Assinaturas de Ataques

Técnicas de Evasão

Rodrigo Rubira Branco
rodrigo@firewalls.com.br

Objetivos

- Explicar os fundamentos de detecção de intrusos.
- Apresentar as dificuldades existentes em detecção de intrusão.
- Fornecer formas de se analizar assinaturas de ataques.
- Demonstrar formas existentes de se burlar sistemas de detecção de intrusos.

Conceitos de Segurança

CIDAL =

C onfidencialidade
I ntegridade
D isponibilidade
A utenticidade
L egalidade

LEMBRAR-SE SEMPRE DISTO NO DECORRER DA PALESTRA!

Ambiente em Questão

Trabalhamos em um mundo real de sistemas mal configurados:

- **Bugs de Software;**
- **Empregados Insatisfeitos;**
- **Administradores de Sistemas Sobrecarregados;**
- **Acomodação de necessidades empresariais;**
- **Falta de Educação em Segurança;**
- **B2B,B2C,B2E,C2C,X2X ?**

Defense In-Depth

- **Um Firewall apenas aumenta o nível de segurança.**
- **O conceito de segurança em camadas garante que as falhas existentes em um Firewall sejam supridas por outros tipos de defesa e segurança.**
- **Através de diversas camadas cria-se um modelo de segurança robusto e capaz de suportar falhas.**

Quando pensar em Segurança pense em uma CEBOLA.

Recursos In-Depth

- Stack nao executavel

- Endereços de bibliotecas randomizados

- Inserçoes nao contiguas na Stack --> Finalizadores de bloco

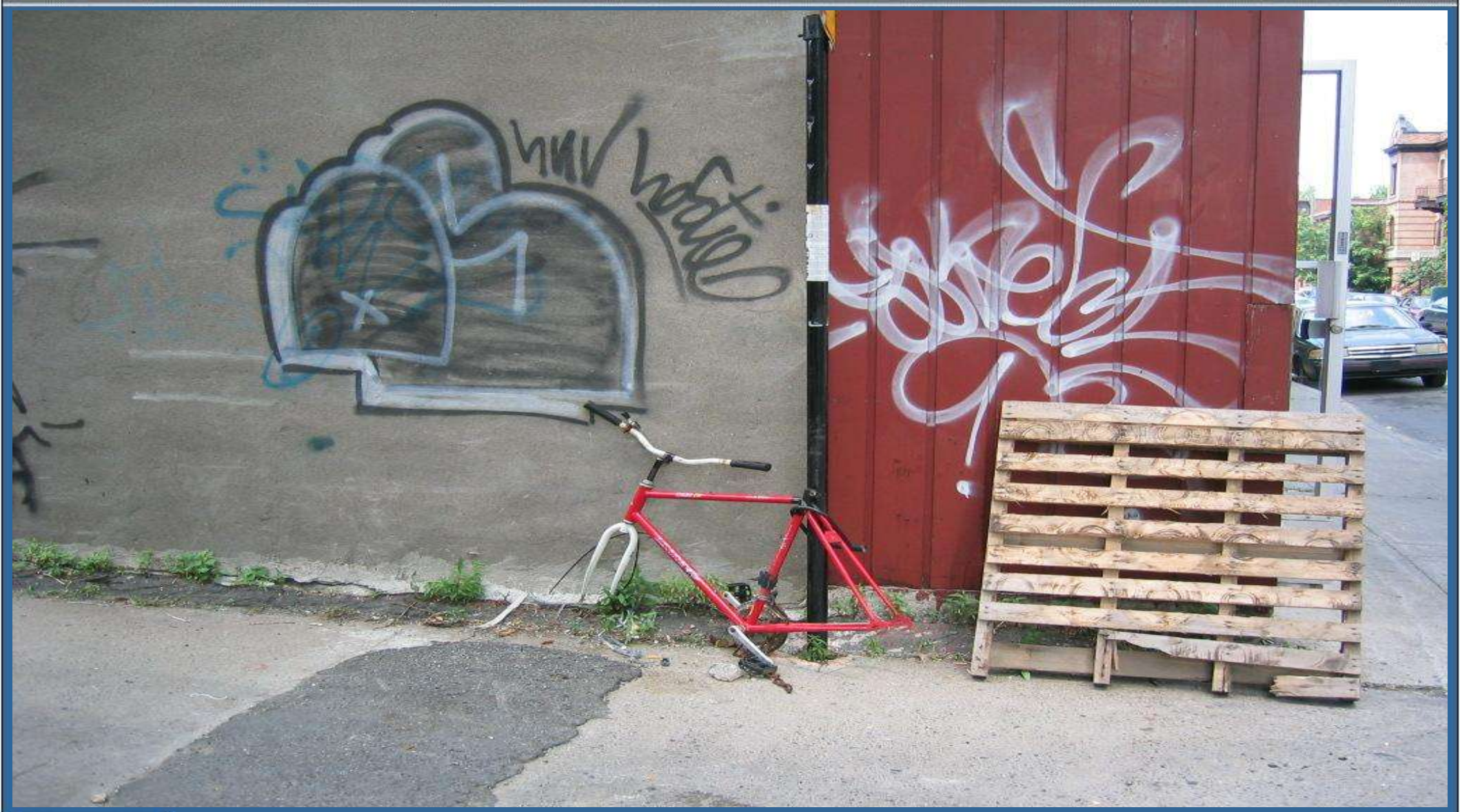
- Heap “segura” -> Insercao de structs de controle

