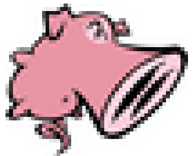
CSC7221 Advanced Topics in Internet Technology

In-Depth Packet Detection and Prevention by Snort: The Open Source Solution of IDS and IPS



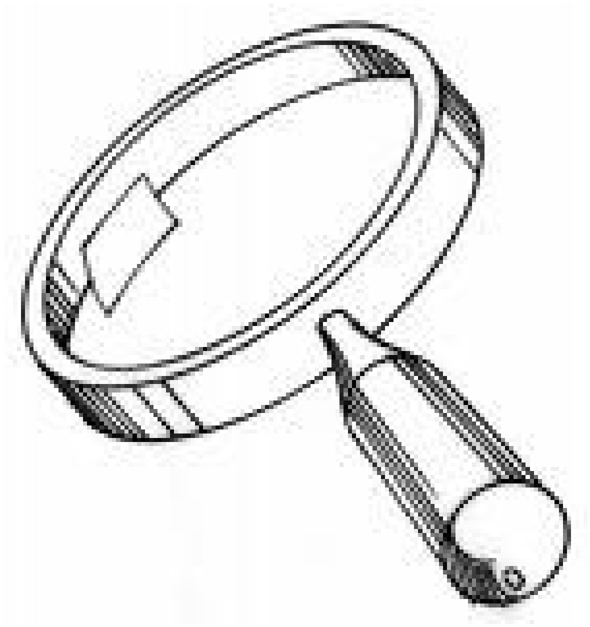
Cheung Ho Wan Richard

S06208150

Saturday, March 29, 2008

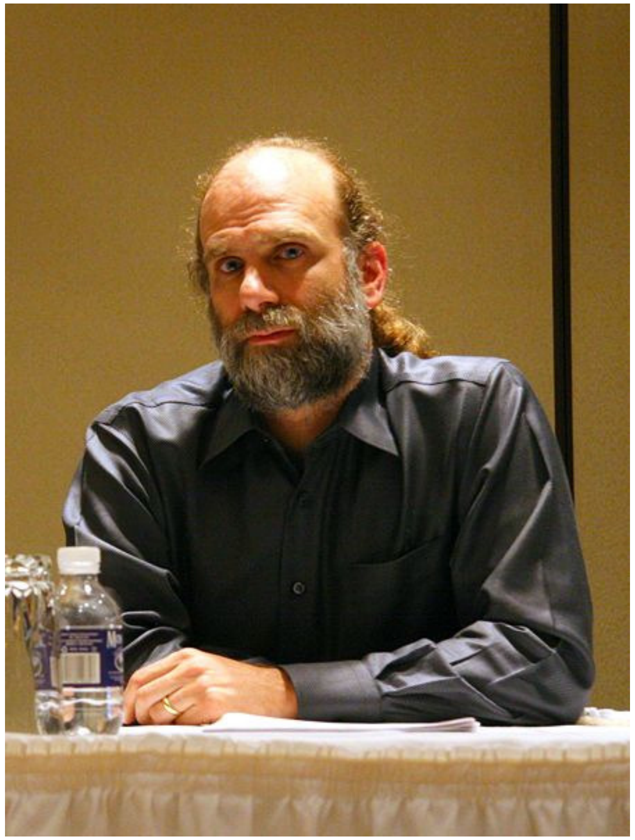
香港中文大學
The Chinese University of Hong Kong
學生證
Student Card
CULINK
張昊雲
CHEUNG, HO WAN
COMPUTER SCIENCE
06208150 Graduate School
Exp 7/2008
中大通

TOC



TOC

A Word about Information Security



- "Security is a process, not a product."
  - "Products provide some protection, but the only way to effectively do business in an insecure world is to put processes in place that recognize the inherent insecurity in the products. The trick is to reduce your risk of exposure regardless of the products or patches."
- Bruce Schneier , May 15, 2000
- Founder and CTO, Counterpane Internet Security, Inc.

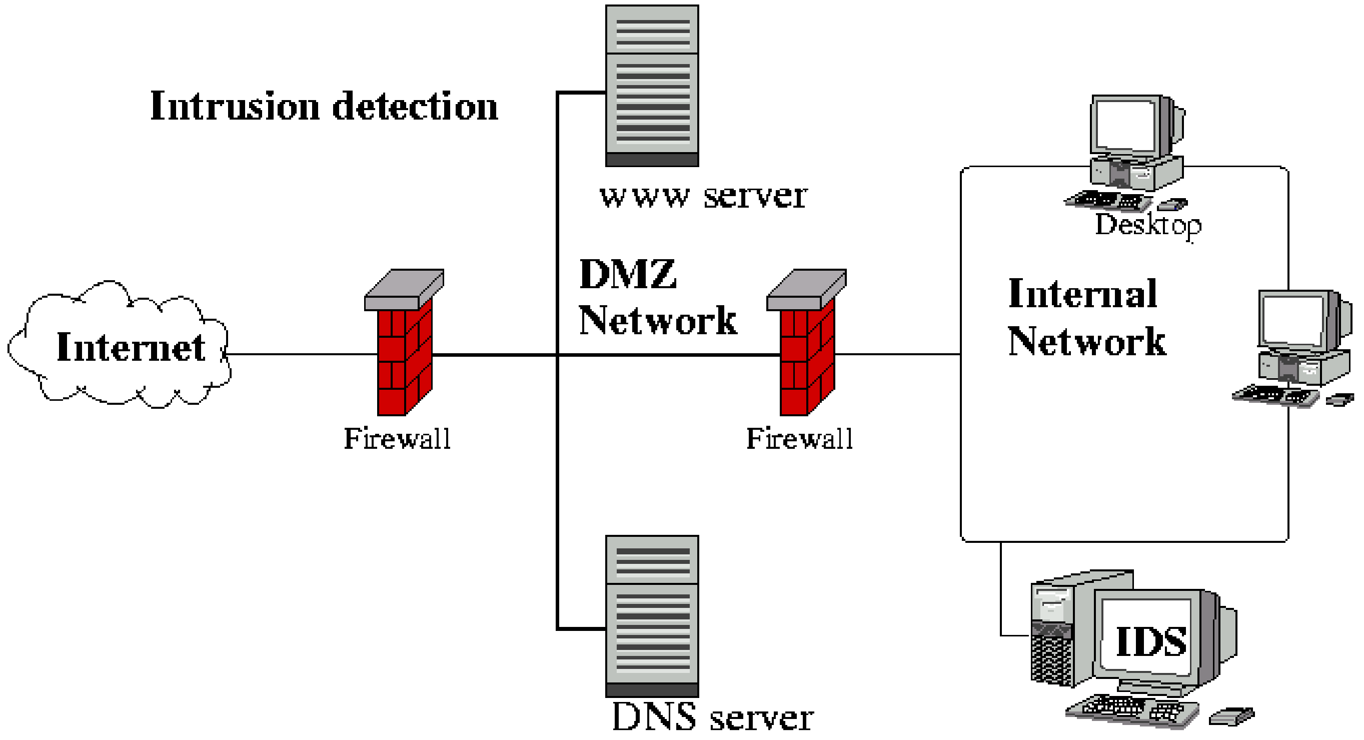
Network Security Components

- Firewall
- IDS/IPS/Honeypot/Honeynet
- Antivirus
- Security assessment tools
- Encryption and other security mechanisms
  - Open source community does well on security issues while some major commercial products do poor, e.g. Windows OS

What is IDS

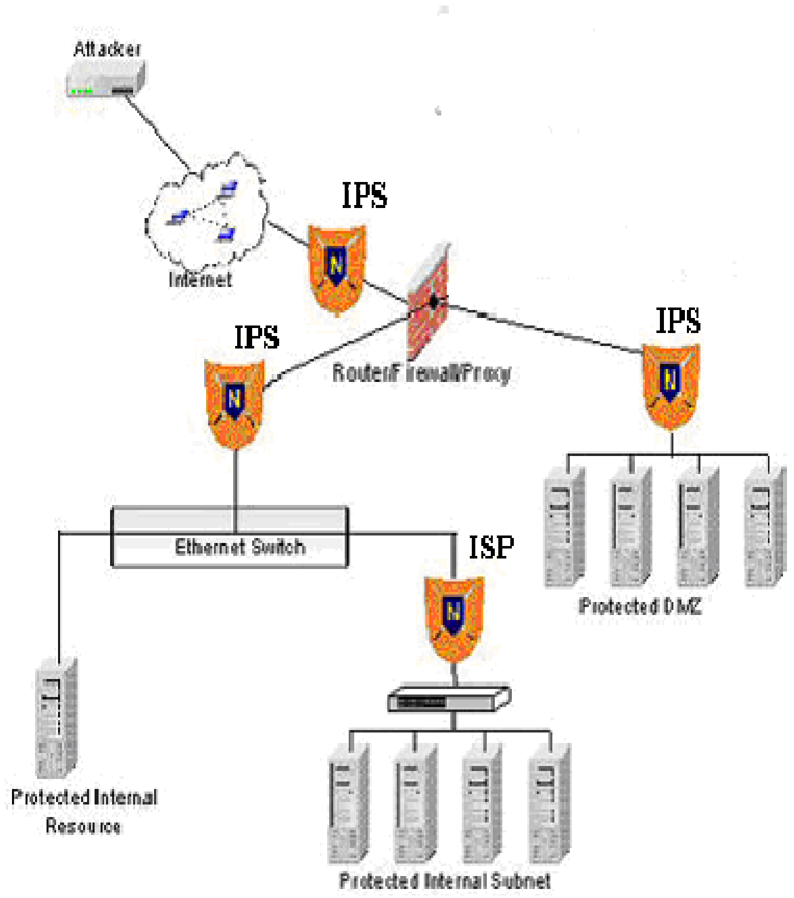
➢ Intrusion Detection System (IDS):

- An intrusion detection system generally detects unwanted manipulations of computer systems, mainly through the Internet.
- Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network.



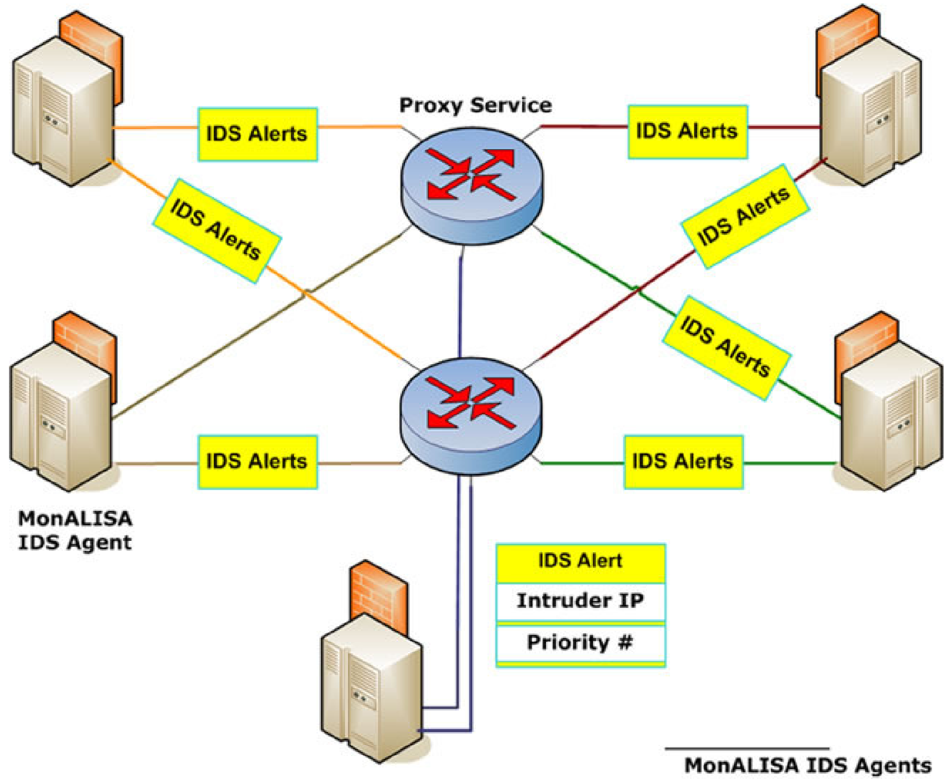
What is IPS

- Intrusion Prevention System (IPS):
  - Software/hardware that detects and logs inappropriate, incorrect, or anomalous activity. IDS are typically characterized based on the source of the data they monitor: host or network. A host-based IDS uses system log files and other electronic audit data to identify suspicious activity. A network-based IDS uses a sensor to monitor packets on the network to which it is attached.
  - An intrusion prevention system is a computer security device that monitors network and/or system activities for malicious or unwanted behavior and can react, in real-time, to block or prevent those activities.

