

# Unidad 5

## ***Monitoreo de redes y Sistemas de Detección de Intrusiones***

# Network Security Monitoring

The Tao of Network Security Monitoring: Beyond Intrusion Detection,  
Richard Bejtlich, Addison-Wesley, 2004.

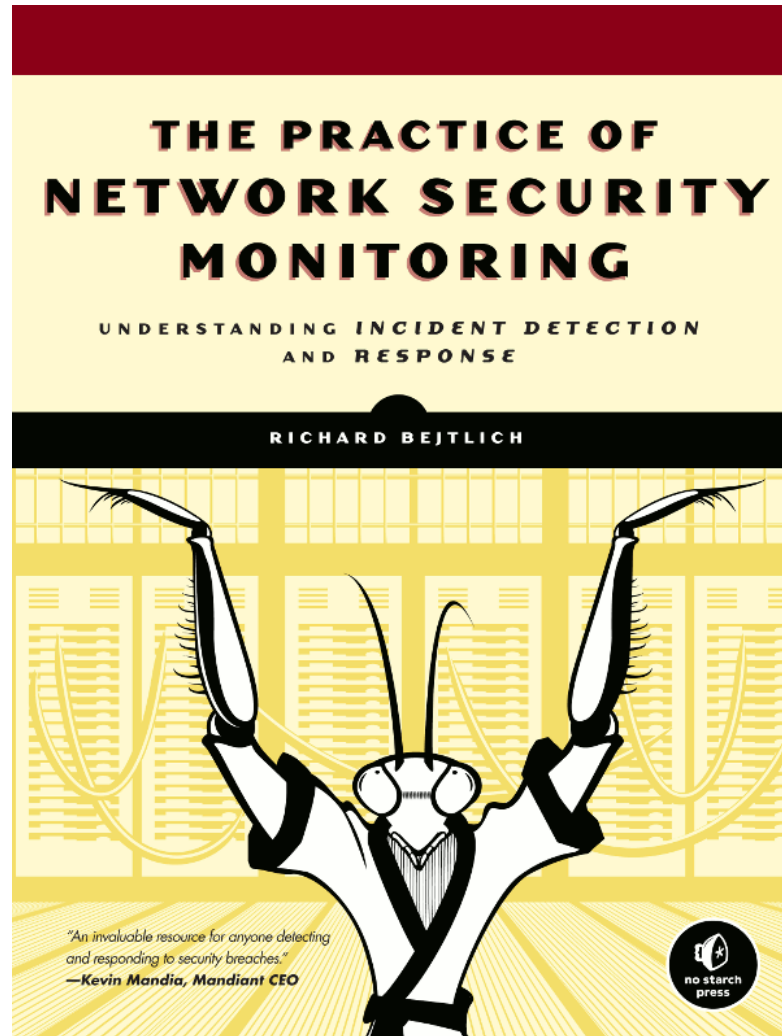


THE TAO OF  
NETWORK  
SECURITY  
MONITORING  
Beyond Intrusion Detection



RICHARD BEJTlich  
*Foreword by* RON GULA,  
CTO, Tenable Network Security

# Y su continuación



- **Contenido completo de paquetes (Tcpdump)**
  - Todos los paquetes, incluida la capa de aplicación
  - Alto costo de almacenamiento, pero permite mayor granularidad en el análisis.
- **Datos de sesiones (Argus, SANCP, NetFlow)**
  - Resumen de conversaciones entre sistemas,
  - Compacto, independiente de los datos; el cifrado no es un problema.
- **Datos estadísticos (Capinfos, Tcpdstat)**
  - Visión de alto nivel de eventos sumarios.
- **Datos de Alerta (Snort, otros IDSs)**
  - Alertas de IDS tradicionales.
  - Ya sea por regla o por anomalía.

- **Proposito**

- Almacenar los paquetes completos brinda maxima flexibilidad para el análisis.
- Los paquetes pueden ser posteriormente analizados por múltiples herramientas de análisis.
- Otras herramientas están limitadas por el tipo de selección que realizan.
- Mayores posibilidades de análisis forense post-incidente.
- El cifrado oculta el contenido pero no los encabezados (en túneles, se ven los extremos)

## Proposito

Interpretar encabezados IP, TCP, UDP e ICMP y resumir el tráfico en formato de conversaciones o sesiones.

Generar tablas de sesión sin almacenar los encabezados ni datos.

No puede ser engañado por encriptación, porque no tiene en cuenta los datos de aplicación.