IDS – Por que utilizar?

- **Firewalls podem agir somente sob ataques, IDS detectam as varreduras.**
- **Firewalls agem segundo regras pré-estabelecidas que geralmente não enxergam a camada de aplicação.**

IDS – Por que utilizar?

IDS – Por que utilizar?



IDS – Por que utilizar?

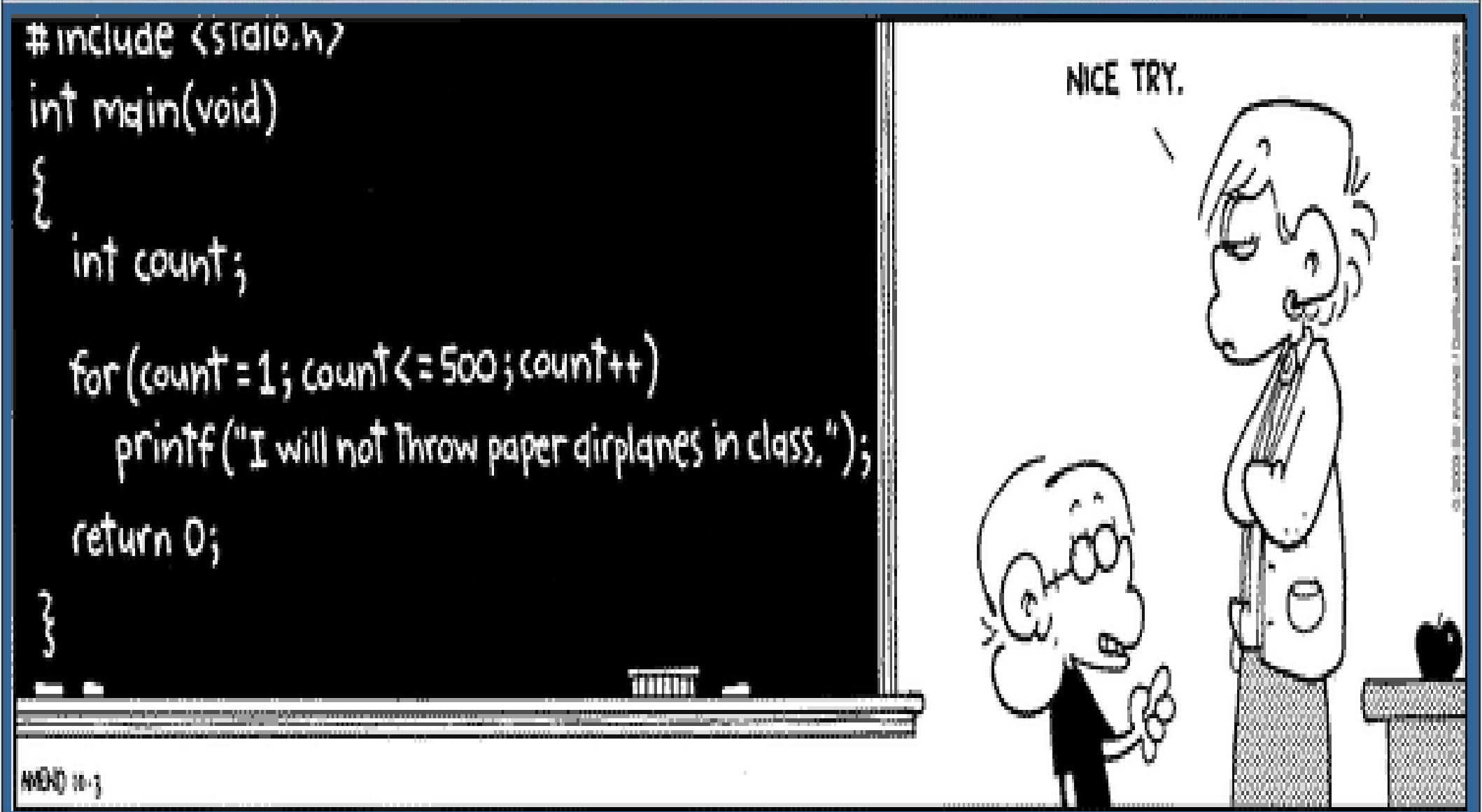
- Anatomia de um Ataque:

  * Detectando informações do alvo;
    nmap -sX -O
    nmap -sX -O -P0
    nmap -sO -O -P0
  * Buscando vulnerabilidades;
  * Explorando vulnerabilidades encontradas.