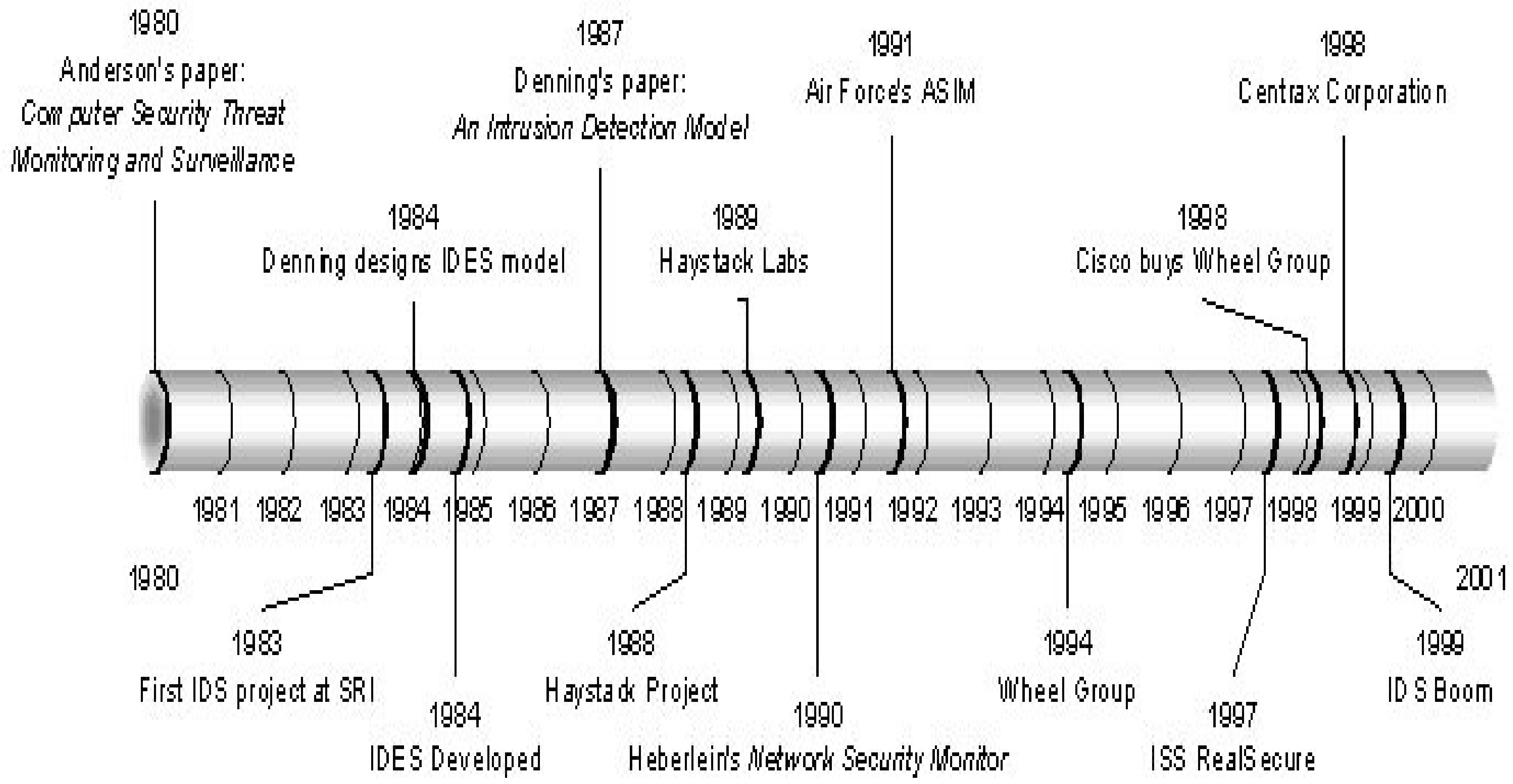


Why IDS

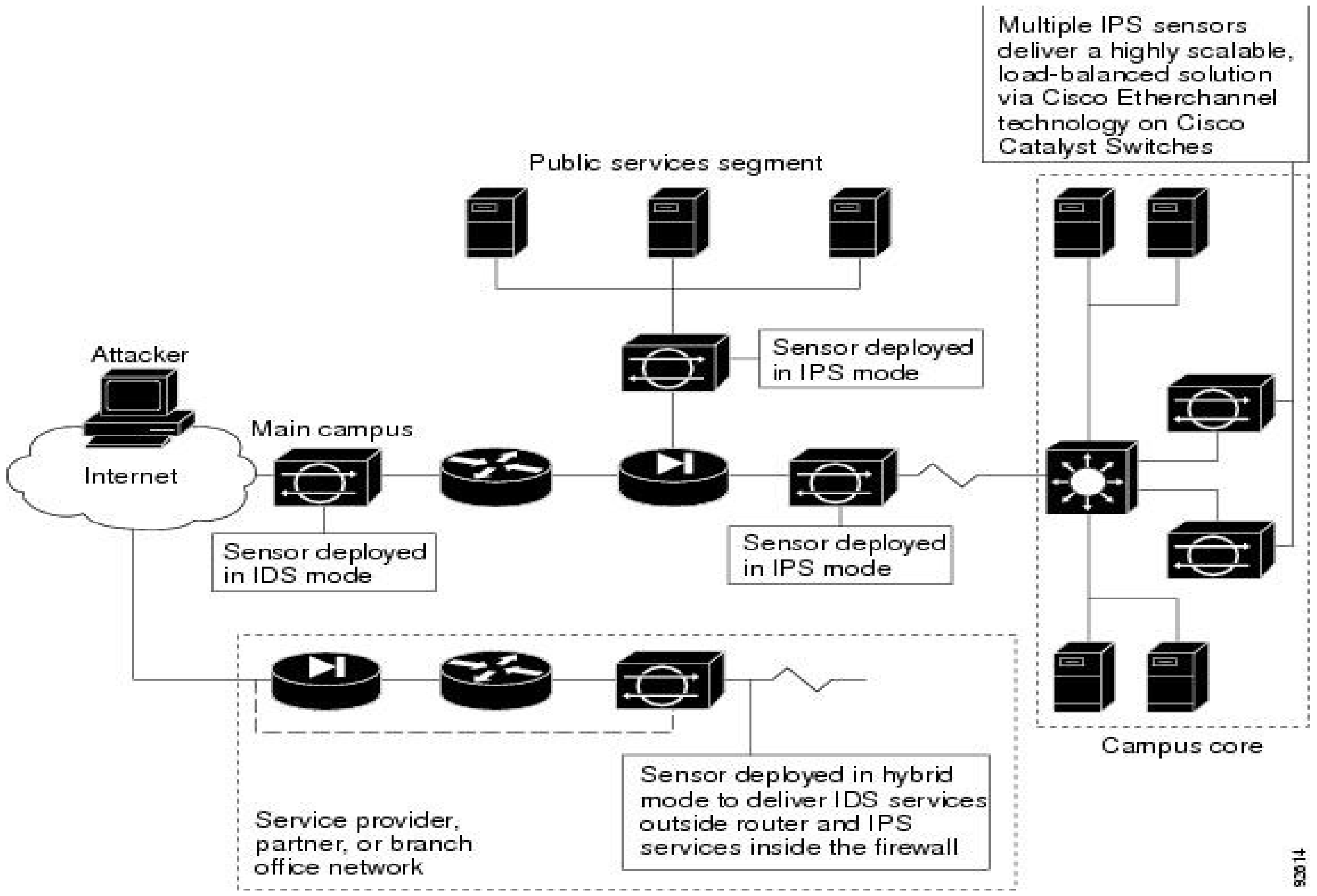
- it helps one to know what is going on one's security
  - recognize damage & affected systems
  - evaluating incidents
  - trace back intrusions
  - forensic analysis
  - prosecute sb. for a crime
- it helps one to defend one's security