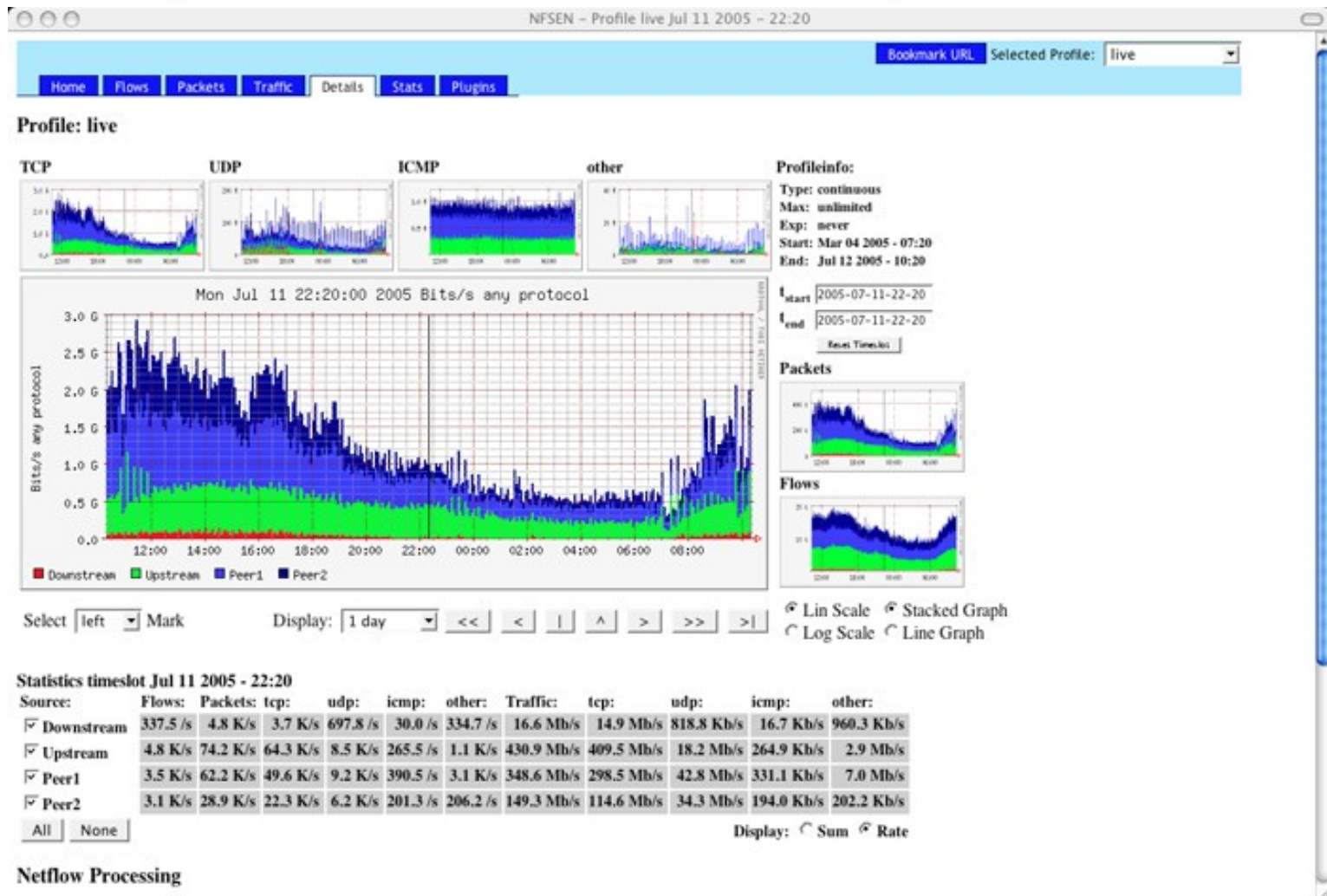
Ejemplos: Netflow, Argus, SANCP

# Ejemplo Argus (<https://openargus.org/>)

timestamp	protocol	src IP	direction	dst IP	status
17 Apr 02 09:59:16	icmp	192.172.1.26	<->	192.172.1.253	ECO
17 Apr 02 09:59:16	tcp	192.172.191.46.458	->	207.68.162.24.80	FIN
17 Apr 02 09:59:16	icmp	192.172.1.25	<->	192.172.1.253	ECO
17 Apr 02 09:59:16	tcp	192.18.221.25.119	->	192.172.191.61.25	FIN
17 Apr 02 09:59:16	tcp	192.172.1.6.3562	->	209.10.33.195.80	FIN
17 Apr 02 09:59:16	tcp	192.172.1.23.5936	->	61.200.81.153.80	EST
17 Apr 02 09:59:16	tcp	192.172.191.46.4585	->	64.4.30.24.80	FIN
17 Apr 02 09:59:17	tcp	192.172.191.46.4990	->	12.12.162.203.80	RST
17 Apr 02 10:00:04	tcp	192.172.191.46.240	->	216.33.240.24.80	RST
17 Apr 02 09:59:17	tcp	142.177.221.77.177	->	192.172.18.27.634	RST
17 Apr 02 10:00:02	icmp	192.172.1.25	->	192.172.1.253	ECO
17 Apr 02 10:00:02	icmp	129.82.45.220	->	192.172.1.3	ECO
17 Apr 02 10:00:02	icmp	129.82.45.220	->	192.172.1.3	ECO
17 Apr 02 10:00:02	udp	205.158.62.41.967	->	192.172.191.6.53	TIM
17 Apr 02 10:00:02	icmp	129.82.45.220	->	192.172.1.3	ECO

**También puede mostrar cantidad de Paquetes y cantidad de bytes**

# NFSen (<http://nfsen.sourceforge.net>)





## Proposito

Primera visión de un archivo con contenido completo.

Estadísticas de uso de protocolos

Equipos que generan mayor cantidad de tráfico

Ejemplos: tcpdstat, trafshow, ntop

# Ejemplo tcpdstat

```
StartTime: Wed Oct 1 18:58:04 2003
EndTime:   Wed Oct 1 20:01:08 2003
# of packets: 7768 (3.33MB)
```

```
### IP address Information ###
```

```
# of IPv4 addresses: 9
```

```
### Protocol Breakdown ###
```

```
<<<<
```

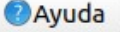
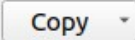
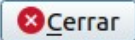
	protocol	packets	bytes	bytes/pkt
[0]	total	7768 (100.00%)	3496942 (100.00%)	450.17
[1]	ip	7752 ( 99.79%)	3495982 ( 99.97%)	450.98
[2]	tcp	7723 ( 99.42%)	3491796 ( 99.85%)	452.13
[3]	http(s)	913 ( 11.75%)	816137 ( 23.34%)	893.91
[3]	http(c)	302 (  3.89%)	28309 (  0.81%)	93.74
[3]	ftp	11 (  0.14%)	828 (  0.02%)	75.27
[3]	other	6497 ( 83.64%)	2646522 ( 75.68%)	407.35
[2]	udp	28 (  0.36%)	4116 (  0.12%)	147.00
[3]	other	28 (  0.36%)	4116 (  0.12%)	147.00
[2]	icmp	1 (  0.01%)	70 (  0.00%)	70.00

# Estadísticas Wireshark

Wireshark · Protocol Hierarchy Statistics · http\_with\_jpegs.cap.gz

Protocol	Percent Packets	Packets	Percent Bytes	Bytes
▼ Frame	100.0	483	100.0	319002
▼ Ethernet	100.0	483	2.1	6762
▼ Internet Protocol Version 4	100.0	483	3.0	9660
▼ Transmission Control Protocol	96.1	464	90.4	288289
▼ Hypertext Transfer Protocol	7.9	38	86.9	277354
Media Type	0.2	1	0.1	433
Line-based text data	1.0	5	2.6	8394
JPEG File Interchange Format	1.0	5	71.6	228534
Data	3.9	19	4.4	13984

No display filter.

Se denomina Detección de Intrusiones al proceso de monitorear los eventos que ocurren en un sistema o red de computadoras, analizándolos en busca de señales que indiquen incidentes de seguridad.

Se puede pensar en sistemas análogos en otras áreas, como por ejemplo las alarmas anti-robo, o sistemas de vigilancia con videocámaras .

# Problemas comunes de los IDS

**Falsos positivos:** Ocurre cuando la herramienta clasifica una acción como una posible intrusión cuando se trata de un comportamiento legítimo, que no constituye una violación de seguridad. Muchas veces se pueden disminuir los falsos positivos haciendo más específicas las descripciones de los patrones de detección de una vulnerabilidad.

**Falsos negativos:** se producen ataques y los mismos no son detectados por el IDS.

**Falsas alarmas (o ruido):** aquellas alertas generados en base a datos que forman parte de un ataque, pero donde el ataque no representa un peligro. Este hecho puede deberse a varias razones, entre las que se destacan tres:

- El ataque afecta a una plataforma distinta a la atacada.
- La vulnerabilidad no existe en el objetivo atacado.
- El ataque no logró alcanzar el objetivo.

- **Detectar una amplia variedad de intrusiones**
  - Ataques conocidos y desconocidos.
  - Esto sugiere la necesidad de aprender/adaptarse a nuevos ataques o cambios en comportamiento.
- **Detectar las intrusiones en un tiempo razonable**
  - Puede ser necesario que sea en tiempo real, especialmente cuando el sistema responde ante una intrusión.
    - Problema: El análisis puede impactar directamente en los tiempos de respuesta del sistema.
  - Puede ser suficiente reportar intrusiones ocurridas hace algunos minutos o hace algunas horas.