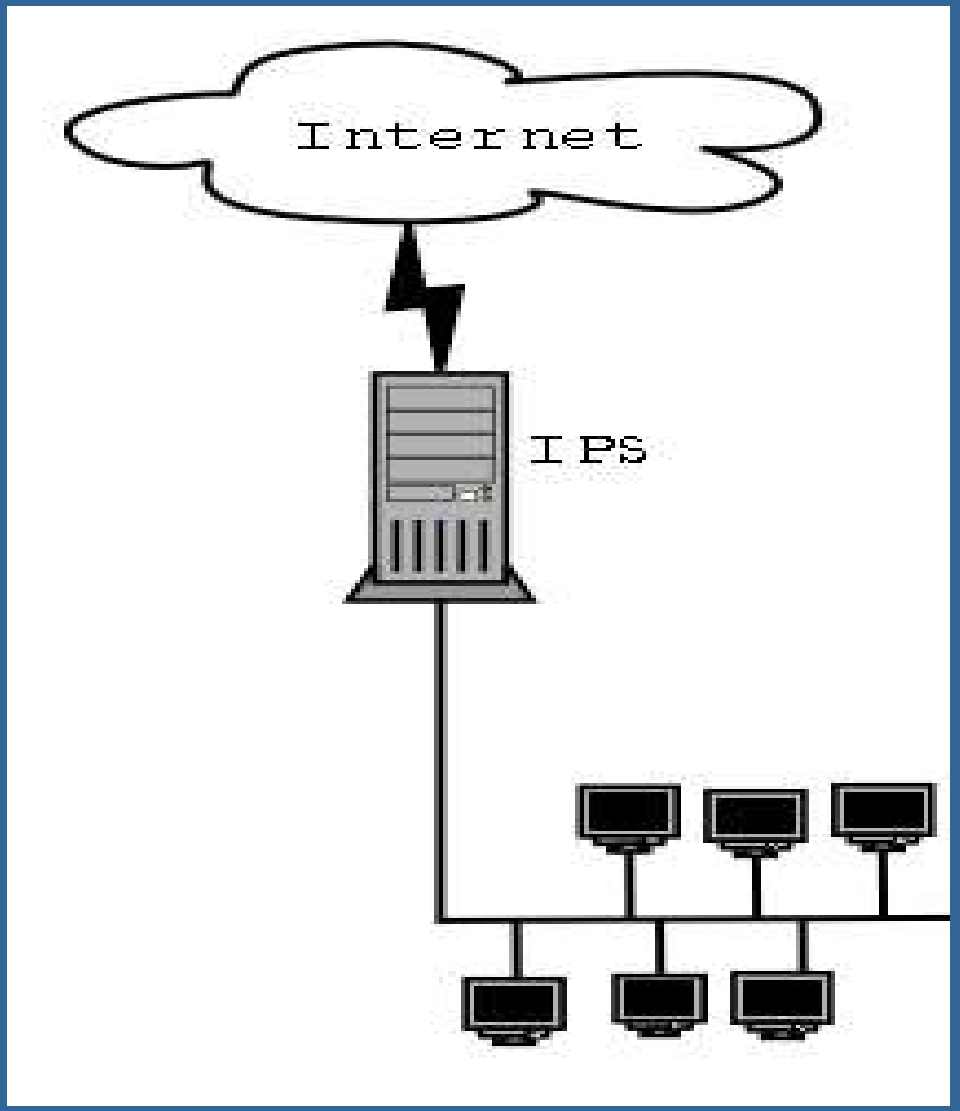
Tipos de Ataques

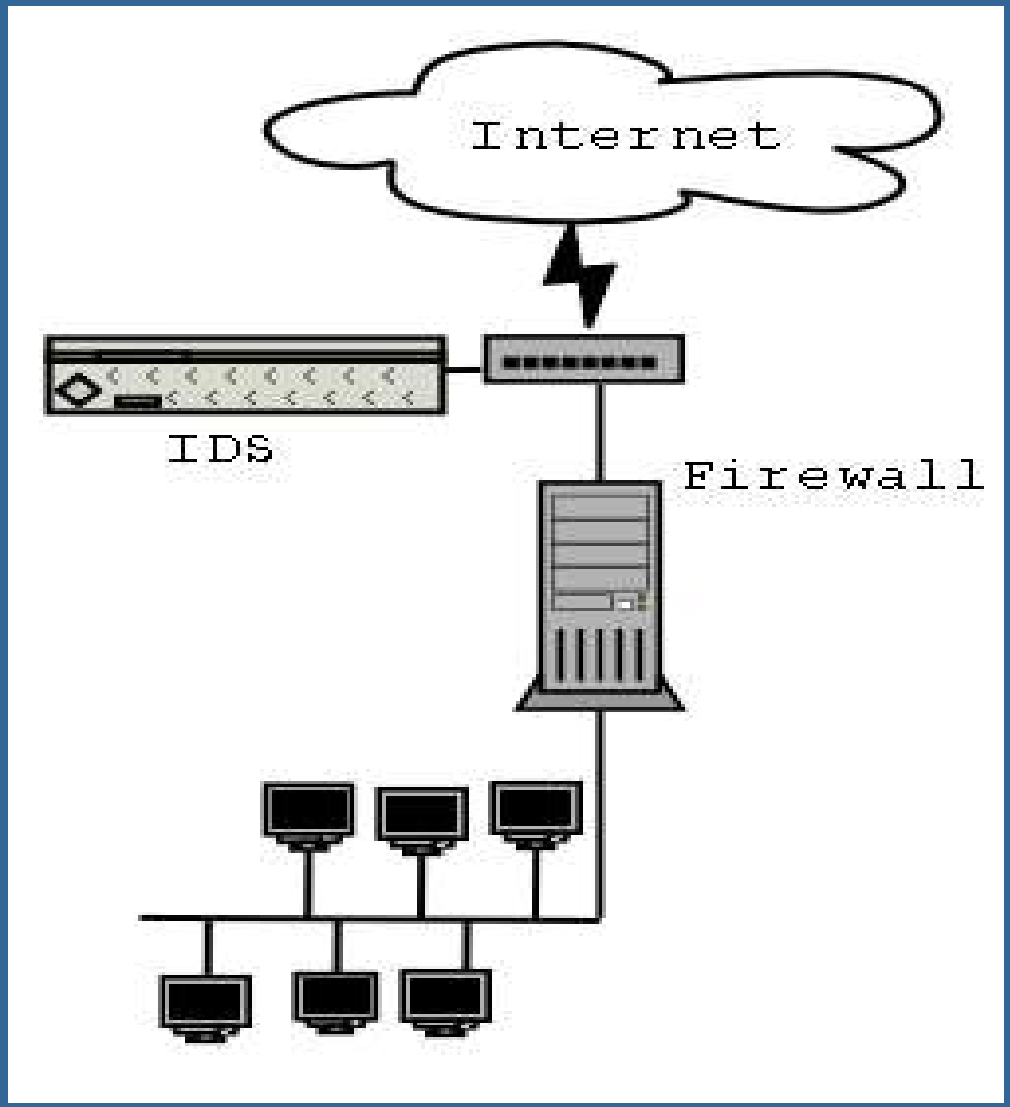
* LinkFlood
* DoS/DDoS
* Smurf
* SynFlood
* Vulnerabilidade
* Land
* Teardrop
* Exploração
* BufferOverflow
* FormatString
* SQL Injection
* Cookie Poisoning
* Cookie Tampering
* ...

Quem implementaria?



IDS x IPS





Tipos de IDS

Constitui-se de 4 grandes tipos, a saber:

- **HIDS**
- **NIDS**
- **IDS Ativo**
- **IDS Passivo**

Baseados em:

- **Assinaturas**
- **Eventos**

Assinaturas de Ataques

- **Todo ataque possui caracteristicas ÚNICAS que permitem distinguí-lo de outros ataques e principalmente do tráfego normal da rede.**