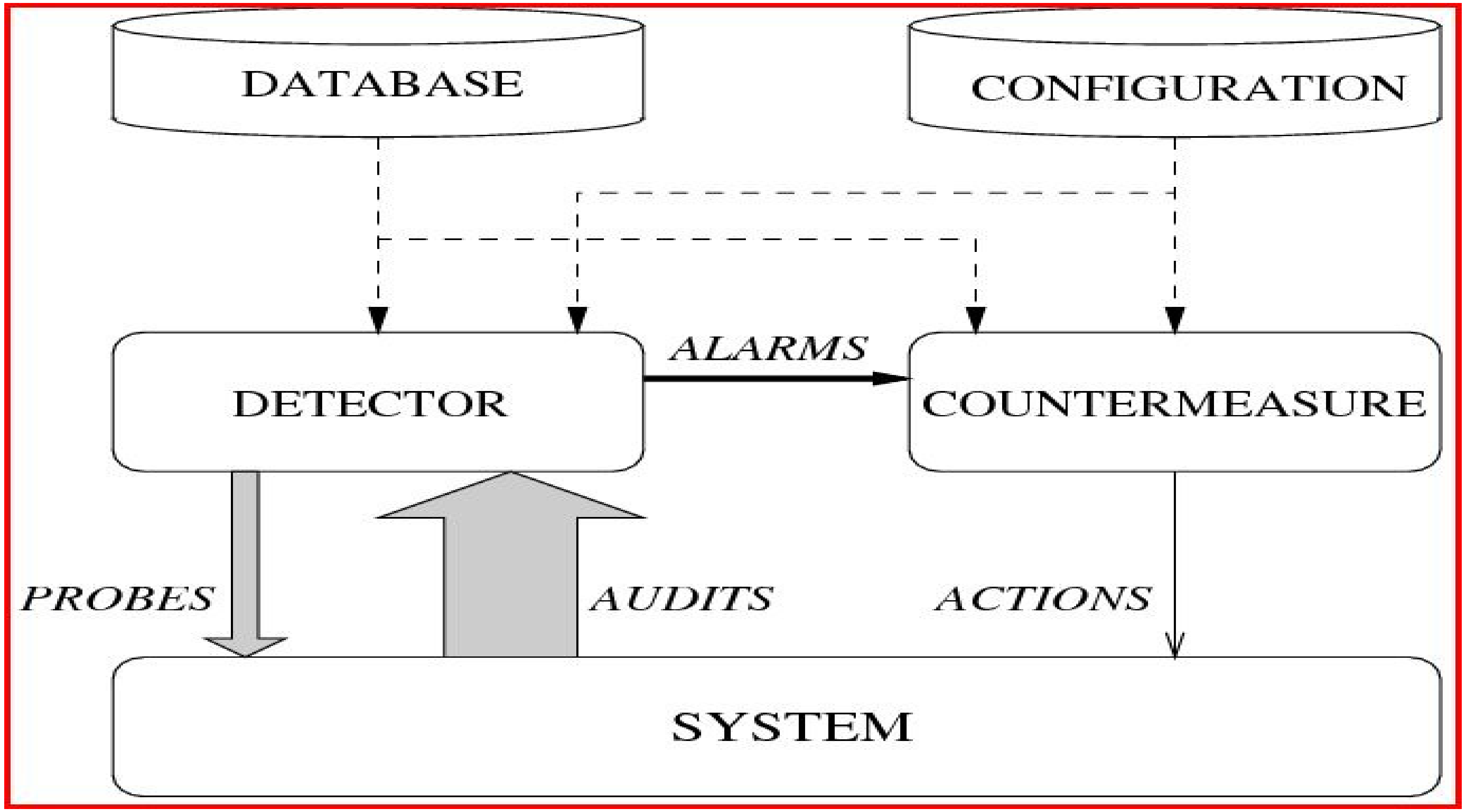
IDS's History



How IDS Works: Networking View

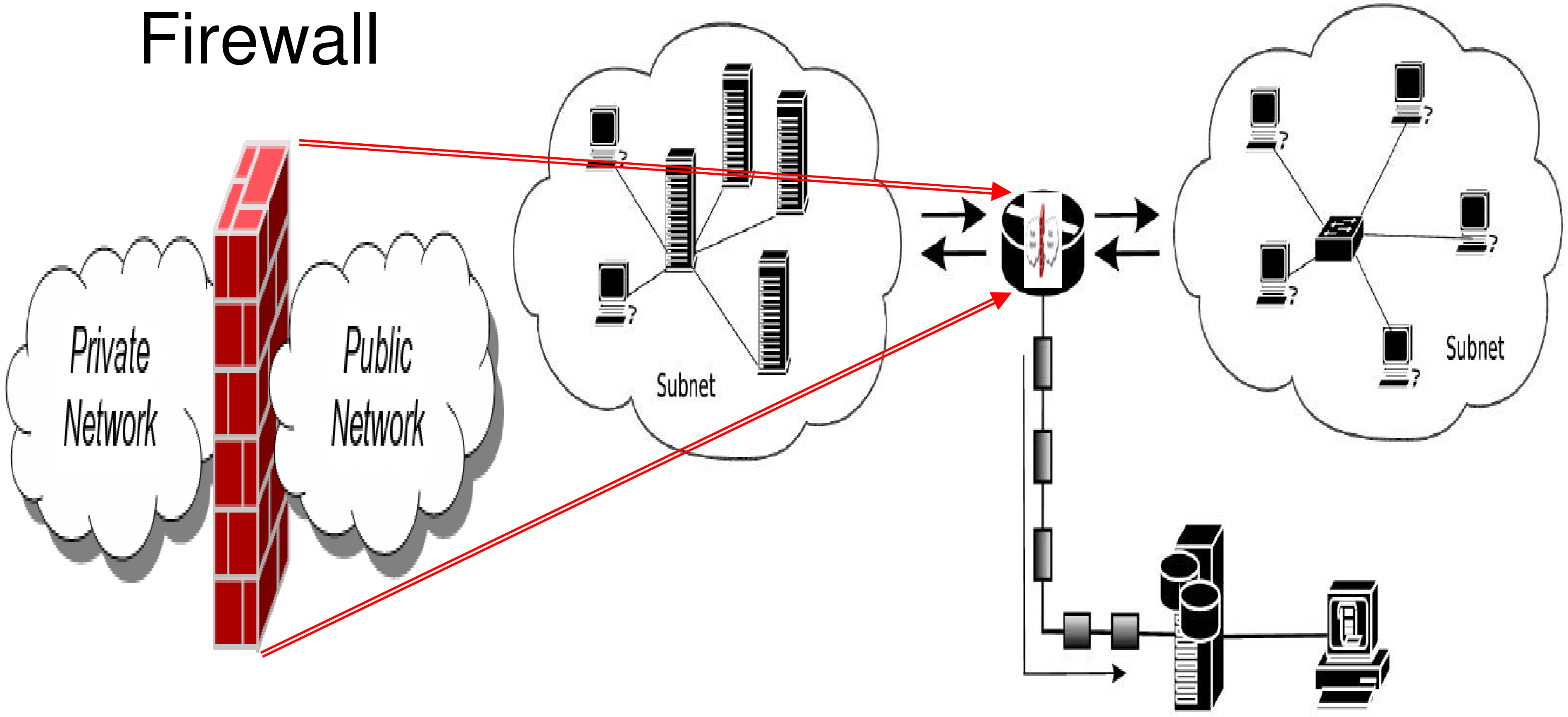

Multiple IPS sensors deliver a highly scalable, load-balanced solution via Cisco Etherchannel technology on Cisco Catalyst Switches
Public services segment
Sensor deployed in IPS mode
Attacker
Main campus
Internet
Sensor deployed in IDS mode
Sensor deployed in IPS mode
Campus core
Sensor deployed in hybrid mode to deliver IDS services outside router and IPS services inside the firewall
Service provider, partner, or branch office network
92814

How IDS Works: Data Flow View



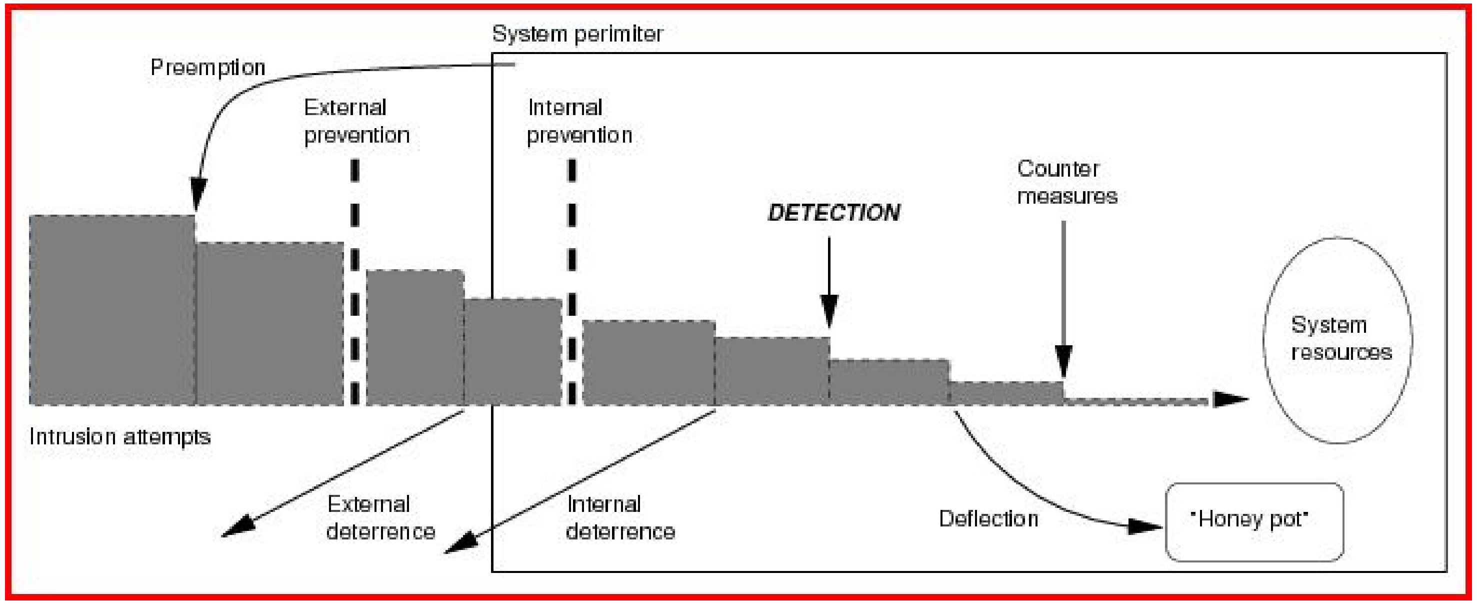
The Relationship of IPS & Firewall

- IPS is considered as the extension of traditional firewall, and now IDS≈Modern Firewall



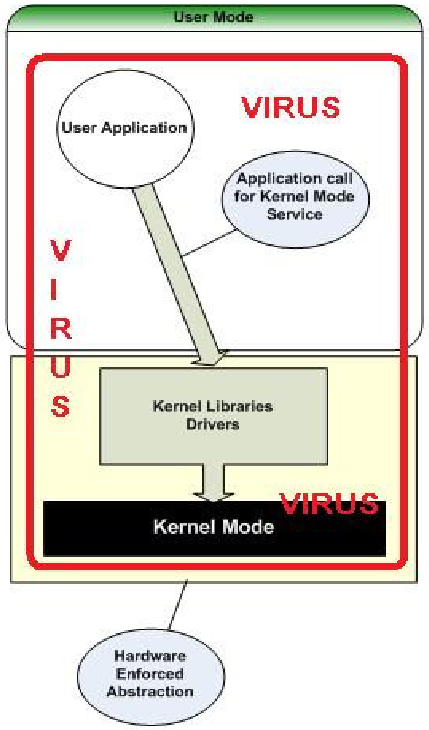
The Relationship of IDS & Honeypot/Honeynet

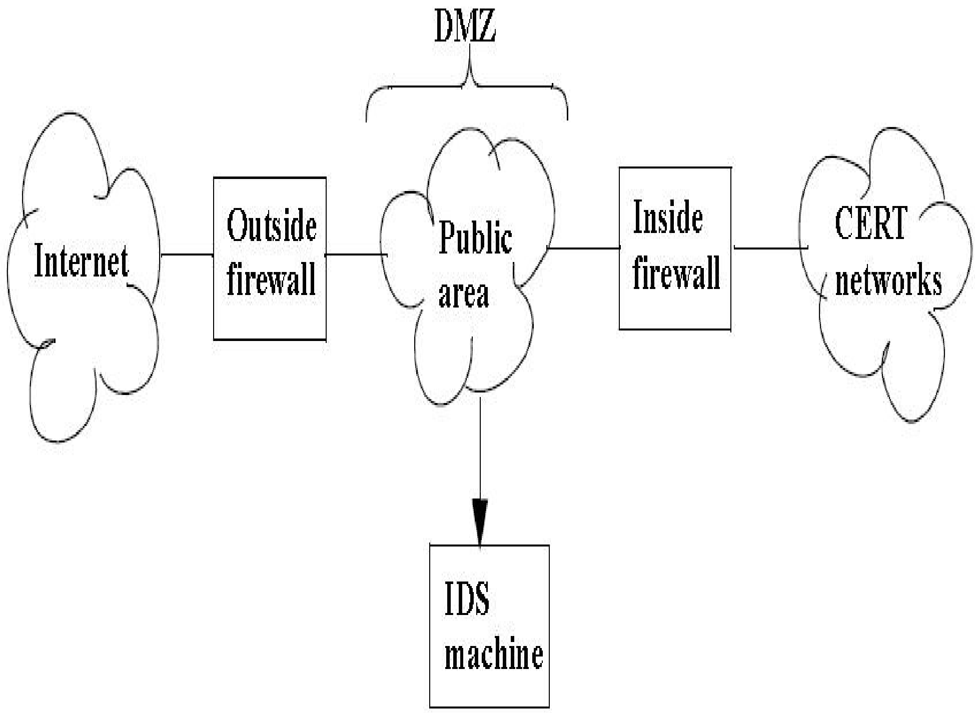
- Honeypot is derived from Snort and is considered as the young brother of Snort and now the Honeynet



The Relationship of IDS & Antivirus Programs

- IDS primarily work on TCP/IP stacks and antivirus programs primarily work inside of kernel & user spaces on a box’s OS





Taxonomy of IDS

- Anomaly Based IDS
- Policy Based IDS
- Host Based IDS
- Network Based IDS
- Distributed NIDS
- Hybrid IDS
- Reactive IDS
- Passive IDS

