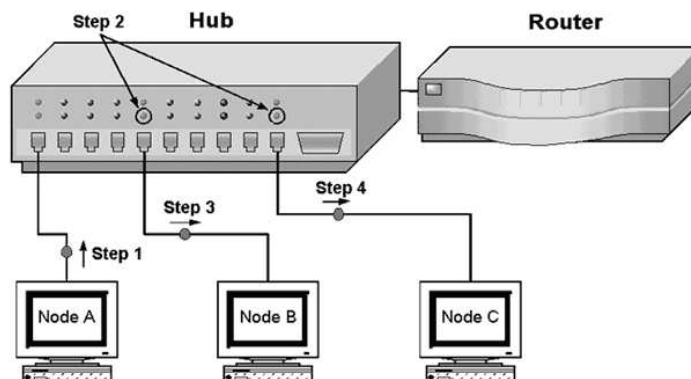
## Detection technique quality factors

- detection rate
- false alarm rate
- detection latency

## Switch vs hub – usual caveat

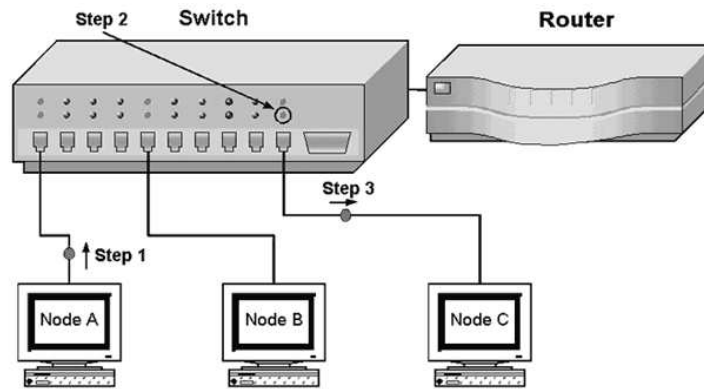
- caveat, for network-wide scope of awareness
- network traffic to an individual host
  - wholesale by hub, host gets all
  - selective by switch, host gets some
- strategies
  - put i.d.s. machine on a hub
  - put i.d.s. machine on a switch's spanning port
  - use global coordinator to gather from multiple hosts

## Hub – B gets A-to-C traffic



see [https://www2.sans.org/resources/idfaq/switched\\_network.php](https://www2.sans.org/resources/idfaq/switched_network.php)

## Switch – B denied A-to-C traffic



B of limited value to run i.d.s. for network-wide awareness

## Network performance – ids vs firewall

- firewall job is lightweight
  - quick binary comparison
- i.d.s. job is heavyweight
  - involved analytical comparison
- i.d.s. subject to overwhelming; dropping packets
- strategies
  - streamlined, non-redundant rulesets
  - fast platform
  - task distribution, global coordination

## Defeating the ids

- fragmentation
  - artificial packet splitting
- spoofing
  - manipulating TCP sequence numbers
- denial of service
  - overwhelm then attack

## What is snort?

“Snort is an open source network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

“Snort uses a flexible rules language to describe traffic that it should collect or pass, as well as a detection engine that utilizes a modular plugin architecture. Snort has a real-time alerting capability as well, incorporating alerting mechanisms for syslog, a user specified file, a UNIX socket, or WinPopUp messages to Windows clients using Samba's smbclient.

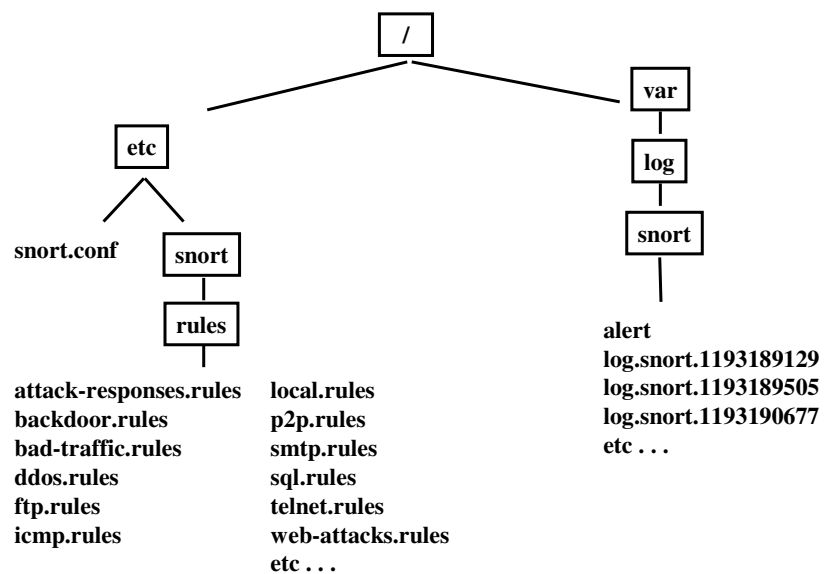
“Snort has three primary uses. It can be used as a straight packet sniffer like tcpdump(1), a packet logger (useful for network traffic debugging, etc), or as a full blown network intrusion prevention system.”