- **Tais caracteristicas são chamadas de Assinaturas de Ataques.**

Assinaturas x Regras x Profile

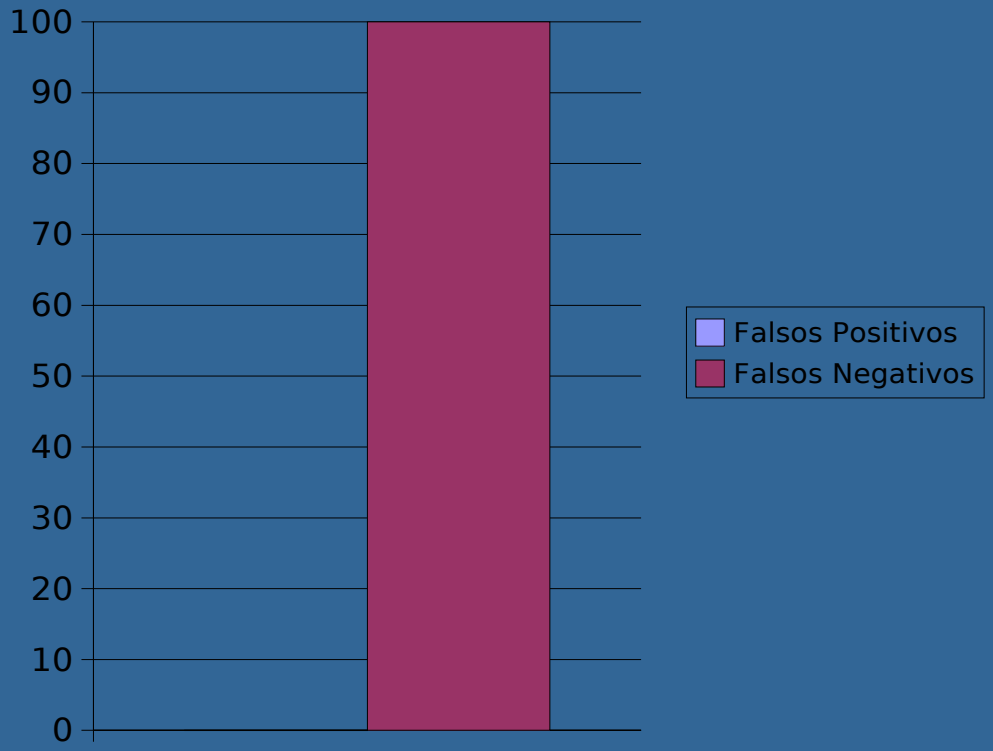
- Assinaturas normalmente sao utilizadas para designarmos apenas uma comparacao de strings (se esta string existir dentro do pacote, entao dispare um alerta)
- Chamamos de regras (rules) o uso de assinaturas como pretendemos utilizar nesta apresentacao, ou seja, envolvendo mais do que simples strings, mas sim as caracteristicas a serem observadas
- Temos ainda o conceito de profile, que seriam as caracteristicas de uso de determinado usuario, como por exemplo horarios do dia em que este utiliza seus emails.

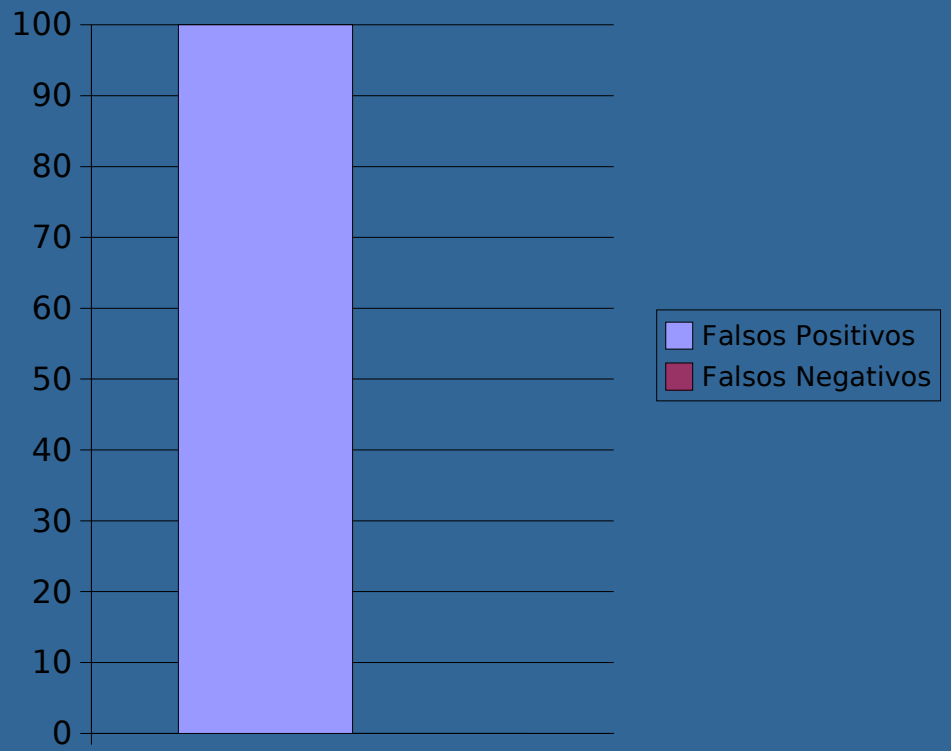
Assinaturas de Ataques

Problemas com Detecção de Intrusão

- Fragmentação;
- Criptografia (VPN);
- UUEncode/Gzip Encode;
- Velocidade dos Links;
- Assinaturas baseadas em “pattern match”, ou seja, reconhecimento de determinada string dentro de um determinado pacote;
- Consolidação e correlação;
- Mantimento de sessões;
- Técnicas de polimorfismo;
- “Falhas” em sensores;
- Falsos Positivos x Falsos Negativos x Falsas Interpretações.