IDS and DOROTHY E. DENNING

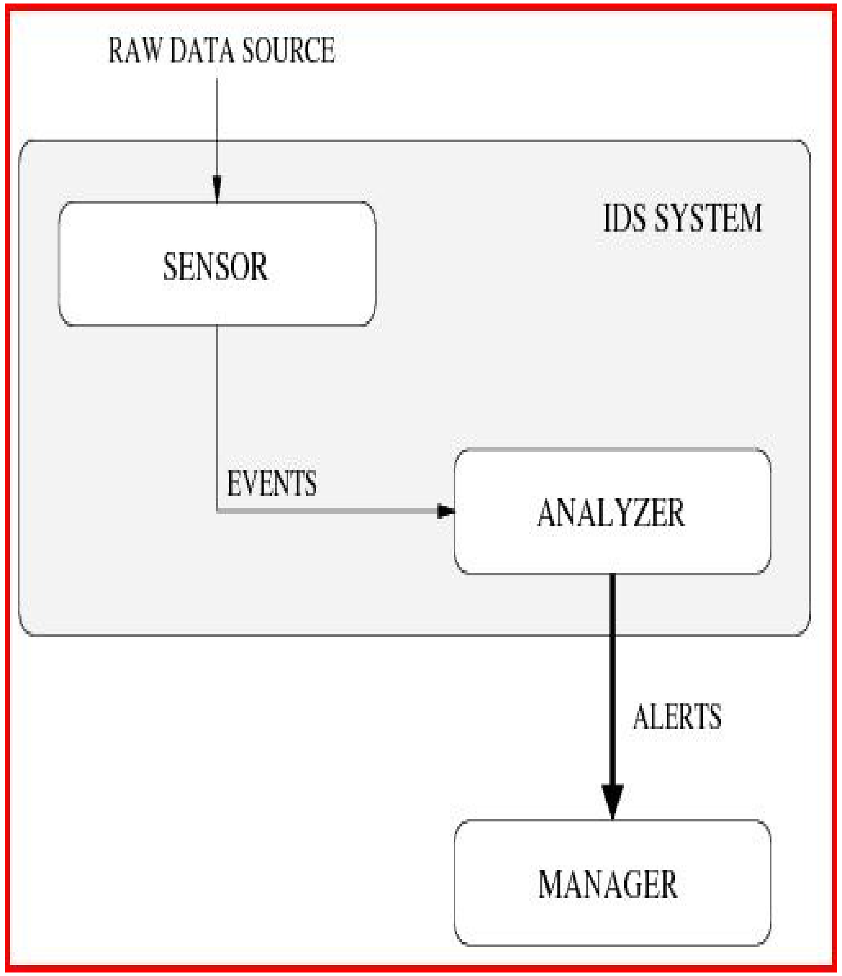
- 6 Main Components
  - Subjects
    - Initiators of activity on a target system- normally users.
  - Objects
    - Resources managed by the system-files, commands, devices, etc.
  - Audit records
    - Generated by the target system in response to actions performed or attempted by subjects on objects-user login, command execution, file access, etc.
  - Profiles
    - Structures that characterize the behavior of subjects with respect to objects in terms of statistical metrics and models of observed activity. Profiles are automatically generated and initialized from templates.
  - Anomaly records
    - Generated when abnormal behavior is detected.
  - Activity rules
    - Actions taken when some condition is satisfied, which update profiles, detect abnormal behavior, relate anomalies to suspected intrusions, and produce reports.

IDS and CIDF and IETF

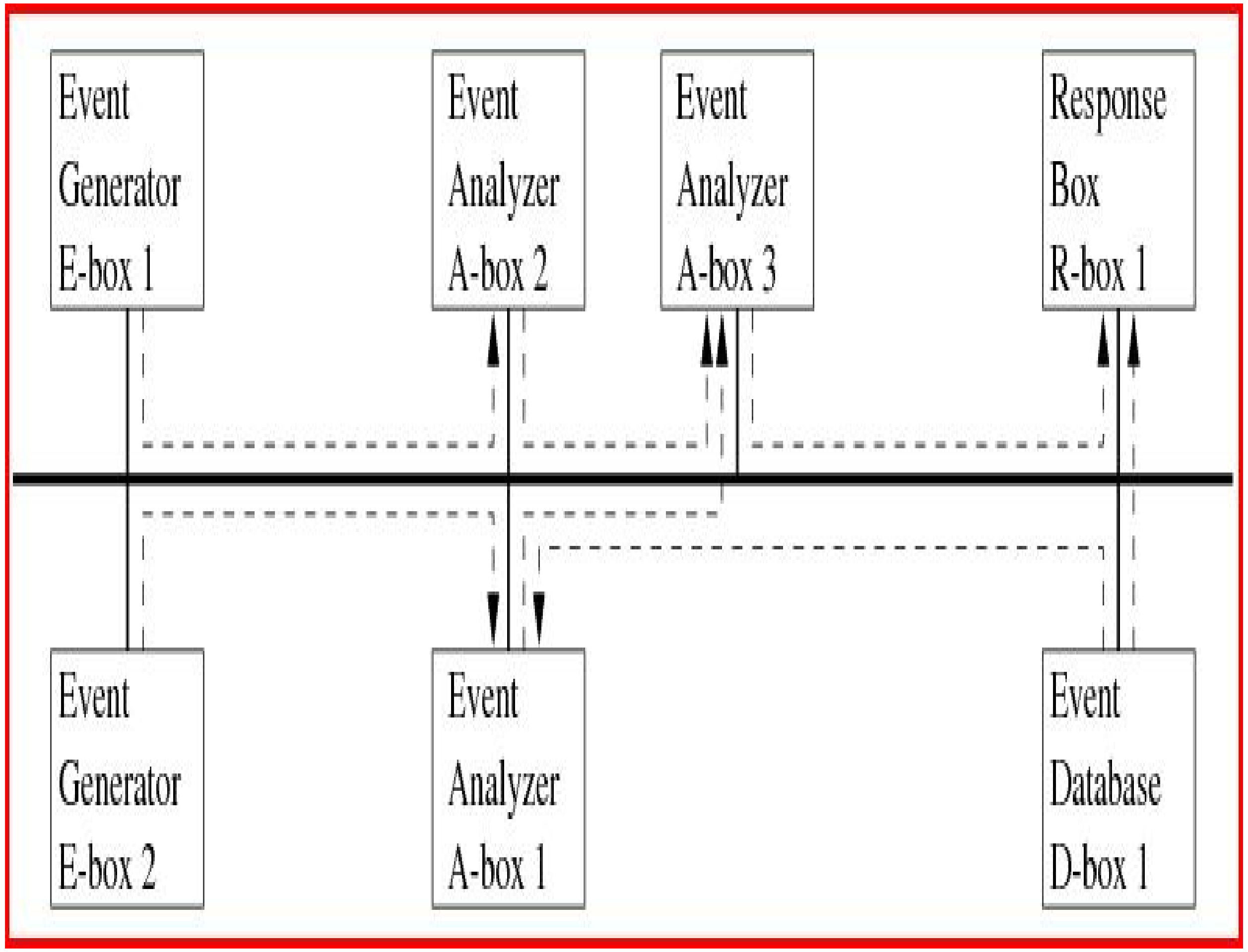
- CIDF
  - Common Intrusion Detection Framework
    - It’s an effort to develop protocols and application programming interfaces so that intrusion detection research projects can share information and resources and so that intrusion detection components can be reused in other systems.
    - Very first, Teresa Lunt, a former ITO of DARPA
    - Now, many companies and organizations with no relationship to DARPA

CIDF Description of IDS

- Some of the ideas involved in CIDF have encouraged the creation of an Internet Engineering Task Force (IETF) working group, named the Intrusion Detection Working Group (IDWG).



IDWG Description of IDS



- Event boxes (E-boxes)
  - generate audit events that are processed by IDS
- Analysis boxes (A-boxes, ≈detector component)
  - process events from the E-boxes to create alarms
- Database boxes (D-boxes)
  - store events for later retrieval
- Response Boxes (R-boxes, ≈countermeasure boxes)
  - apply countermeasures to the system according to the alarms generated

Accuracy of IDS: FPR, FNR

- FPR
  - False Positive Rate

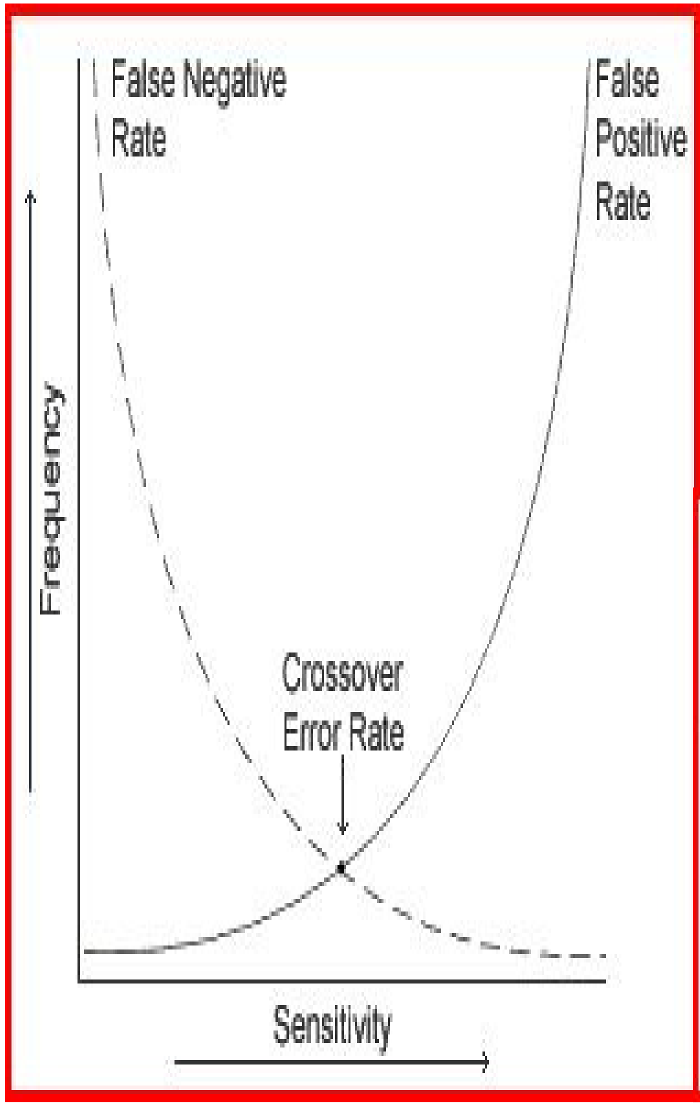
$$\text{false positive rate} = \frac{\text{number of false positives}}{\text{total number of negative instances}}$$ [5]

    - the frequency with which the IDS reports malicious activity in error
- FNR
  - False Negative Rate

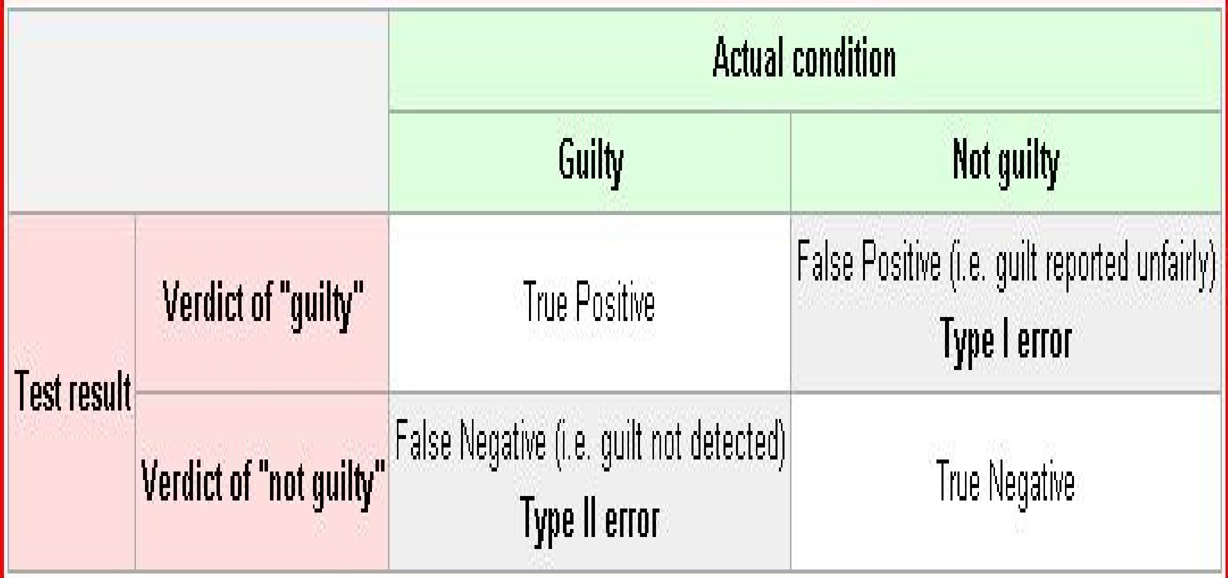
$$\text{false negative rate} = \frac{\text{number of false negatives}}{\text{total number of positive instances}}$$

    - the frequency with which the IDS fails to raise an alert when malicious activity actually occurs

Accuracy of IDS: CER



- CER
  - Crossover Error Rate
    - often used to provide a baseline measure for comparison of intrusion-detection systems