- **Presentar la información en una forma fácil de entender**
  - Idealmente, con un indicador binario.
  - Usualmente más complejo, permitiendo al analista examinar la información relacionada con el supuesto ataque.
  - La interface de usuario es crítica, especialmente cuando se monitorean muchos sistemas.
- **Ser preciso**
  - Minimizar falsos positivos y falsos negativos.
  - Minimizar el tiempo utilizado para buscar y verificar ataques.

# Clasificación de IDS

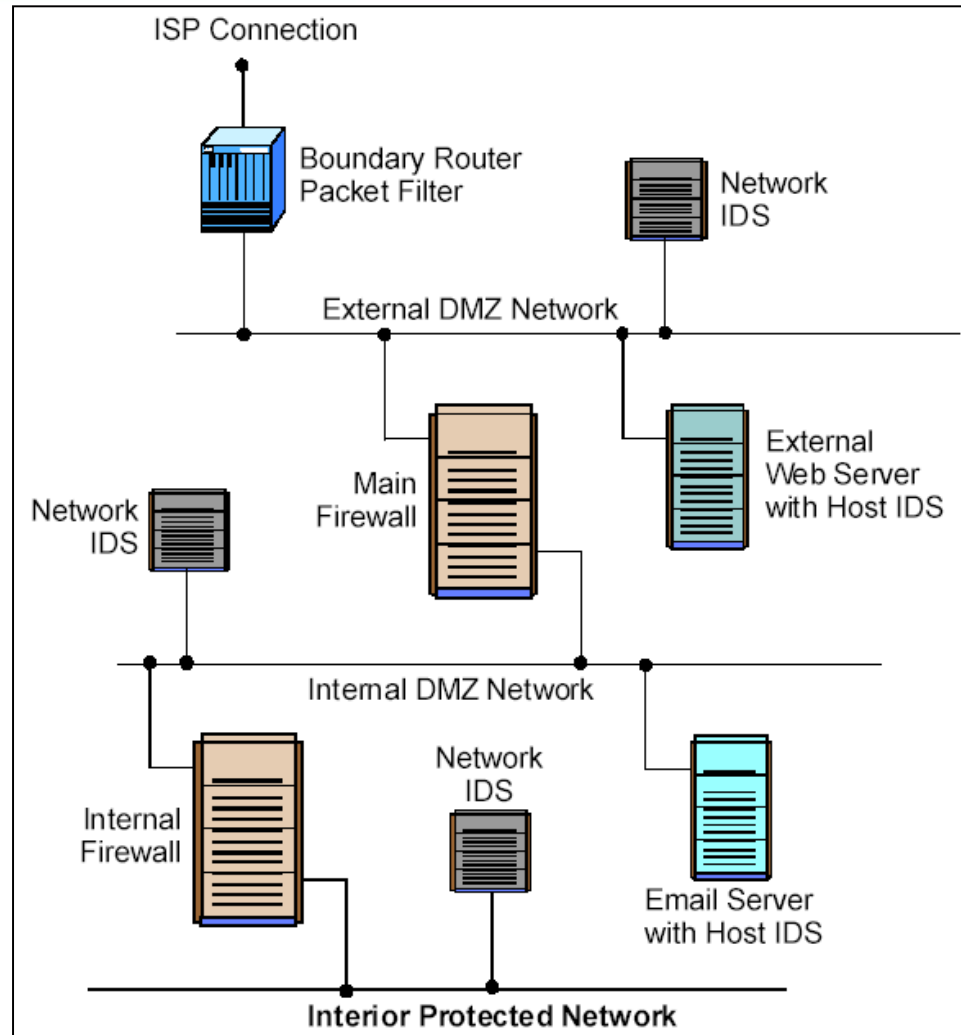
Ségún el ámbito de ejecución	Según el método de detección
IDS de Host (HIDS)	Basado en patrones conocidos
IDS de Red (NIDS)	Basado en heurísticas o estadística



- **NIDS**
  - Reconocimiento de patrones en el tráfico de red para identificar intentos de ataques conocidos
- **HIDS**
  - Monitoreo automático de Logs
  - Chequeo de integridad de archivos

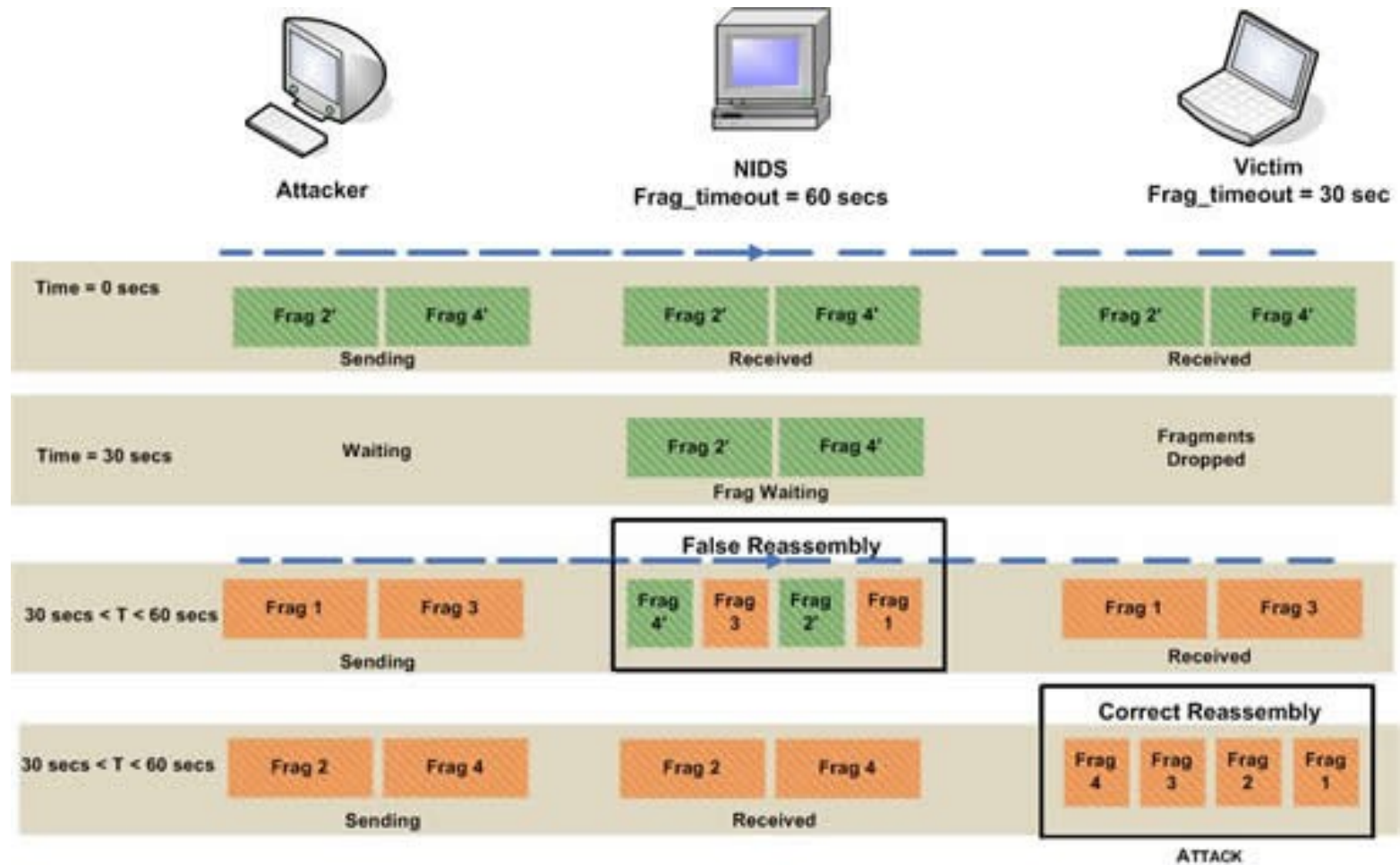
- **NIDS**
  - Monitoreo estadístico de tráfico para determinar actividad de DoS, escaneo de puertos, brute-force login
- **HIDS**
  - Comportamiento extraño de usuarios
  - Parametros del sistema excedidos: Logins fallidos, Carga de CPU, etc.