Falsos Positivos x Negativos





Exposição x Vulnerabilidade

- **Vulnerabilidade**

  Existência de determinada falha de segurança que pode ser explorada, dando acesso a execução de códigos arbitrários ou informações confidenciais.

- **Exposição**

  Existência de determinada má configuração de sistema, que permite coleta de informações. Chamamos de exposição qualquer tipo de configuração de segurança não adequada.

Origem de um Ataque

Quando topamos com determinado ataque, devemos precisar sua origem. Esta pode ser:

- **Real**
- **“Spoofada”**
- **Efeito colateral**

Ataques polimorficos?

- Consiste em ataques que se auto-decodificam durante a execução no sistema alvo.
- Evitam análises do tipo “pattern-matching”
- Impossíveis de serem detectados, causam problemas em assinaturas genéricas.
- Assinaturas Genéricas: Visam detectar todo tipo de ataque que possua uma caracteristica comum, por exemplo NOPs em instruções assembly.

0 DAY x Polimorfismo

- **Impossível o uso de Técnicas utilizadas pelos Antivirus, devido a necessidade de análise em tempo real.**
- **Talvez a criação de assinaturas baseadas em serviços, porém não detectariam ataques 0 DAY.**
- **SCMorphism -> Ferramenta para testes de detectores de intrusos contra técnicas polimorficas.**

Mecanismos de Ataque

Antes de escrevermos uma assinatura de ataque, precisamos de alguns conhecimentos basicos sobre o que desejamos:

- **Estímulo ou resposta?**
- **Serviço alvo**
- **Existe vulnerabilidade ou exposição pública ao serviço?**
- **Tráfego normal, exploração, DoS ou reconhecimento?**
- **Dispositivos auxiliares de informação (Firewalls, roteadores, sistemas)**

Severidade

- As assinaturas de um ataque podem ser mais específicas ou genéricas.

- A severidade de um ataque pode facilmente ser vista através de simples fatos:
  - O scan foi lançado contra toda a rede ou contra uma máquina específica?
  - A máquina específica a qual o scan foi lançado possuia o serviço requisitado ou foi feita uma busca por todos os serviços?
  - A busca específica ao recurso foi feita por uma versão específica ou tentou-se descobrir a versão?
  - Houve um resultado a despeito da busca efetuada?

Exemplo 1: DNS

[**] [1:256:5] DNS named authors attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/04-02:40:31.425495 192.168.0.11:32848 -> 192.168.0.10:53
UDP TTL:64 TOS:0x0 ID:2746 IpLen:20 DgmLen:58 DF
Len: 30

SEVERIDADE = (CRITICIDADE + LETALIDADE) – (DEFESAS SISTEMA + DEFESAS REDE)

CRITICIDADE = 5 (DNS)
LETALIDADE = 2 (LEVANTAMENTO DE INFOS)
DEFESAS SISTEMA = 4 (SO ATUALIZADO E MODERNO)
DEFESAS REDE = 1 (FIREWALL DEIXOU PASSAR)

- SE FOI SUCEDIDO, SOMAR 1 A SEVERIDADE TOTAL

Exemplo 1: DNS

[**] [1:1616:6] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/04-02:36:53.489089 192.168.0.10:1041 -> 192.168.0.11:53
UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:58 DF
Len: 30

[**] [1:255:12] DNS zone transfer TCP [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/04-11:32:58.584907 192.168.0.10:1052 -> 192.168.0.1:53
TCP TTL:64 TOS:0x2 ID:53804 IpLen:20 DgmLen:72 DF
AP Seq: 0x4F5E4B1 Ack: 0xE4D61682 Win: 0x16D0 TcpLen: 20

[**] [1:624:6] SCAN SYN FIN [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/04-03:37:43.085849 192.168.0.11:10004 -> 192.168.0.10:53
TCP TTL:255 TOS:0x0 ID:2304 IpLen:20 DgmLen:40
******SF Seq: 0x56F2CD88 Ack: 0x0 Win: 0x1000 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS198]

Protegendo o IDS

- Firewall Connection Less (filtro de pacotes).
- Firewall Proxy (Firewall de aplicação).
- Firewall Gateway.
- Firewall com Estado (Stateful).

OBS: Um equipamento pode possuir diversas destas características.

