

# Using IPS for Web Protection

***Alejandro Sánchez Acosta***  
***[asanchez@s21sec.com](mailto:asanchez@s21sec.com) S21SEC***

# Agenda

- Web Protection
- Intrusion Detection Systems (IDS)
- Active and response IDS
- IPS as Web Protection
- Advanced Signatures
- Demo
- Conclusions

## Web Protection

- Common risks
  - (XSS, SQL Injection, Command Execution)
- Solutions
  - Static detection (logs & reports)
  - Dynamic detection (application firewall, IDS/IPS)
  - Anomaly detection

## Intrusion Detection Systems (IDS)

- Functions:
  - Capture events
  - Pattern matching
  - Detection
  - Logging
- IDS:
  - HIDS, NIDS and LIDS

## Intrusion Prevention Systems (IPS)

- Prevention systems
- IDS evolution
- Works in Network and Application layer
- Firewall IDS
  - Iptables with string matching
  - Difficult configuration
  - Solutions based in iptables and IDS.
  - snort2c, snort2iptables, snort2pf, etc.
- IPS (ModSecurity or Snort IPS)

# ModSecurity

- Developed by Ivan Ristick
- Easy configuration
- Can work as reverse proxy using ModProxy
- Works in Application layer

# Snort

- Developed by Martin Roesch.
- Operation modes
  - Sniffer, packet logger, NIDS and IPS.
- Output
  - Binary tcpdump, ASCII, Mysql/Postgresql.

## Snort as IPS

- Preprocessors
  - HTTPInspect, Stream4 reassembly
- Web rules configuration
- Response protection
  - (Snort Inline, Snort Divert Mode, Flexresponse)

## Web configuration rules

- Community rules
  - Web-misc, web-cgi, web-dos, web-client, sql-injection
- Snort rules
  - Custom web snort.org rules
- Bleeding Snort

## Cross Site Scripting

- Existing snort signatures:

```
<script>alert(document.cookie)</script>
```

attack:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS  
$HTTP_PORTS (msg:"WEB-MISC cross site scripting  
attempt"; flow:to_server,established;  
content:"<SCRIPT>"; nocase; classtype:web-  
application-attack; sid:1497; rev:6;)
```

<img src=javascript> attack:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS  
$HTTP_PORTS (msg:"WEB-MISC cross site scripting HTML  
Image tag set to javascript attempt";  
flow:to_server,established; content:"img  
src=javascript"
```

## Easy evasion

- Can be trivially evaded using:
  - `<a href="javas&#99;ript&#35;[code]">`
  - `<div onmouseover="[code]">`
  - ``
  - `<xml src="javascript:[code]">`
- Solution: Using complex rules

## Advanced signatures

- Enter PCRE - Perl Compatible Regular Expressions
- Greater flexibility
- One signature can catch multiple attacks
- Lower learning curve for Unix admins – regex is part of daily life
- Regular expressions work with:
  - Snort IDS
  - Apache's mod\_security.

## Regular expressions samples

- XSS:
  - `/((\%3D) | (=)) [^\n]* ((\%3C) | <) [^\n]+ ((\%3E) | >)`
- SQL Injection:
  - `/(\%27) | (\') | (\-\-\) | (%23)`
- Redirection:
  - `site=[^(www.fistconference.org)]`
- Command Execution: `/(system|exec|\|)/`
- Null byte poison: `%00`
- Overflows: `/[\w]+\[^\=]{500,+}\&/`

## Snort Inline

- **Drop**
  - The drop rule tells iptables to drop the packet and log it via usual snort
- **Sdrop**
  - The sdrop rule tells iptables to drop the packet. Nothing is logged
- **Reject**
  - The reject rule type tells iptables to drop the packet; log it via usual snort means; and answer source host

## Replace contents

- **Replacement:**

Another option replaces portions of the payload (disabling the effectiveness of the attack) but allowing the connection to continue:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET
  80
  (msg:"Apache Exploit";flags:A+;
  content:"/bin/sh";
  replace:"/ben/sh");)
```

## Divert Mode

- FreeBSD Inline implementation
  - Use divert sockets and libnet
  - Integrated in current Snort
- Part of snort-inline project
  - Reset, drop, sdrop and replacement

## FlexResponse

- Portable implementation
- Doesn't support replacement
- Rule modification:

```
var RESP_TCP resp:rst_snd;  
var RESP_TCP_URG resp:rst_all;  
var RESP_UDP resp:icmp_port,icmp_host;
```

## Conclusion

- Signature-based IDS is good enough to detect a large majority of initial web app attacks
- It fails in detecting certain unique attacks, such as price manipulation or forceful browsing
- Some signatures may yield large number of false positives
- Maybe best solution is a combination of signature-based to detect majority of simpler attacks, and anomaly-based to detect sophisticated application-specific attacks

## Demo

- ModSecurity
- Snort web rules using FlexResponse

## More information

- Get Snort
  - [www.snort.org](http://www.snort.org)
  - [www.modsecurity.org](http://www.modsecurity.org)
- Rules
  - Bleeding Snort: <http://www.bleedingsnort.org>
  - Community Rules: <http://www.snort.org>

# Creative Commons Attribution-NoDerivs 2.0

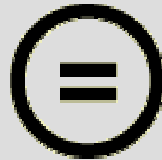
## You are free:

- to copy, distribute, display, and perform this work
- to make commercial use of this work

## Under the following conditions:



**Attribution.** You must give the original author credit.



**No Derivative Works.** You may not alter, transform, or build upon this work.

For any reuse or distribution, you must make clear to others the license terms of this work.

Any of these conditions can be waived if you get permission from the author.

**Your fair use and other rights are in no way affected by the above.**

This work is licensed under the Creative Commons Attribution-NoDerivs License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nd/2.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.