		Actual condition	
		Guilty	Not guilty
Test result	Verdict of "guilty"	True Positive	False Positive (i.e. guilt reported unfairly) Type I error
	Verdict of "not guilty"	False Negative (i.e. guilt not detected) Type II error	True Negative

Example of IDS Analysis (1)

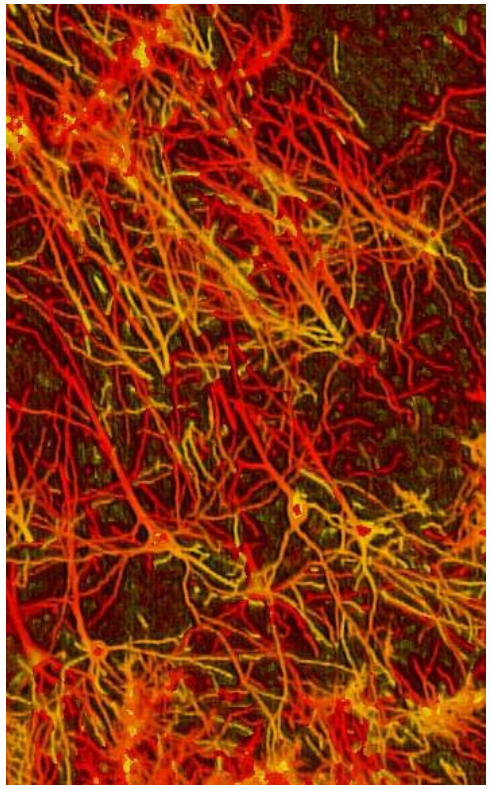
- Signature matching technique
  - snort
- Expert system technique
  - NIDES CMDS
- State Transition/CP-Nets technique
  - STAT USTAT NSTAT NetSTAT
- Quantitative technique
  - threshold detection, heuristic threshold detection, target-based integrity checks, quantitative technique and data reduction

Example of IDS Analysis (2)

- Statistical technique
- Rule-based technique
- Neural network technique
- Immunes system technique
- Genetics technique
- Agent technique

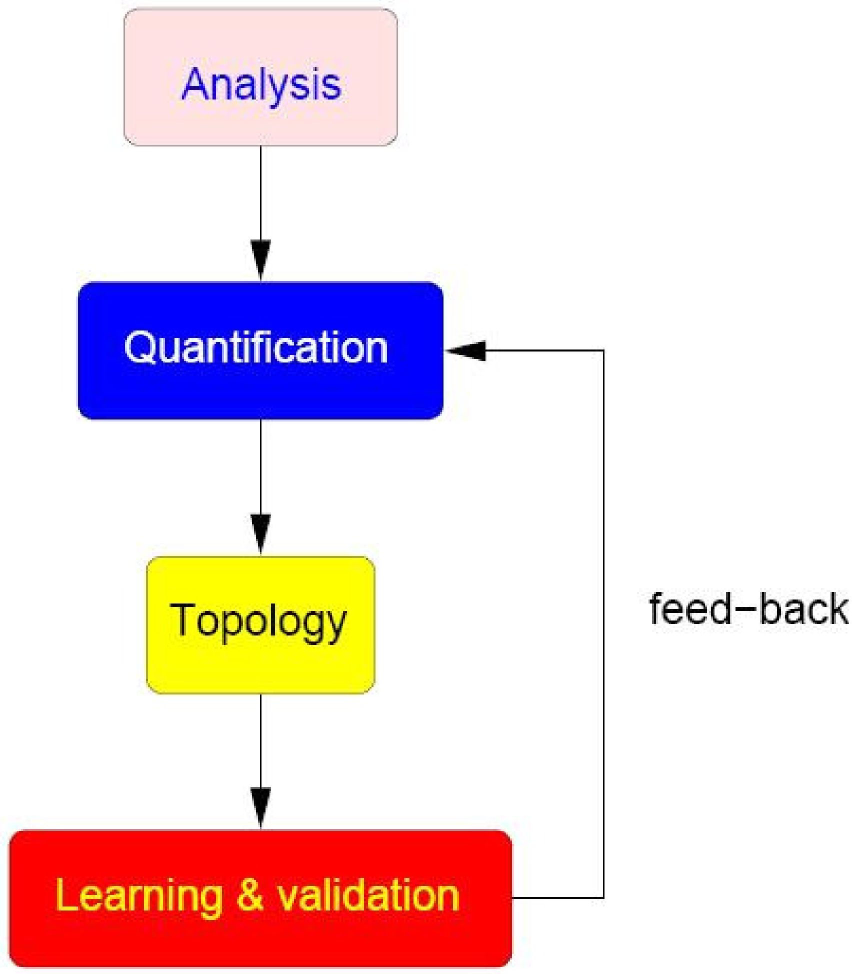
New Development of IDS: AI/ANN

- Let machines do human's works
  - lower cost
  - higher accuracy
  - faster speed
  - self-learning
  - Adaptive ability
  - automated recognition
- Neural Networks
- Fuzzy Logic



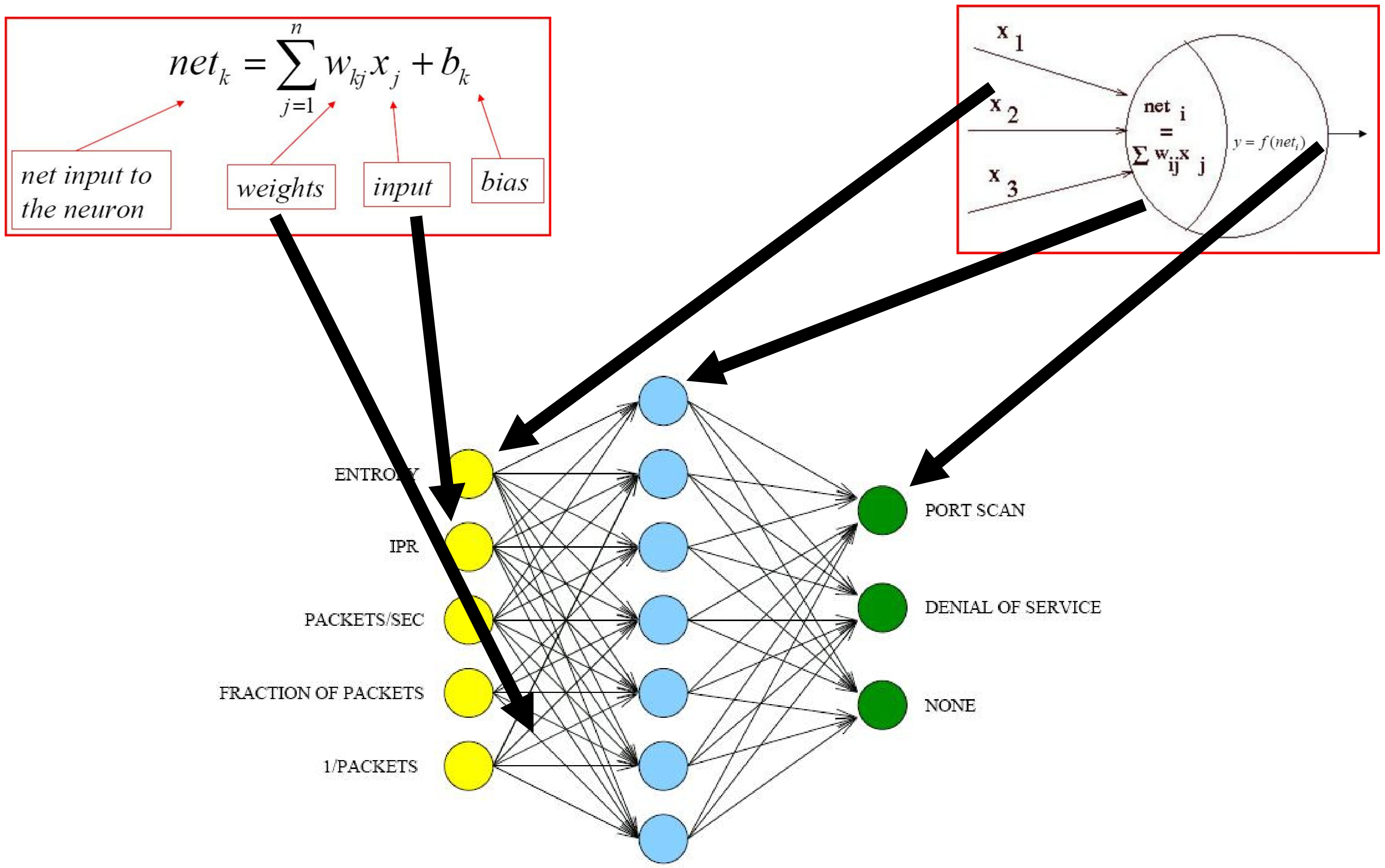
Main Neural IDS

- Self-Organizing Maps (SOM)
- Radial basis neural networks
- Artificial Neural Networks (ANN)



Neural IDS Example: ANN

$$net_k = \sum_{j=1}^{n} w_{kj} x_j + b_k$$



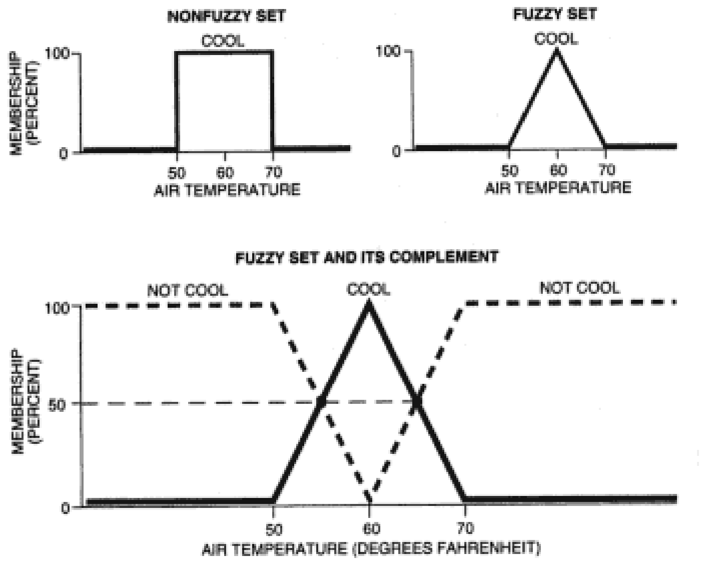
Fuzzy Logic IDS

Classical rules

IF user=root and port=22
THEN pass
IF ip!=10.0.0.1-10.0.0.200
    and port < 1024
THEN alert
……

Fuzzy rules

IF user is privileged and port is
    secure
THEN pass
IF ip isn't trusted and port is low
THEN alert
……



Defects with IDS: Evasion (1)

- Polymorphic buffer overflow attacks
  - attack does not have a single detectable signature

```
Start:
GOTO Decryption_Code
Encrypted:
   ...
   lots of encrypted code
   ...
Decryption_Code:
   A = Encrypted
Loop:
   B = *A
   B = B XOR CryptoKey
   *A = B
   A = A + 1
   GOTO Loop IF NOT A = Decryption_Code
   GOTO Encrypted
 CryptoKey:
   some_random_number
```

$\longrightarrow$

```
Start:
GOTO Decryption_Code
Encrypted:
   ...
   lots of encrypted code
   ...
Decryption_Code:
   C = C + 1
   A = Encrypted
Loop:
   B = *A
   C = 3214 * A
   B = B XOR CryptoKey
   *A = B
   C = 1
   C = A + B
   A = A + 1
   GOTO Loop IF NOT A = Decryption_Code
   C = C^2
   GOTO Encrypted
 CryptoKey:
   some_random_number
```

Defects with IDS: Evasion (2)

- Unicode directory traversal
  - .. /.. /.. / .->%co%af
  - The repeated ../ caused traversal to the root directory, finally caused response including /etc/passwd

```
<?php
$template = 'blue.php';
if ( isset( $_COOKIE['TEMPLATE'] ) )
   $template = $_COOKIE['TEMPLATE'];
include ( "/home/users/phpguru/templates/" .
$template );
?>
```

```
GET /vulnerable.php HTTP/1.0
Cookie: TEMPLATE=../../../../../../../../../etc/passwd
```

```
HTTP/1.0 200 OK
Content-Type: text/html
Server: Apache

root:fi3sED95ibqR6:0:1:System Operator:/:/bin/ksh
daemon:*:1:1::/tmp:
phpguru:f8fk3j1OIf31.:182:100:Developer:/home/users/phpguru/:/bi
n/csh
```