# ¿Dónde ubicar el IDS?



- **Inserción:** Un NIDS puede aceptar un paquete que el sistema final rechaza. Un atacante puede aprovechar esto para “insertar” alertas que no son tenidas en cuenta por el sistema final.
- **Evasión:** Un sistema final puede aceptar un paquete que un IDS rechaza. Un IDS que por error descarta el paquete pierde completa su contenido. Estos paquetes pueden contener ataques que el IDS no detecta.

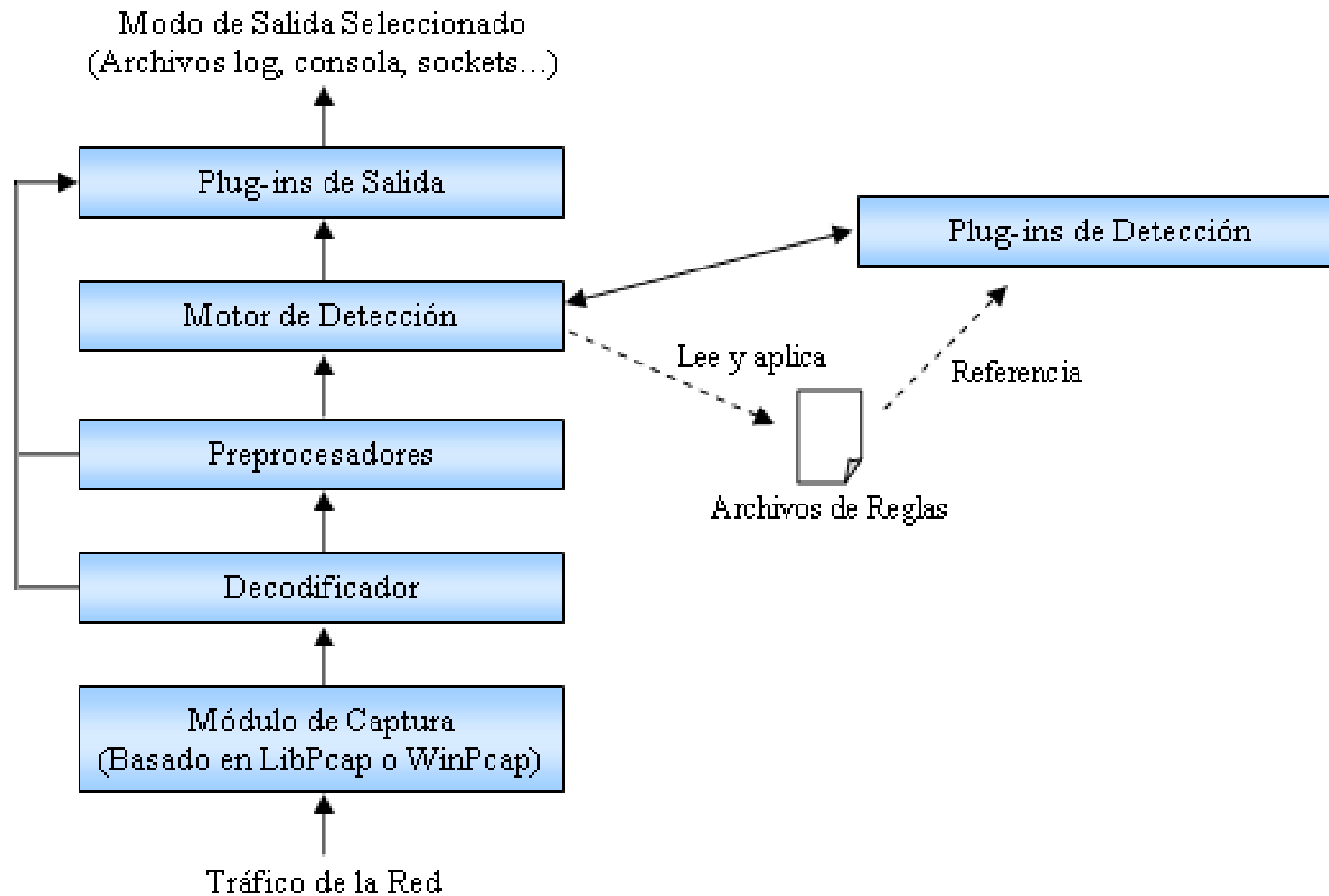
# Ejemplo: Fragmentación IP



# Ejemplo NIDS: SNORT

- Snort es un NIDS basado en patrones, y usa reglas para chequear paquetes que pasan por la red.
- Una regla es un conjunto de requerimientos que debe tener un evento (paquete, conjunto de paquetes reensamblados o flujo de datos), para disparar un alerta.
- Disponible en: <http://www.snort.org>

# Arquitectura



# Snort: Componentes básicos

**Sniffer:** es el encargado de capturar todos los paquetes de la red. Utiliza la librería libpcap para realizar esta tarea.

**Decodificador:** Arma estructuras de datos con los paquetes capturados e identifica los protocolos de enlace, red, etc. Puede detectar problemas en encabezados.

**Preprocesadores:** permiten extender las funcionalidades preparando los datos para la detección. Se realizan tareas como la detección de escaneos, Ensamblado de datos contenidos en paquetes de una misma sesión o reensamblado de paquetes IP fragmentados.