Protegendo o IDS

- Cabo half-duplex
- Conexao cross-over com servidor de consolidacao
- “Rede de IDS”

Stealth Scan

- Tipo de ataque que visa passar por Firewalls que não sejam Stateful.
- Utiliza-se de algumas características inerentes a conexões TCP:

Flags:

SYN – Iniciar conexão.

FIN – Terminar conexão normalmente.

URG – Campo URG possui dados.

ACK – Aceitar conexão.

PSH – Dar prioridade a este pacote.

RST – Terminar conexão caso haja erros.

O ataque Stealth simplesmente envia um pacote com o Flag ACK ativo sem haver enviado um pacote com o flag SYN (iniciar conexão).

Arquitetura IDS

- Modo de captura de pacotes (libpcap)
- Filtragem de pacotes (nao necessariamente suportado pelo IDS)
- Decodificacao de pacotes (aqui pode existir normalizacao de pacotes)
- Armazenamento (muito preocupante)
- Remontagem de fragmentos
- Remontagem de stream (dados)
- Stateful Inspection

Fragmentos x IDS

- **Fragmentos que sobrescrevem partes de um fragmento anterior**
- **Fragmentos muito grandes ou muito pequenos (tiny)**
- **Fragmentos com offsets totalmente errados**

Stream x IDS

- Se os pacotes nao forem montados corretamente (na ordem de chegada), torna-se impossivel detectar ataques nos mesmos
- Cuidados como pAacotes do tipo SYN ou FIN com dados (isto existe na RFC)

Exemplo 2: Servidor Web

[**] [1:1260:11] WEB-MISC long basic authorization string [**]
[Classification: Attempted Denial of Service] [Priority: 2]
11/04-02:36:16.731399 192.168.0.10:1065 -> 192.168.0.11:80
TCP TTL:64 TOS:0x0 ID:29020 IpLen:20 DgmLen:1500 DF
A* Seq: 0x8AA2EA40 Ack: 0x507F49DB Win: 0x16D0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 629685 620484

[**] [1:498:6] ATTACK-RESPONSES id check returned root [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/04-01:34:25.713398 192.168.0.10:61200 -> 192.168.0.11:32790
TCP TTL:64 TOS:0x0 ID:41757 IpLen:20 DgmLen:91 DF
AP Seq: 0x791C8E50 Ack: 0x3FA7E3AD Win: 0x16A0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 601366 592250

Kernel Security

Ambiente Chroot

Modulos de Segurança (chamadas de sistema monitoradas, por exemplo ptrace) – systrace

Securelevels, proteçao contra carregamento de modulos, etc

Registradores

A arquitetura aqui apresentada é Intel-X86. Todos os registradores da plataforma intel, suportam 32 bits dos quais podem ser divididos em subseções de 16 e 8 bits.

32 bits	16 bits	8 bits	8 bits
EAX	AX	AH	AL
EBX	BX	BH	BL
ECX	CX	CH	CL
EDX	DX	DH	DL

Shellcodes x Assembly

- "Inline Assembly" - Sintaxe AT&T.
- Nome dos registradores são precedidos pelo simbolo "%"

MOV - Esta instrução nos permite mover um valor em um registrador.
mov %0x8, %al

PUSH - Empura dados sobre a Stack.
push $0x0

POP - Ao contrario de push, pop retira dados da stack
pop %edx

INT - Chamada de interrupção. 0x80 passa o controle ao kernel
int $0x80

Falhas

Os sistemas operacionais possuem atualmente seu proprio espaço de endereco virtual, gerenciado pela MMU (memory management unit).

Estes espaços contem as seguintes seçoes:

.data -> Usado por constantes que nao sao modificadas em tempo de execuçao

.bss -> Usado como reserva de espaço para o programa com variaveis utilizadas em tempo de execuçao

.text -> Espaço READ-ONLY que possui as instruçoes em assembly do programa

O sistema operacional tambem prove a STACK e um espaço de memoria livre (HEAP) que podem ser modificados em tempo de execuçao.
A STACK e utilizada para armazenar variaveis locais e argumentos de funçao.
A HEAP e utilizada para armazenar variaveis alocadas dinamicamente.

Polimorfismo e Virus

Tecnicas de identificacao de codigo virotico

Contaminaçao de binarios em memoria – Por que e possivel

Diferenciando Virus e Worms – Detectando Ambos

Definiçoes e Surgimento do Polimorfismo e tecnicas
de detecçao baseadas em estatisticas

Polimorfismo

----

call decryptor

----

shellcode

----

decryptor

----

jmp shellcode

Polimorfismo

O decoder sera responsavel por realizar o processo inverso ao utilizado para codificar o shellcode.