Defects with IDS: Evasion (3)

- Protocol anomalies
  - behaviors deviated from normal behavior will be classified as anomalous
    - HTTP traffic on a non-standard port, say port 53 (protocol anomaly)
    - Backdoor service on well-known standard port, e.g., peer-to-peer file sharing using Gnutella on port 80 (protocol anomaly and statistical anomaly)
    - A segment of binary code in a user password (application anomaly)
    - Too much UDP compared to TCP traffic (statistical anomaly)
    - A greater number of bytes coming from an HTTP browser than are going to it (application and statistical anomaly)

Defects with IDS: Evasion (4)

- Fragmentation
  - split the attack payload into multiple small packets
  - session splicing
    - Put session date into multiple packets to evade IDS
  - Fragmentation overlap
  - Fragmentation overwrite
  - Fragmentation time-outs

pkt no	content
1	G
2	E
3	T
4	20
5	/
6	H

Defects with IDS: Evasion (5-7)

- Denial of Service
  - disable IDS by overwhelming of packets
    - stick
    - snot
- Path obfuscation
  - e.g. /winnt/. /. /. / = /winnt
- Hex encoding
  - not all IDS know %20 = hex 20
  - GET %65%74%63/%70a%73%73%77d

    $\downarrow$

  - GET %65%74%63/%70%61%73%73%77%64

Defects with IDS: Others (8-11)

- IDS can't compensate for poor security design
- IDS can't against new and sophisticated attacks effectively
- No integrated tool to do IDS once for all
- IDS can't be operated in switched environment effectively

Defects with IDS: Others (12-15)

- IDS' accuracy is low, often produce false alarms
- IDS' speed is low, may slow down the overall network speed or host speed, otherwise miss traffic
- IDS' outputs waste a lot of operational cost, e.g. human capital, time
- The ability of IDS' proactive countermeasures is limited, e.g. the automation with firewall

Defects with IDS: Others (16-19)

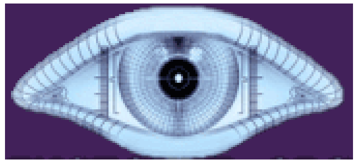
- Commercial IDS is expensive while open source IDS is hard to use
- No factual unified industrial standard such as APIs, languages, storage formats
- Few qualified technical staff can manage IDS
  - e.g. installation, configuration, integration, and analysis are complex
- The development speed of intrusion tools/mechanisms is faster than IDS' development speed

Defects with IDS: Others (20-23)

- Network complexity and size makes the difficulty of implementing IDS to be exponential
- Non-IT staff don’t like to work under IDS’ monitoring so IDS’ implementation often refused by top management
- IDS can’t find out the hackers’ identification and what they want
- IDS may provide information about intrusion but almost no help to recover damages

ID Industry: The Top 5

- survey launched by Gordon Lyon in 2002, 2003, 2006
- users are from the nmap-hackers mailing list
- 3,243 responded in the 2006 survey



① Snort : Everyone's favorite open source IDS

② OSSEC HIDS : An Open Source Host-based Intrusion Detection System

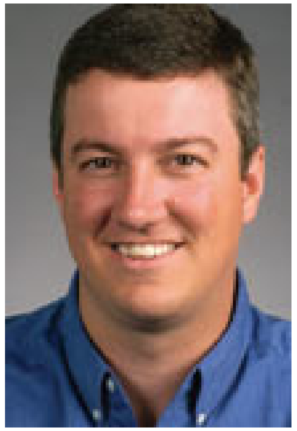
③ Fragroute/Fragrouter : A network intrusion detection evasion toolkit

④ BASE : The Basic Analysis and Security Engine

⑤ Sguil : The Analyst Console for Network Security Monitoring

What is Snort

- Snort is a free and open source IDP that perform packet logging and real-time traffic analysis on IP networks.
  - Martin Roesch
  - Sourcefire
  - snort-2.8.0.2.tar.gz
  - Tue Feb 19 2008
  - Linux BSD Windows
  - GNU GPL
  - www.snort.org



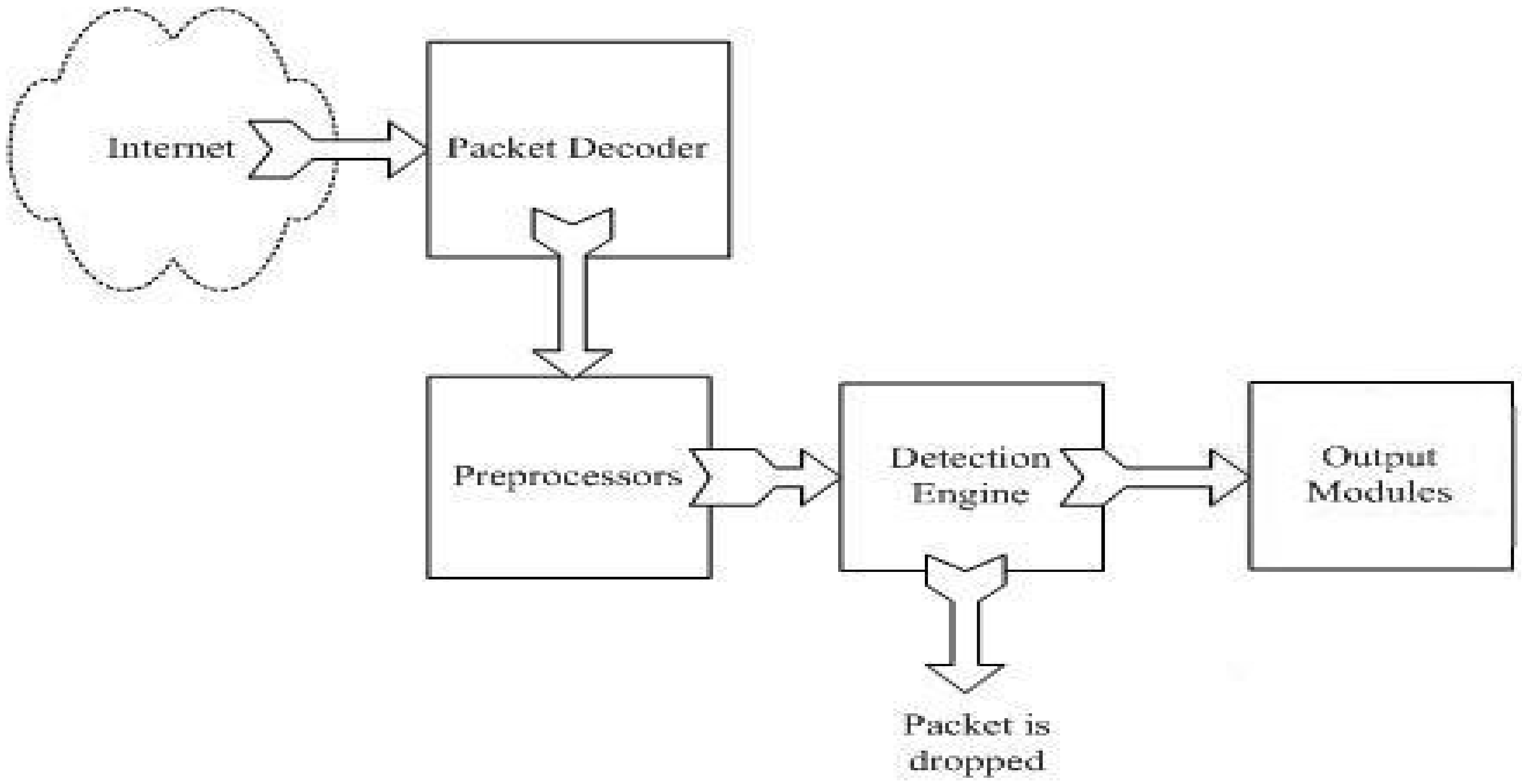
Snort’s 4 Modes

- Sniffer mode
  - simply reads the packets off of the network and displays them for you in a continuous stream on the console (screen).
- Packet Logger mode
  - logs the packets to disk
- NIDS mode
  - the most complex and configurable configuration, which allows Snort to analyze network traffic for matches against a user-defined rule set and performs several actions based upon what it sees
- Inline mode
  - obtains packets from iptables instead of from libpcap and then causes iptables to drop or pass packets based on Snort rules that use inline-specific rule types

Snort’s Components

- Packet sniffer
- Packet decoder (1/3 primary subsystems)
- Preprocessors
- Detection engine (2/3 primary subsystems)
- Output modules (Logging and alerting system, 3/3 primary subsystems)

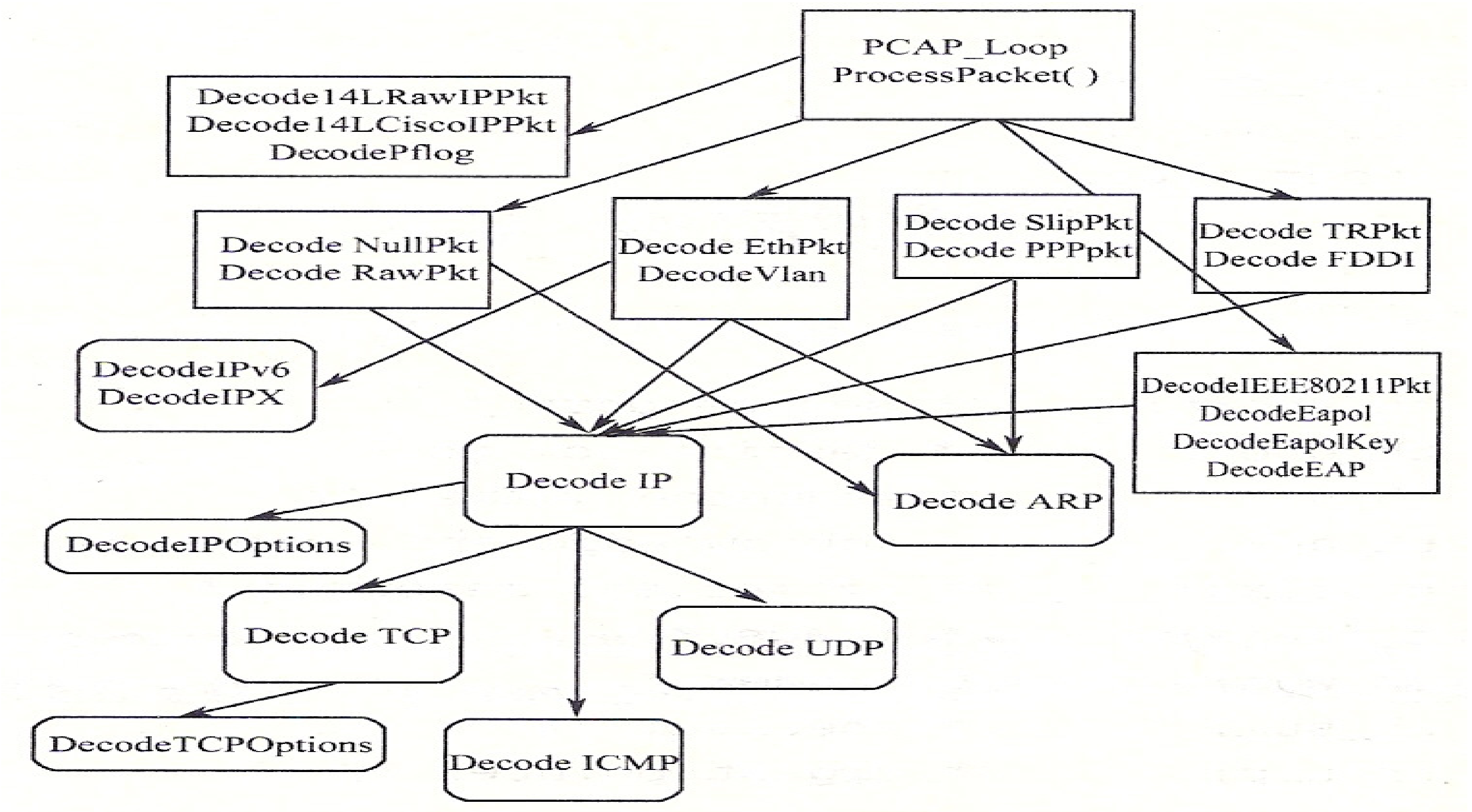
Snort's Components' Relationship



Snort’s Packet Sniffer & Decoder

- Sniffer
  - takes packets from NICs
    - Ethernet, SLIP, PPP etc.
    - raw packet capture
      - libpcap
      - winpcap
- Decoder
  - Prepares packets to preprocessor
    - to form internal Snort data structure
    - as a uniform basis for later analysis