**Motor de detección:** analiza los paquetes en base a las reglas definidas para detectar ataques.

**Salida o Postprocesadores:** permiten definir qué, cómo y dónde se guardan las alertas y los correspondientes paquetes de red que las generaron. Pueden utilizar archivos de texto, traps SNMP, bases de datos, syslog, etc.



## HTTP\_inspect

Este preprocesador es utilizado para procesar las URIs contenidas en las solicitudes HTTP. Su principal función consiste en normalizar las URIs para que aparezcan en su forma más simple y unívoca. Algunas técnicas de evasión de IDS se basan en escribir los URIs de un request HTTP de manera distinta para que el IDS no detecte el ataque, y en este hecho reside la gran utilidad del preprocesador `http_inspect`.

# Ejemplos Reglas Snort

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS  
(msg:"WEB-CGI HyperSeehsx.cgi directory traversal attempt";  
uricontent:"/hsx.cgi";content:"../..";content:"%00";  
flow:to_server,established; reference:bugtraq,2314;  
reference:cv,CAN-2001-0253; classtype:web-application-attack;  
sid:803; rev:6;)
```

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any  
(msg:"P2P BitTorrent announce request";  
flow:to_server,established; content:"GET"; depth:4;  
content:"/announce"; distance:1; content:"info_hash="; offset:4;  
content:"event=started"; offset:4;  
classtype:policy-violation; sid:2180; rev:2;)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (msg:  
" MS15-034 Range Header HTTP.sys Exploit"; content: "|  
0d 0a|Range: bytes="; nocase; content: "-"; within: 20 ;  
byte_test: 10,>,1000000000,0,relative,string,dec ; sid:  
1001239;)
```

*(byte\_test is limited to 10 bytes, so I just check if the first 10 bytes are larger then 1000000000)*

**Ojo con las formas de saltar reglas:**

**<http://blog.didierstevens.com/2015/04/17/ms15-034-detection-some-observations/>**

# Alertas generadas por Snort

## **[\*\*] [1:1444:3] TFTP Get [\*\*]**

[Classification: Potentially Bad Traffic] [Priority: 2]

10/30-12:11:07.031155 200.59.73.153:33206 -> 200.59.103.1:69

UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:50 DF

Len: 22

## **[\*\*] [1:2404:6] NETBIOS SMB-DS Session Setup AndX request unicode username overflow attempt [\*\*]**

[Classification: Attempted Administrator Privilege Gain] [Priority: 1]

10/30-12:15:18.219433 200.59.105.60:1431 -> 200.59.73.153:445

TCP TTL:46 TOS:0x0 ID:42991 IpLen:20 DgmLen:1484 DF

\*\*\*AP\*\*\* Seq: 0xE03BC43C Ack: 0xBF9A96C1B Win: 0xFFBF TcpLen: 20

[Xref => <http://www.eeye.com/html/Research/Advisories/AD20040226.html>]

[Xref => <http://www.securityfocus.com/bid/9752>]

**Plataforma integral para el análisis del tráfico de red. Si bien a menudo se compara con los sistemas clásicos de detección / prevención de intrusiones, Zeek adopta un enfoque diferente al proporcionar a los usuarios un marco flexible que facilita el monitoreo personalizado y en profundidad mucho más allá de las capacidades de los sistemas tradicionales.**

**Genera logs de “transacciones” de ciertos protocolos como http, ssh, smtp, certificados SSL, DNS**

# Ejemplo log zeek

- 2013-01-16T19:09:47+0000 90E6goBBSw3  
192.168.238.152 41482 217.160.51.31 80 1 GET  
www.testmyids.com / - Mozilla/5.0 (X11; Ubuntu; Linux  
x86\_64; rv:18.0) Gecko/20100101 Firefox/18.0 0 39  
**200** OK - - - (empty) - - - text/plain - -
- 2013-01-16T19:09:47+0000 90E6goBBSw3  
192.168.238.152 41482 217.160.51.31 80 2 GET  
www.testmyids.com /favicon.ico - Mozilla/5.0 (X11;  
Ubuntu; Linux x86\_64; rv:18.0) Gecko/20100101  
Firefox/18.0 0 640 **404** Not Found - - - (empty) - - -  
text/html - -

**Swatch comenzó siendo el "simple watchdog" para monitorear actividad de log producida por el syslog de UNIX. Fue evolucionando para permitir monitorear otro tipo de logs.**

**Permite definir entradas a resaltar o a ignorar.**

**Similar a swatch, con reglas muy actualizadas para Debian. Normalmente se ejecuta cada una hora, revisa los logs del sistema en busca de ciertas entradas anormales, y genera un reporte que puede ser enviado por correo electrónico.**



## Security Events

=====

```
Aug  1 09:09:34 matute sshd[1401]: (pam_unix) authentication
failure; logname= uid=0 euid=0 tty=ssh ruser=
rhost=10.xx.yy.26
Aug  1 09:09:36 matute sshd[1401]: error: PAM: User not
known to the underlying authentication module for illegal
user n3ssus from 10.xx.yy.26
```

## System Events

=====

```
Aug  1 09:21:31 matute sshd[2473]: Did not receive
identification string from 10.xx.yy.26
Aug  1 09:21:37 matute sshd[2474]: Bad protocol version
identification 'GET / HTTP/1.0' from 10.xx.yy.26

Jul 30 11:55:43 matute Incripcion[25721]: desde
200.xx.yy.132 - invalid: cdi:20-12430832|
```

- Permite al administrador monitorear la actividad relacionada con agregado, borrado y modificación de archivos. Genera una base de datos con la información de cada archivo en el sistema. Periodicamente, vuelve a generar la información y la compara.
- Chequea Archivos nuevos, eliminados, y cambios en atributos de archivos:
  - Tamaño,
  - Fecha y hora de acceso y modificación,
  - Permisos,
  - Firmas o hashes

# Ejemplo Reporte Aide

```
Envelope-to: root@localhost.localdomain
Delivery-date: Fri, 14 Apr 2006 06:25:16 -0300
To: root@localhost.localdomain
Subject: Daily AIDE report for localhost.localdomain
From: root <root@localhost.localdomain>
```

This is an automated report generated by the Advanced Intrusion Detection Environment on localhost.localdomain at 2006-04-14 06:25.

added:/usr/bin/tftp

File: /bin/tar

Size	: 163820	, 163852
Bcount	: 328	, 336
Mtime	: 2004-08-03 11:31:59	, 2006-02-24 18:21:24
Ctime	: 2006-01-27 09:24:39	, 2006-03-27 19:56:39
Inode	: 2326559	, 2326584
MD5	: lqHdZO5kJKbPp4OQFdmNZw==	, 000qYuZFk3VRPSEaK8Xp+w==
SHA1	: sB7B1di8CecXlpNZ16kw/kSLVVs=	, Xu6ulr5mPWN0T4iB3w4Dp9KRp58=

