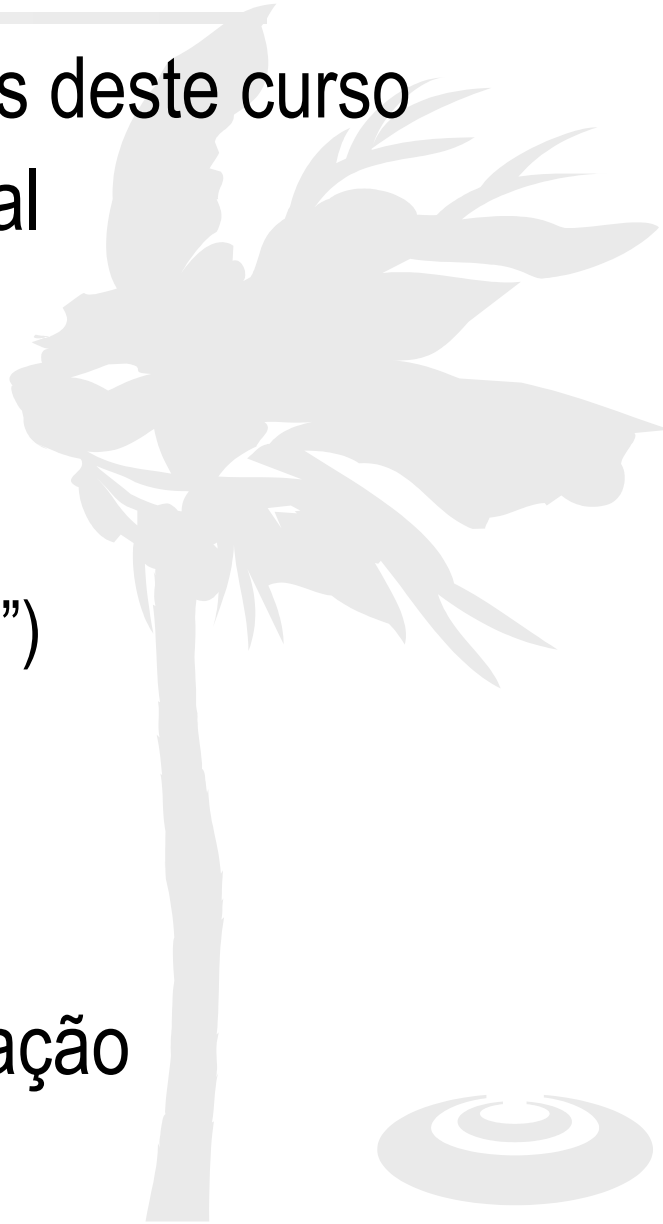
A Evolução dos Firewalls e da Segurança Perimetral

Marco "Kiko" Carnut, CISSP <kiko@tempest.com.br>
Cristiano Lincoln Mattos, CISSP, SSCP <lincoln@tempest.com.br>
Evandro Curvelo Hora, M.Sc, <evandro@tempest.com.br>
V Simpósio Segurança em Informática – Novembro/2003 – CTA/ITA, SJC

Agenda 1/2

- Prólogo: Por que e quais os objetivos deste curso
- Introdução: O Problema Fundamental
- Conceituação geral de Firewall
- Filtros de Pacotes:
  - Com Inspeção de Estado (“stateful”)
  - Tradicionais ou sem estado (“stateless”)
- Tradução de endereço
  - Estática sem estado
  - Dinâmica com estado
- Proxies/Gateways em nível de aplicação



Agenda 2/2

- Bases de Regras, autenticação e logging
- “Firewall Piercing”: Furando firewalls
- Desenvolvimentos recentes
  - Bridge-firewalls, bridge-level firewalls
  - Halted firewalls
  - Firewalls vs IDSs
- Idéias, necessidades e tendências
- Tunelamento
  - Encapsulamento “recursivo”
  - Implicações no roteamento
  - A ponte para as VPNs

(Minha) Motivação

- Certa vez ouvi alguém dizer que:
  - Firewalls são um “commodity”.
    - Todo mundo já tem
    - É difícil se diferenciar
    - Efetivamente resolve o problema
  - Pode ser mas,...
    - Muita gente tem e não sabe usar
    - Muita gente não usa direito
    - Se não sabe usar, não adianta tanto ter
    - Atrapalha, se levada a extremos
    - É bem mais fácil de burlar (para o bem ou para o mal) do que se imagina