Snort’s Packet Decoder Calling Diagram



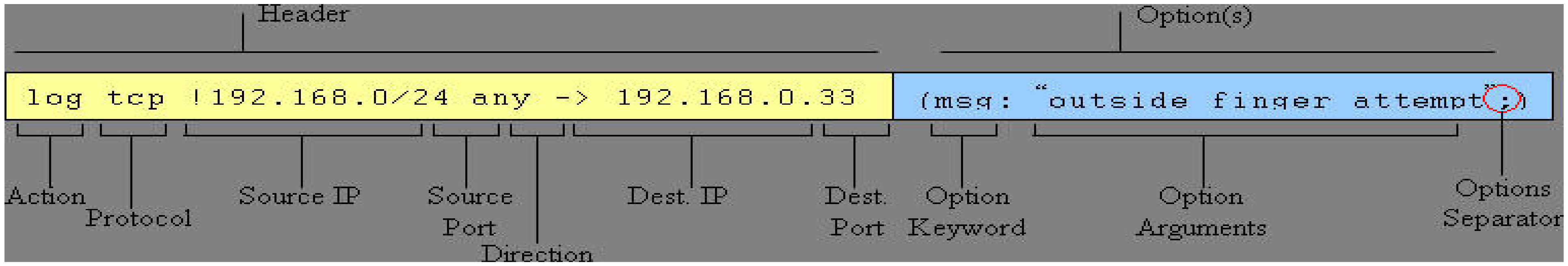
Example of Snort Data Dtructure

DynamicPluginMeta structure

```
#define TYPE_ENGINE 0x01
#define TYPE_DETECTION 0x02
#define TYPE_PREPROCESSOR 0x04
typedef struct _DynamicPluginMeta
{
        int type;
        int major;
        int minor;
        int build;
        char uniqueName[MAX_NAME_LEN];
        char *libraryPath;
} DynamicPluginMeta;
```

Snort's Preprocessors

- Frag3
- Streams
- Flow
- Stream5
- sfPortscan
- RPC Decode
- Performance Monitor
- SMTP Preprocessor
- FTP/Telnet Preprocessor
- SSH
- DEC/RPC
- DNS



```
log tcp !192.168.0/24 any -> 192.168.0.33 (msg: "outside finger attempt";)
```

Example of Main Preprocessor (Frag3)

- The frag3 preprocessor is a target-based IP defragmentation module for Snort.
- Target-based analysis is a relatively new concept in network-based intrusion detection.
- In an environment where the attacker can determine what style of IP defragmentation is being used on a particular target, the attacker can try to fragment packets such that the target will put them back together in a specific manner while any passive systems trying to model the host traffic have to guess which way the target OS is going to handle the overlaps and retransmits.

  - preprocessor frag3_global: prealloc_nodes 8192
  - preprocessor frag3_engine: policy linux, bind_to 192.168.1.0/24
  - preprocessor frag3_engine: policy first, bind_to [10.1.47.0/24,172.16.8.0/24]
  - preprocessor frag3_engine: policy last, detect_anomalies

Example of Main Preprocessor (Stream5)

- The Stream5 preprocessor is a target-based TCP reassembly module for Snort.
  - This configuration maps two network segments to different OS policies, one for Windows and one for Linux, with all other traffic going to the default policy of Solaris.
    - preprocessor stream5_global: track_tcp yes
    - preprocessor stream5_tcp: bind_to 192.168.1.0/24, policy windows
    - preprocessor stream5_tcp: bind_to 10.1.1.0/24, policy linux
    - preprocessor stream5_tcp: policy solaris

Example of Main Preprocessor (sfPortscan) (1)

- The sfPortscan module, developed by Sourcefire, is designed to detect the first phase in a network attack: Reconnaissance.
- In the Reconnaissance phase, an attacker determines what types of network protocols or services a host supports. This is the traditional place where a portscan takes place. This phase assumes the attacking host has no prior knowledge of what protocols or services are supported by the target; otherwise, this phase would not be necessary.
- As the attacker has no beforehand knowledge of its intended target, most queries sent by the attacker will be negative (meaning that the service ports are closed). In the nature of legitimate network communications, negative responses from hosts are rare, and rarer still are multiple negative responses within a given amount of time. The primary objective in detecting port scans is to detect and track these negative responses.

Example of Main Preprocessor (sfPortscan) (2)

```
preprocessor sfportscan: proto <protocols> \
scan_type
   <portscan|portsweep|decoy_portscan|distributed_portsc
   an|all>\
sense_level <low|medium|high> watch_ip <IP or IP/CIDR>
   ignore_scanners <IP list>\
ignore_scanned <IP list> logfile <path and filename>

preprocessor flow: stats_interval 0 hash 2
preprocessor sfportscan: proto { all } \
scan_type { all } \
sense_level { low }
```

Example of Main Preprocessor (SSH)

- The SSH preprocessor detects the following exploits: Gobbles, CRC 32, Secure CRT, and the Protocol Mismatch exploit.
  - Looks for attacks on SSH server port 22. Alerts at 19600 bytes within 20 encrypted packets for the Gobbles/CRC32 exploits.
    - preprocessor ssh: server_ports { 22 } max_client_bytes 19600 max_encrypted_packets 20

Example of Main Preprocessor (DNS)

- The DNS preprocessor decodes DNS responses and can detect the following exploits: DNS Client RData Overflow, Obsolete Record Types, and Experimental Record Types.
  - Looks for traffic on DNS server port 53. Check for the DNS Client RData overflow vulnerability. Do not alert on obsolete or experimental RData record types.
    - preprocessor dns: server_ports { 53 } enable_rdata_overflow

Snort’s Detection Engine

- Rules
  - Rules Headers
  - Rule Options
    - alert tcp any any -> 192.168.1.0/24 111\ (content:"|00 01 86 a5|"; msg:"mountd access";)
- Detection plug-ins
- Rule Chain
  - Activation
  - Dynamic
  - Alert
  - Pass
  - log

RTN: TCP, UDP, ICMP, IP
OTN: Content, Offset, Distance, Within etc.
These components just like the rule chains in firewalls, e.g. netfilter

Detection Engine Example: IcmpTypeCheck

IcmpTypeCheck PLUG-IN

```
/****************************************************************************
 *
 * Function: IcmpTypeCheck(char *, OptTreeNode *)
 *
 * Purpose: Test the packet's ICMP type field value against the option's
 *          ICMP type
 *
 * Arguments: data => argument data
 *            otn => pointer to the current rule's OTN
 *
 * Returns: void function
 *
 ****************************************************************************/
int IcmpTypeCheck(Packet *p, struct _OptTreeNode *otn, OptFpList
        *fp_list)
{
   IcmpTypeCheckData *ds_ptr;
   int success = 0;

   ds_ptr = otn->ds_list[PLUGIN_ICMP_TYPE];

   /* return 0  if we don't have an icmp header */
   if(!p->icmph)
      return 0;

   switch(ds_ptr->operator)
 {
      case ICMP_TYPE_TEST_EQ:
         if (p->icmph->type == ds_ptr->icmp_type)
            success = 1;
         break;
case ICMP_TYPE_TEST_GT:
         if (p->icmph->type > ds_ptr->icmp_type)
            success = 1;
         break;
      case ICMP_TYPE_TEST_LT:
         if (p->icmph->type < ds_ptr->icmp_type)
            success = 1;
         break;
      case ICMP_TYPE_TEST_RG:
         if (p->icmph->type > ds_ptr->icmp_type &&
               p->icmph->type < ds_ptr->icmp_type2)
            success = 1;
         break;
   }

   if (success)
   {
      DEBUG_WRAP(DebugMessage(DEBUG_PLUGIN, "Got icmp type
        match!\n"););
      return fp_list->next->OptTestFunc(p, otn, fp_list->next);
   }

   /* return 0 on failed test */
   DEBUG_WRAP(DebugMessage(DEBUG_PLUGIN, "Failed icmp code
        match!\n"););
   return 0;
}
```

Snort’s Output Modules

- Alert_syslog
- Alert_fast
- Alert_full
- Alert_unixsock
- Log_tcpdump
- Database
- Csv
- Unified
- Unified2
- Log null
- Alert_aruba_action

Example of Database Output Module (DB)

- This module from Jed Pickel sends Snort data to a variety of SQL databases. These are mssql, mysql, postgresql, oracle, and odbc.

  - output database: log, mysql, dbname=snort user=snort host=localhost password=xyz

Example of Database Output Module (unified2)

- The unified2 output plugin is designed to be the fastest possible method of logging Snort events. The unified output plugin logs events in binary format, allowing another programs to handle complex logging mechanisms that would otherwise diminish the performance of Snort.
  - output alert_unified2: snort.alert, limit 128, nostamp
  - output log_unified2: snort.log, limit 128, nostamp
  - output unified2: merged.log, limit 128, nostamp

A Simple Test: the Settings

- Debian
  - 4.0r3 Arch:i386 released on 17-Feb-2008
- Snort
  - snort-rules-default 2.3.3-11
  - debconf 0.2.80 Syslogd
  - libc6 2.3.6-6 libpcap0.8 0.9.3-1
  - libpcre3 4.5 snort-common 2.3.3-11
  - Logrotate coreutils

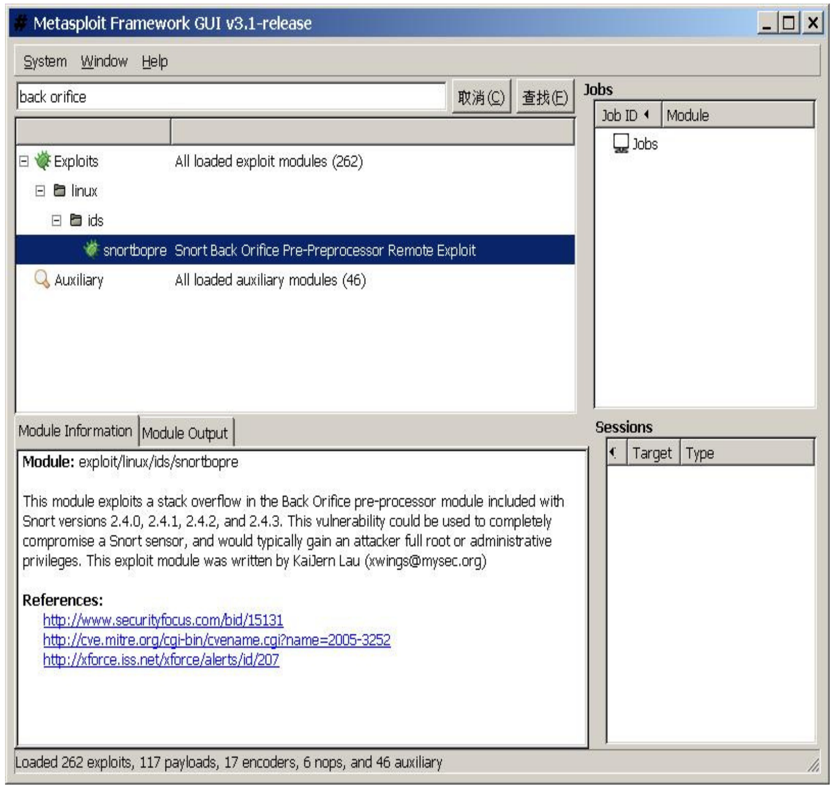
A Simple Test: the Attack

- The Metasploit Project
  - Metasploit provides useful information to people who perform penetration testing, IDS signature development, and exploit research.
- A short use example (≈4mins)
  - type: Flash Video
  - by: Chris Gates of LearnSecurityOnline.com
  - uses the VNC Injection payload to break a locked Windows desktop and monitor the user.
  - http://www.learnsecurityonline.com/vid/MSF3-VNC/MSF3-VNC.html

metasploit

A Simple Test: the Payload

- Snort Back Orifice Pre-Preprocessor Remote Exploit
- Listen for a connection and spawn a command shell
- Details of configuration of the attack
  - [linux/ids/snortbopre]
  - EnableContextEncoding=false
  - PAYLOAD=generic/shell_bind_tcp
  - ContextInformationFile=
  - LPORT=4444
  - EncoderDontFallThrough=false
  - RHOST=10.10.10.10
  - RPORT=9080
  - TARGET=0



```
Exploits        All loaded exploit modules (262)
  linux
    ids
      snortbopre  Snort Back Orifice Pre-Preprocessor Remote Exploit
Auxiliary       All loaded auxiliary modules (46)
```