- **OSSEC es un HIDS Open source. Realiza análisis de logs, chequeos de integridad de archivos, monitoreo de registry de windows, detección de rootkits, etc**
- **Corre en la mayoría de los sistemas operativos, incluyendo Linux, OpenBSD, FreeBSD, MacOS, Solaris y Windows.**

# Ejemplos OSSEC

OSSEC HIDS Notification. 2007 Aug 14 13:20:23

**Received From: monitor->/var/log/apache2/error.log**

**Rule: 30109 fired (level 9) -> "Attempt to login using a non-existent user."**

**Portion of the log(s):**

**[Tue Aug 14 13:20:23 2007] [error] [client 10.xx.yy.51] user test not found: /nagios2/,  
referer: http://monitor.arcert.gov.ar/**

OSSEC HIDS Notification.2007 Aug 14 13:08:05

**Received From: monitor->/var/log/auth.log**

**Rule: 5701 fired (level 12) -> "Possible attack on the ssh server (or version  
gathering)."**

**Portion of the log(s):**

**Aug 14 13:08:04 monitor sshd[21642]: Bad protocol version identification 'quit' from  
UNKNOWN**

# Ejemplos OSSEC

OSSEC HIDS Notification. 2006 Sep 06 23:15:21

**Received From: (xx) 1.2.3.4->/usr/pages/xx/logs/web.access\_log**

**Rule: 31151 fired (level 10) -> "Mutiple web server 400 error codes from same source ip."**

**Portion of the log(s):**

**64.46.38.151 - - [06/Sep/2006:23:14:41 -0300] "POST /xmlsrv/xmlrpc.php HTTP/1.1" 404 223 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:14:41 -0300] "POST /xmlrpc/xmlrpc.php HTTP/1.1" 404 223 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:14:40 -0300] "POST /xmlrpc.php HTTP/1.1" 404 216 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:14:39 -0300] "POST /xmlrpc.php HTTP/1.1" 404 216 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:13:52 -0300] "POST /xmlsrv/xmlrpc.php HTTP/1.1" 404 223 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:13:52 -0300] "POST /xmlrpc/xmlrpc.php HTTP/1.1" 404 223 "-" "Internet Explorer 6.0"**

**64.46.38.151 - - [06/Sep/2006:23:13:50 -0300] "POST /xmlrpc.php HTTP/1.1" 404 216 "-" "Internet Explorer 6.0"**

# Ejemplos OSSEC

OSSEC HIDS Notification. 2006 Oct 24 18:46:29

**Received From: (xx) 200.1.2.a->/var/log/maillog**

**Rule: 3354 fired (level 12) -> "Multiple misuse of SMTP service (bad sequence of commands)."**

**Portion of the log(s):**

**postfix/smtpd[6741]: NOQUEUE: reject: RCPT from unknown[201.82.55.24]: 503 <nplxfbtk@fbi.com>: Sender address rejected: Improper use of SMTP command pipelining; from=<nplxfbtk@fbi.com> to=<x@x.br> proto=SMTP helo=<ran-2h991bqbujq>**

**postfix/smtpd[6741]: NOQUEUE: reject: RCPT from unknown[201.82.55.24]: 503 <nplxfbtk@fbi.com>: Sender address rejected: Improper use of SMTP command pipelining; from=<nplxfbtk@fbi.com> to=<x@xl.org.br> proto=SMTP helo=<ran-2h991bqbujq>**

**postfix/smtpd[6741]: NOQUEUE: reject: RCPT from unknown[201.82.55.24]: 503 <nplxfbtk@fbi.com>: Sender address rejected: Improper use of SMTP command pipelining; from=<nplxfbtk@fbi.com> to=<y@y.org.br> proto=SMTP helo=<ran-2h991bqbujq>**

# Security Onion

(<https://securityonionsolutions.com/>)



☰

Security Onion

👤

🏠 Overview

🔔 Alerts

🔍 Hunt

☰ PCAP

📊 Grid

📁 Downloads

🛡️ Administration

Tools

🔗 Kibana

🔗 Grafana

🔗 CyberChef

🔗 Playbook

🔗 Fleet

🔗 TheHive

🔗 Navigator

Alerts

☐

Acknowledged

☐

Escalated

⌛

Never

⌵

Automatic refresh interval

Total Found: 2,021

Time Zone: America/New\_York

🔍

Group By Name, Module

🔍

🕒

Last

24

hours

⌵

REFRESH

🔄

Click the clock icon to change to absolute time

Group: rule.name

Group: event.module

Group: event.severity\_label

	Count	rule.name	event.module	event.severity_label
🔴	298	ET MALWARE Backdoor family PC RAT/Gh0st CnC traffic (OUTBOUND) 102	suricata	high
🔴	297	ET MALWARE Backdoor family PC RAT/Gh0st CnC traffic	suricata	high
🟡	130	ET JA3 Hash - Possible Malware - Various Trickbot/Kovter/Dridex	suricata	low
🔴	65	ET MALWARE ABUSE.CH SSL Blacklist Malicious SSL certificate detected (Dridex/Trickbot CnC)	suricata	high
🟡	60	ET POLICY OpenSSL Demo CA - Internet Widgits Pty (O)	suricata	low
🔴	45	ET MALWARE Gh0st Remote Access Trojan Encrypted Session To CnC Server	suricata	high
🟡	42	ET HUNTING Suspicious GET To gate.php with no Referer	suricata	medium
🟡	42	ET HUNTING Suspicious HTTP Request to .bit domain	suricata	medium
🔴	42	ET MALWARE Generic gate[.]php GET with minimal headers	suricata	high
🔴	41	ET INFO Suspicious Windows NT version 9 User-Agent	suricata	high



- Además de detectar un ataque, los IPS pueden detener al mismo.
- Pueden trabajar a nivel de red o a nivel de host.

Normalmente funciona como dispositivo `in_line`, en modo bridge. Además de detectar alertas en base a patrones, puede decidir filtrar el tráfico para que no llegue a destino.

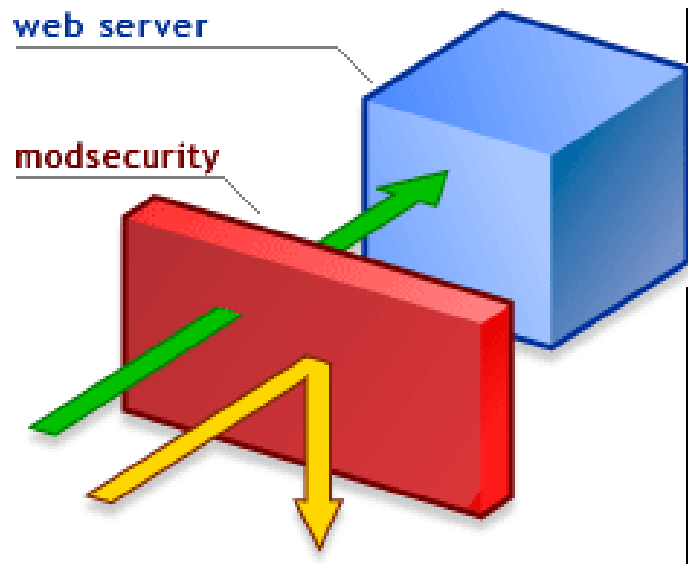
Ej: `Snort Inline`

Utiliza las mismas reglas, pero define nuevas acciones: `block`, `reject`, `sdrop`.