Este processo pode ser, por exemplo:

- **ADD**
- **SUB**
- **XOR**
- **SHIFT**
- **Criptografia (DES)**

Polimorfismo

Quando executamos uma chamada (call em assembly), o proximo endereco (no nosso caso, o endereco do shellcode) sera colocado na stack (push). Entao, o decoder pode pegar este endereco facilmente apenas executando um pop para qualquer registrador de uso geral. Apos isto, basta que o decoder manipule os bytes do shellcode e de um jmp para o endereco deste.

Polimorfismo

Precisamos apenas concatenar o shellcode ao final do nosso decoder.

Polimorfismo - SCMorphism

Software que automatiza o processo de geracao de shellcodes polimorficos, atuando com os seguintes problemas e dificuldades existentes:

- Evit BADChars (like \r \t \0)

- Generate Alphanumeric Polymorphic Codes

- Eliminate constants in decoders

- Decoders build

- Mantain decoders for many platforms

- Insert “do nothing” instructions in the generated decoder/shellcode

- Optimize the decoder len

Polimorfismo - SCMorphism

O SCMorphism nao apenas codifica o shellcode original com um valor randomico e o coloca junto com o decoder. Ele tambem tem a capacidade de gerar randomicamente um decoder.

SCMorphism pode criar XOR/ADD/SUB/ByteShift decoders randomicos.

Os decoders sao cortados em pecas (ideia original de Zillion@safemode.org, usada na ferramenta s-poly).

Com estas pecas a ferramenta consegue gerar decoders randomicos apenas manipulando estas partes e introduzindo instrucoes DO-Nothing.

Polimorfismo - SCMorphism

Com decoders randomicamente gerados, torna-se impossivel a geracao de assinaturas de IDS para a ferramenta (ou o administrador teria milhares de falsos positivos).

Quando voce chama a funcao de codificacao, ela gera um numero randomico para ser usado nos calculos (tambem pode ser escolhido pelo usuario).

Polimorfismo - SCMorphism

O usuario pode escolher caracteres proibidos (passagem por filtros de aplicacoes), sendo que no entanto o sistema utiliza \0, \r e \t por padrao.

Com uma matriz de instrucoes do nothing, a ferramenta consegue substituir as operacoes NOOP normais (como o ADMutate) e mais ainda, incluir randomicamente operacoes deste tipo no meio do decoder ou do proprio shellcode, conforme o tamanho escolhido pelo usuario.

LEMBRETE

"Segurança, segundo nossa filosofia e um processo complexo que vai desde uma boa interface com o usuario, ate uma boa administraçao e atualizaçao dos servidores, aliados a uma programaçao em constante teste e correçao.

- **INDEPENDE DO USUARIO**

- **EMBORA SEJA REGULAMENTADA, DEVE SER IMPOSTA**

Referências

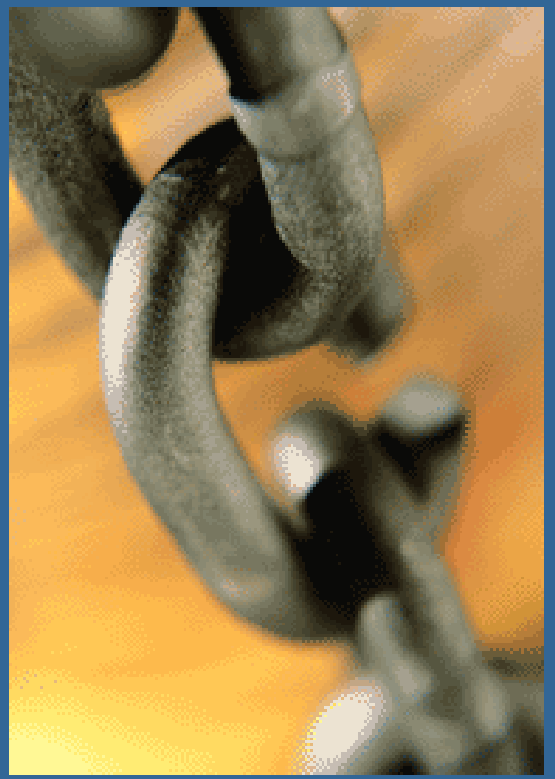
- SCMorphism
  http://www.bsdaemon.org
- SANS
  http://www.sans.org
- Security Focus
  http://www.securityfocus.com
- NFR Security
  http://www.nfr.com
- ISS
  http://www.iss.net

Agradecimentos

- **Eventos, Workshops, Seminários, Palestras, RoadShows, Conversas**

- **Comunicação**

- **Elo mais fraco da segurança da informação**



FIM! Será mesmo?

DÚVIDAS ?

Rodrigo Rubira Branco
rodrigo@firewalls.com.br