Module: exploit/linux/ids/snortbopre

This module exploits a stack overflow in the Back Orifice pre-processor module included with Snort versions 2.4.0, 2.4.1, 2.4.2, and 2.4.3. This vulnerability could be used to completely compromise a Snort sensor, and would typically gain an attacker full root or administrative privileges. This exploit module was written by KaiJern Lau (xwings@mysec.org)

References:
http://www.securityfocus.com/bid/15131
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2005-3252
http://xforce.iss.net/xforce/alerts/id/207

Loaded 262 exploits, 117 payloads, 17 encoders, 6 nops, and 46 auxiliary

A Simple Test: the Result

```
23:02:50
    Initialized the Metasploit Framework GUI.
23:02:52
    Saved configuration to:
       C:\Users\cuistar\AppData\Local/.msf3/config
23:03:18
    snortbopre [*] Launching exploit linux/ids/snortbopre...
23:03:18
     snortbopre [*] Started bind handler
```

A Simple Test: the Defense (1)

Before launch msf attack:

```
s06208150:/var/log/snort# ls -lah
total 12K
drwxr-s---  2 snort adm  4.0K 2008-03-26 23:25 .
drwxr-xr-x 11 root  root 4.0K 2008-03-26 22:10 ..
-rw-r-----  1 snort adm     0 2008-03-05 23:09 alert
-rw-r-----  1 snort adm   344 2008-02-22 14:40 alert.1.gz
```

And then I issue:

```
s06208150:/var/log/snort# snort -A fast -c /etc/snort/snort.conf
```

A simple Test: the Defense (2)

Snort outputs:

```
Running in IDS mode

Initializing Network Interface eth4

        --== Initializing Snort ==--
Initializing Output Plugins!
Decoding Ethernet on interface eth4
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file /etc/snort/snort.conf

+++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
,-----------[Flow Config]----------------------
| Stats Interval:  0
| Hash Method:     2
| Memcap:          10485760
| Rows  :          4099
| Overhead Bytes:  16400(%0.16)
`----------------------------------------------
……
……
```

A Simple Test: the Defense (3)

```
===============================================================================
Final Flow Statistics
,----[ FLOWCACHE STATS ]----------
Memcap: 10485760 Overhead Bytes 16400 used(%0.166645)/blocks (17474/7) Overhead blocks: 1
    Could Hold: (58579)
IPV4 count: 6 frees: 0 low_time: 1206545212, high_time: 1206545391, diff: 0h:02:59s
   finds: 33 reversed: 4(%12.121212)
   find_sucess: 27 find_fail: 6 percent_success: (%81.818182) new_flows: 6
 Protocol: 1 (%6.060606) finds: 2  reversed: 0(%0.000000)
  find_sucess: 1 find_fail: 1 percent_success: (%50.000000) new_flows: 1
 Protocol: 6 (%84.848485) finds: 28  reversed: 4(%14.285714)
  find_sucess: 26 find_fail: 2 percent_success: (%92.857143) new_flows: 2
 Protocol: 17 (%9.090909) finds: 3  reversed: 0(%0.000000)
  find_sucess: 0 find_fail: 3 percent_success: (%0.000000) new_flows: 3
Snort exiting
s06208150:/var/log/snort# ls -lah
total 20K
drwxr-s---  2 snort adm  4.0K 2008-03-26 23:26 .
drwxr-xr-x 11 root  root 4.0K 2008-03-26 22:10 ..
-rw-r-----  1 snort adm   125 2008-03-26 23:27 alert
-rw-r-----  1 snort adm   344 2008-02-22 14:40 alert.1.gz
-rw-------  1 root  adm  1.2K 2008-03-26 23:27 tcpdump.log.1206545197
```

A Simple Test: the Defense (4)

```
s06208150:/var/log/snort# cat alert
03/26-23:27:06.292693  [**] [105:1:1]
   (spo_bo) Back Orifice Traffic detected [**]
   {UDP} 10.10.10.1:60649 ->
   10.10.10.10:9080
```

- The content tell us that the attack was captured and stored in the alert file for later analysis.

A Simple Test: the Hero

```
# bo: Back Orifice detector
# -------------------------
# Detects Back Orifice traffic on the network.  This
        preprocessor
# uses the Back Orifice "encryption" algorithm to search
        for
# traffic conforming to the Back Orifice protocol (not
        BO2K).
# This preprocessor can take two arguments.  The first
        is "-nobrute"
# which turns off the plugin´s brute forcing routine
        (brute forces
# the key space of the protocol to find BO traffic).  The
        second
# argument that can be passed to the routine is a
        number to use
# as the default key when trying to decrypt the
        traffic.  The
# default value is 31337 (just like BO).  Be aware that
        turning on
# the brute forcing option runs the risk of impacting the
        overall
# performance of Snort, you´ve been warned...
# The Back Orifice detector uses Generator ID 105 and
        uses the
# following SIDS for that GID:
#  SID     Event description
# -----   -------------------
#   1      Back Orifice traffic detected
```

```
preprocessor bo

preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first
        detect_anomalies
preprocessor stream5_global: max_tcp 8192, track_tcp
        yes, \
#preprocessor stream5_tcp: policy first,
        use_static_footprint_sizes
preprocessor http_inspect: global \
preprocessor http_inspect_server: server default \
preprocessor ftp_telnet: global \
preprocessor ftp_telnet_protocol: telnet \
preprocessor ftp_telnet_protocol: ftp server default \
preprocessor ftp_telnet_protocol: ftp client default \
preprocessor smtp: \
preprocessor sfportscan: proto  { all } \
preprocessor dcerpc: \
preprocessor dns: \
```

A Simple Test: the Hero’s Mother

- It's You!
- You are the person who know the importance of security, you are the person who know the value of security
- Security will grow flourish with your understanding, and your support
- Security is necessary and is good for every one

A Simple Test: Conclusion

- Technical/policy view
  - If the target has not patched up to date and be protected well by different security mechanisms, it will be cracked finally, if the there is enough time.
- Business/management view
  - Information security just like insurance, extra security need extra cost, continuous security need continuous pay.

References (1)

- http://www.cert.org/archive/pdf/IEEE_IDS.pdf
- http://dcs.ics.forth.gr/Activities/papers/digenis_new.pdf
- http://woozle.org/~mfisk/papers/setmatch-raid.pdf
- http://www.win.tue.nl/~watson/2R080/opdracht/p333-aho-corasick.pdf
- http://www.informit.com/content/images/0131407333/downloads/0131407333.pdf
- http://www.nersc.gov/nusers/security/TheSpinningCube.php
- http://www.raid-symposium.org/raid98/Prog_RAID98/Full_Papers/Sommer_text.pdf
- http://www.snort.org/
- http://hms.harvard.edu/hmsit/pg.asp?pn=security_glossary
- http://articles.techrepublic.com.com/5100-10925_11-6045911.html
- http://linuxmafia.com/presentations/linuxids-xga-sf/img0.html
- http://www.cisco.com/en/US/i/000001-100000/90001-95000/92001-93000/92614.jpg
- http://www.cs.chalmers.se/~sax/pub/survey.ps
- http://www.cisco.com/en/US/i/000001-100000/95001-100000/97001-98000/97332.jpg
- http://netlab.hut.fi/opetus/s38310/02-03/jussila_060503.pdf
- http://westlabp4-7.poly.edu/CS682-Network_Security_Projects/2006-ise-spring-studentsPresentations/08-IDS/IDS-Presentation.ppt
- http://blogs.techrepublic.com.com/security/images/windows%20kernel%20mode.jpg
- http://www.securityfocus.com/infocus/1514
- http://en.wikipedia.org/wiki/Image:IP_stack_connections.svg
- http://www.yesky.com/14/1815014.shtml
- http://www.learnsecurityonline.com/vid/MSF3-VNC/MSF3-VNC.html

References (2)

- http://219.238.6.200/article?code=jos181639&jccode=52
- http://www.cc.gt.atl.ga.us/~wenke/papers/sasn.pdf
- http://www.yesky.com/14/1815014.shtml
- http://en.wikipedia.org/wiki/False_positive
- http://www.snenug.org/ppt/2005/Susan%20Young%20Snort_presentation_v1.14.ppt
- http://www.blackhat.com/presentations/bh-usa-01/MartyRoesch/bh-usa-01-Marty-Roesch.ppt
- http://www.securityfocus.com/infocus/1577
- http://www.sans.org/resources/idfaq/polymorphic_shell.php
- http://en.wikipedia.org/wiki/Polymorphic_code
- http://csrc.ncsl.nist.gov/publications/nistpubs/800-94/SP800-94.pdf
- http://en.wikipedia.org/wiki/Directory_traversal
- http://www.mcafee.com/us/local_content/white_papers/wp_ddt_anomaly.pdf
- http://packetstormsecurity.nl/distributed/stick.htm
- http://www.iv2-technologies.com/~rbidou/HowToTestAnIPS.pdf
- http://www.ussrback.com/docs/papers/IDS/whiskerids.html
- http://www.securityfocus.com/infocus/1577
- http://en.wikipedia.org/wiki/Intrusion_detection_system_evasion_techniques
- http://www.linuxjournal.com/files/linuxjournal.com/linuxjournal/articles/046/4668/4668f2.png
- http://www.ll.mit.edu/IST/ideval/data/1999/1999_data_index.html
- http://www.fistconference.org/data/presentaciones/idswithartificialintelligence.pdf
- http://home.eng.iastate.edu/~julied/publications/NAFIPSpaper2000.pdf

END

- Q&A
  - $\approx$15 mins
- Thanks
  - My professor: John C.S. Lui
  - My classmates: every mate
- Thank you for allotting time to me
- Thanks again!
- Bye!!

About the Project

- Founded in 1999
- International
- non-profit (501c3) *
- research organization
- dedicated to improving Internet security
- Very active

*501(c) is a provision of the United States Internal Revenue Code (26 U.S.C. § 501(c)), listing twenty-seven types of non-profit organizations exempt from some Federal income taxes

About the Project: Goal

- Goal
  - Simply put
    - to make a difference
  - Fully put
    - to improve the security of the Internet by sharing lessons learned about the most common threats
- Open Sourced
- at no cost to the public

About the Project: Approaches

- Awareness
  - raise awareness of the threats and vulnerabilities
  - series of papers: Know Your Enemy
- Information
  - provide details to better secure and defend resources
  - series of papers: Know Your Enemy
  - address of problems: Scan of the Month
- Tools
  - provide tools and techniques
  - honeypots & honeynets etc.

About the Project: Activities

- From it’s establishment in 1999
- On average
  - News releases, 1/wk
  - 25 Online Articles or Whitepapers, 3.14/yr
  - 41 Academic or Scientific Papers, 5.85/yr
  - 18 Conference Presentations, 2.57/yr
  - 15 tools, 2.14/yr
  - 1 book, Know Your Enemy, 2nd Edition

About the Project: Challenges

- Scan of the Month Challenges
  - decode attacks in the wild
  - 90MB, monthly
- The Reverse Challenge
  - decode binaries captured in the wild
  - 27MB, 06 May to 31 May, 2002
- The Forensic Challenge
  - conduct full forensic analysises in the wild
  - 12MB, 15 January to 19 February of 2001

BACK