También puede cambiar el contenido del tráfico (`replace`)

## **Ejemplo de Regla:**

```
drop tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"SMTP AUTH user overflow attempt"; flow:to_server,established; content:"AUTH"; nocase; pcre:"/^AUTH\s+\S+\s+[\n]{128}/mi"; reference:bugtraq,13772; classtype:attempted-admin; sid:3824; rev:1;)
```



**ModSecurity es un motor de detección y prevención de intrusiones para aplicaciones web. Operando como un módulo del servidor web apache, el propósito de ModSecurity es aumentar la seguridad de aplicaciones web, protegiéndolas de ataques conocidos y desconocidos.**

**Es un Web application Firewall.**

Intercepta pedidos HTTP antes de que sean procesados en forma completa por el webserver.

Intercepta el cuerpo de los pedidos (ej: en los pedidos de tipo POST)

Intercepta, almacena y opcionalmente valida los archivos subidos.

Realiza acciones anti-evasión en forma automática.

Realiza analisis de los pedidos procesando un conjunto de reglas configurables.

Intercepta la respuestas antes de que sean enviadas al cliente y las analiza en base a reglas.

***# Allow supported request methods only.***

*SecFilterSelective REQUEST\_METHOD !^(GET|HEAD|POST)\$*

***# Require the Host header field to be present.***

*SecFilterSelective HTTP\_Host ^\$*

***# sql injection***

*SecFilterSelective ARGS "delete[[:space:]]+from"*

***# Command "id"***

*SecFilterSelective OUTPUT "uid=[[[:digit:]]]+\([[:alnum:]]+\) gid=[[[:digit:]]+\([[:alnum:]]+\)"*

***#collectors***

*SecFilterSelective HTTP\_USER\_AGENT "autoemailspider"*

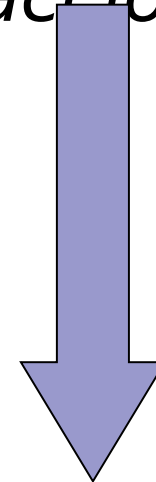
# Honeypots

- **Baja Interacción**
  - Emula Servicios, aplicaciones, SO
  - Bajo riesgo y facil de instalar y mantener, pero captura información limitada.
- **Alta Interacción**
  - Servicios, aplicaciones y SO reales
  - Captura mucha información, pero riesgo alto y consumen mucho tiempo para mantener.

# Ejemplos de honeypots

- **Honeyd**
- **Dionaea**
- **Honeynets**

*Baja  
Interacción*



*Alta interacción*



- Honeyd (<http://www.honeyd.org>)

**Crea equipos virtuales en una red**

**Los equipos pueden ser configurados**

**Para ejecutar servicios arbitrarios**

**Y adaptados para simular distintos**

**Sistemas operativos.**

**Puede reportar a Prelude**



Como el honeyd, es un honeypot de baja interacción que emula vulnerabilidades conocidas, para recolectar información de ataques potenciales. Esta diseñado para simular vulnerabilidades usadas por worms para diseminarse, y capturarlos. Utilizado en el proyecto RECAMAR, del cuál hablaremos en la clase de malware.

Glastopf – Snare/Tanner: web application honeypot

(<https://github.com/mushorg/snare>)

Cowrie: ssh. Interacción media (<https://github.com/cowrie/cowrie>)

Conpot: Sistemas de control industrial (<http://conpot.org/>)

T-POT: agrupa varios honeypots (<https://github.com/dtag-dev-sec/tpotce>)

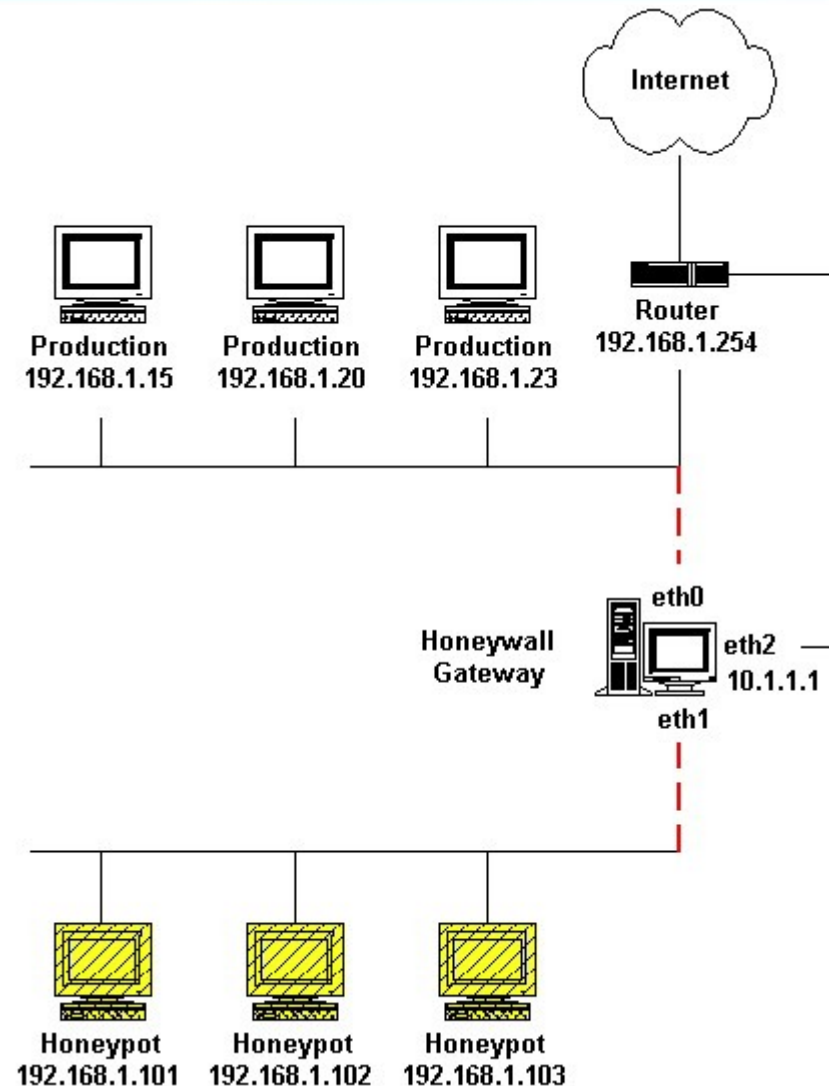
- Honeypots de alta interacción designados para capturar información detallada.
- La información tiene diferentes valores para diferentes organizaciones.
- Es una arquitectura en la cual se agregan equipos enteros, no un producto o un software.
- Todo tráfico entrante y saliente es sospechoso.