[http://www.snort.org/about\\_snort/](http://www.snort.org/about_snort/)

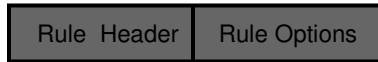
## Snort operational modes

- packet sniffer (-v)
- packet logger (-l)
- network intrusion detector (-c)

## Snort default directories and files



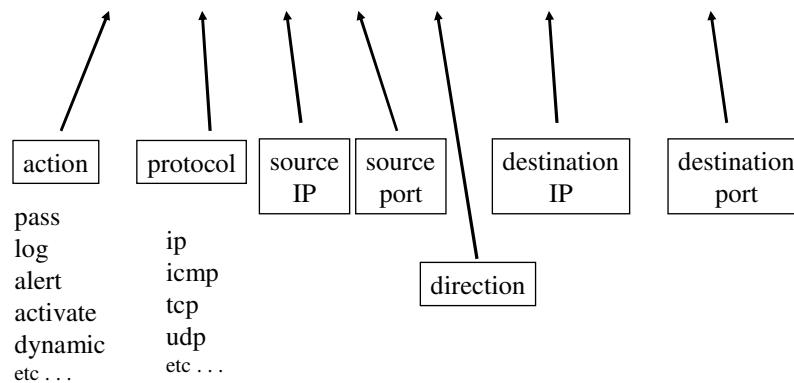
# Rule Structure



**alert tcp any any -> 92.168.1.0/24 111 (content:"100 01 86 a5"; msg:"mountd access");**

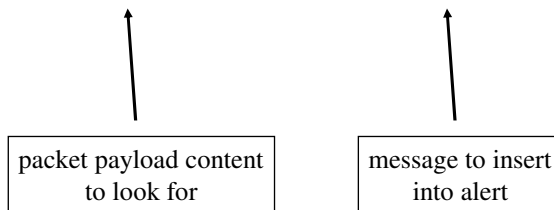
# Rule header

**alert tcp any any -> 92.168.1.0/24 111**



## Rule options

```
(content:"|00 01 86 a5|"; msg:"mountd access");
```



## Snort Rules

a bad, un-useful rule

```
alert ip any any -> any any (msg: "ip packet detected");
```

alert: the action to be performed,

ip : rule applies to all ip packets

any : rule applies to any source ip address

any : rule applies to any source port

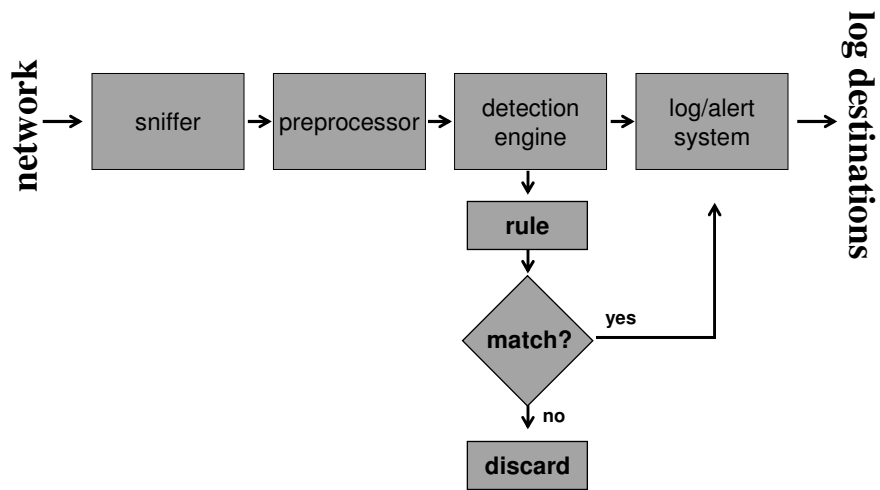
-> : direction of packet

any : rule applies to any destination ip address

any : rule applies to any destination port

writes a lot to /var/log/snort/alert

## Snort architecture



## What is swatch?

**Simple WATCHdog** - file viewer with regexp matching, designed to monitor a file which may contain pattern(s), and to provide action(s) to perform when each pattern is found. Arbitrary file. Arbitrary action.

## What does swatch have to do with snort?

You want to get paged whenever there's a port scan on your network.

snort can detect port scans, can't page  
swatch can page, can't detect port scans

Use snort to watch for port scans and write to a log file when one occurs. Swatch doesn't watch for port scans, it watches for log messages. But if snort reliably issues messages when there's a scan, then swatch is positioned to pick up scans in effect, indirectly, by picking up the messages. Swatch can be the lookout for anything that gets logged-- port scans, bad logins, whatever. It just has to get logged.

## swatch operation

- watches a file (e.g., a log)
- notices designated content
- reacts to its presence
- uses pattern-action specification to do so
  - pattern detection = trigger
  - action = resultant response

## main swatch action keywords

- echo
- mail
- exec
- pipe

**PATTERN: hello.** A line in a file matches if it contains "hello".

watchfor(/hello/)

**ACTIONS: 1) print that line on the terminal in red, and 2) send it out in an email address**

echo=red

mail addresses=you\@isp.com,subject="hello visible"