**Una red altamente controlada en la que todo paquete entrante o saliente es monitoreado, capturado y analizado.**

- Control de Flujo de Datos
- Captura de Datos
- Análisis de Datos

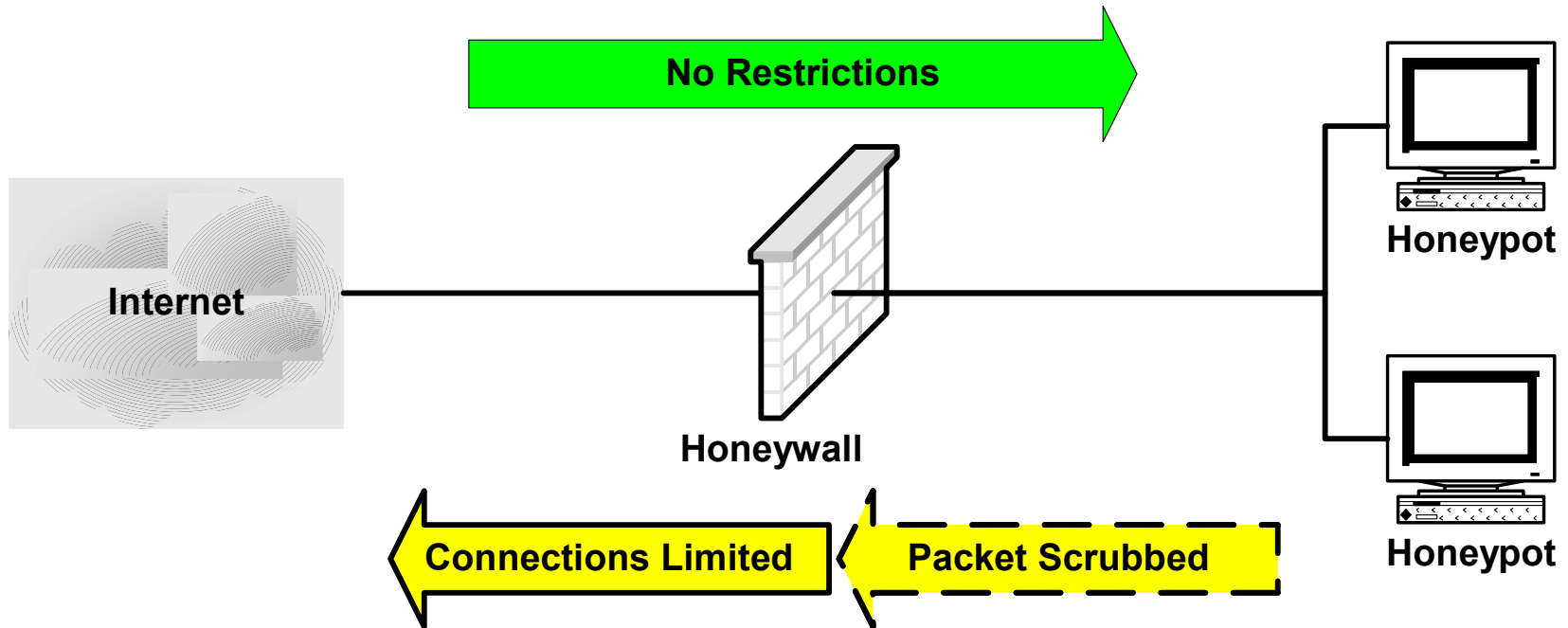
*<http://www.honeynet.org/papers/honeynet/>*

# Arquitectura de honeynets



- Mitigar el riesgo de que la Honeynet sea utilizada para atacar sistemas que no son parte de la honeynet.
- Contar conexiones salientes.
- IPS (Snort-Inline)
- Control de ancho de banda

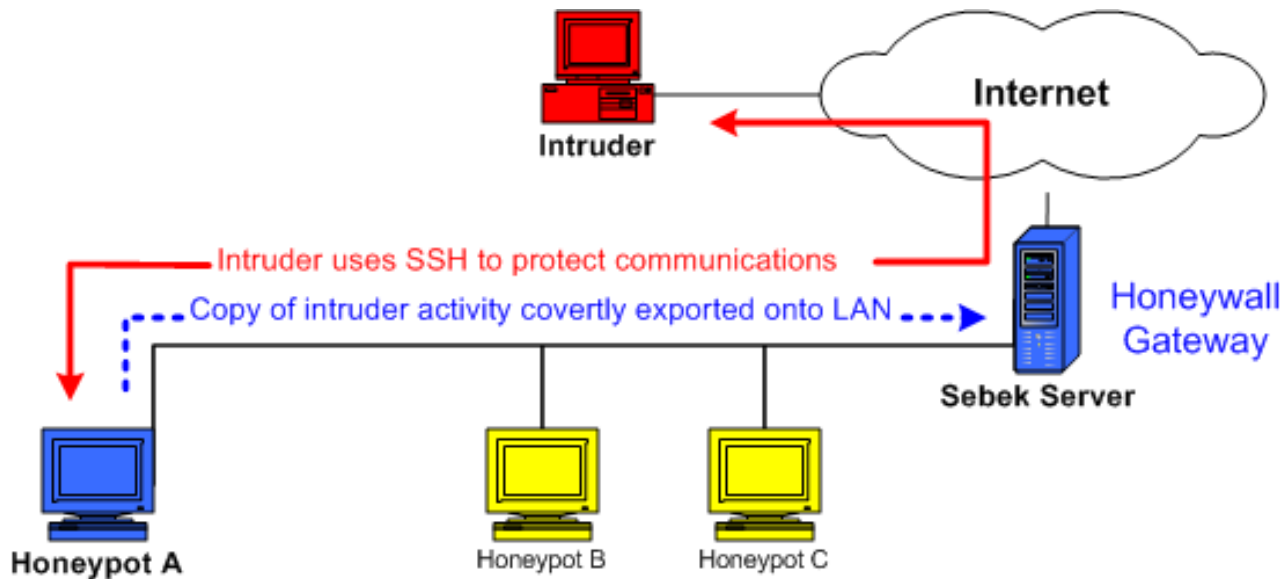
# Control de flujo de datos





- **Capturar toda la actividad a distintos niveles**
- **Actividad de red**
- **Actividad de aplicaciones**
- **Actividad del sistema**

- **Módulo de Kernel oculto que captura la actividad del host**
- **Envía la información a la red**
- **El atacante no puede husmear el tráfico basado en el puerto destino**
- **Posteriormente, se desarrollo qebek. En vez de ser un modulo de kernel, monitorea en la capa de virtualización (en Qemu)**



- **Honeypot de alta interacción. Corre en una VM con distintos navegadores, media players y aplicaciones office. Accede a servidores potencialmente maliciosos, y detecta si la interacción con los mismos genera cambios en el cliente.**