Objetivos

- Recapitular a evolução recente dos conceitos e técnicas de bloqueio seletivo de tráfego, especialmente na proteção perimetral
- Mostrar que a área continua florescendo e recheada de inovações, idéias intrigantes e implicações filosóficas
- Experimentar uma nova didática: falar sobre filtros com estado *antes* de filtros sem estado
  - Tese: fica muito mais claro o que se perde
- Preencher algumas lacunas na formação do engenheiro de rede típico

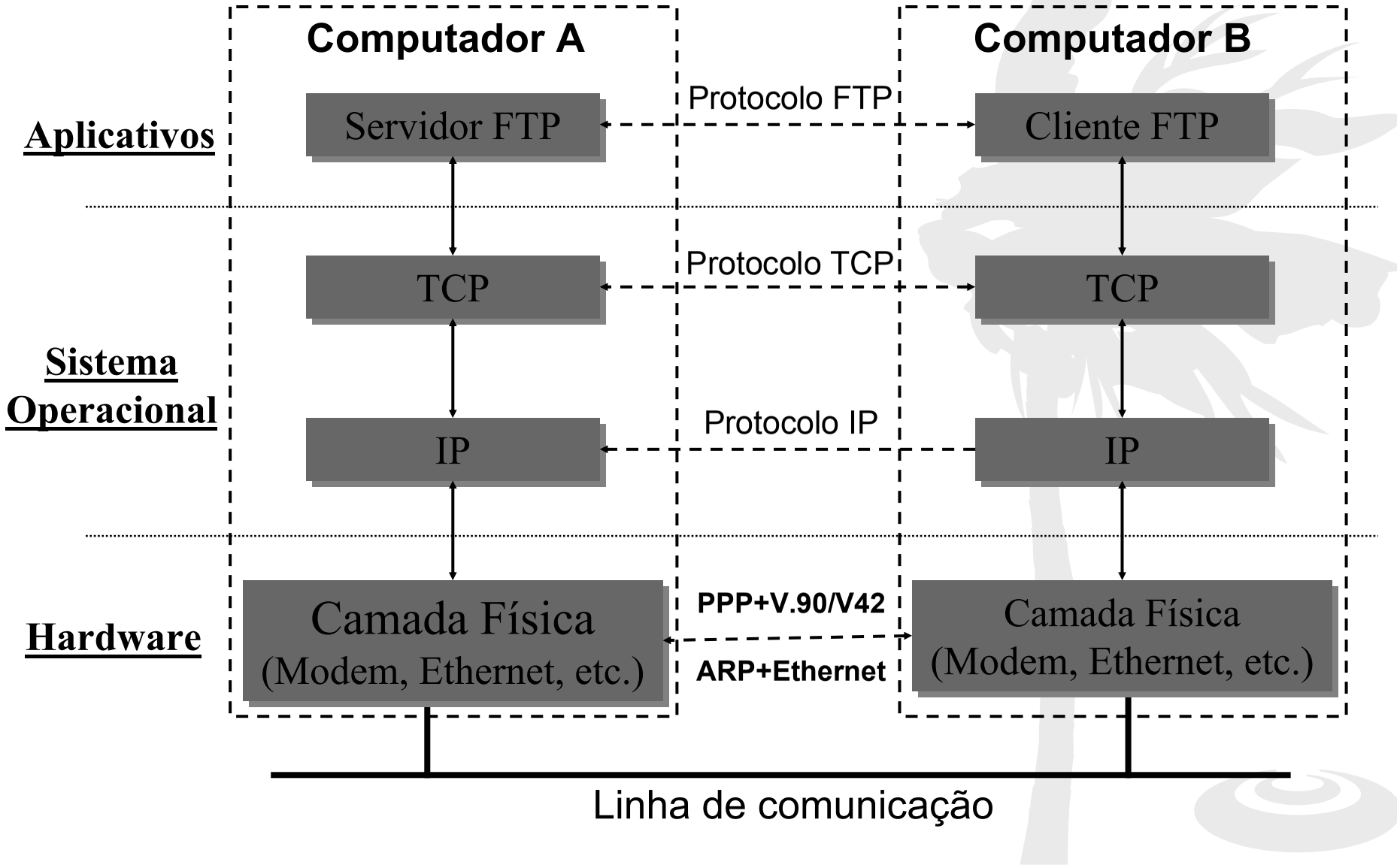
Parte I: Reconsiderando os Fundamentos

Uma brevíssima visão geral

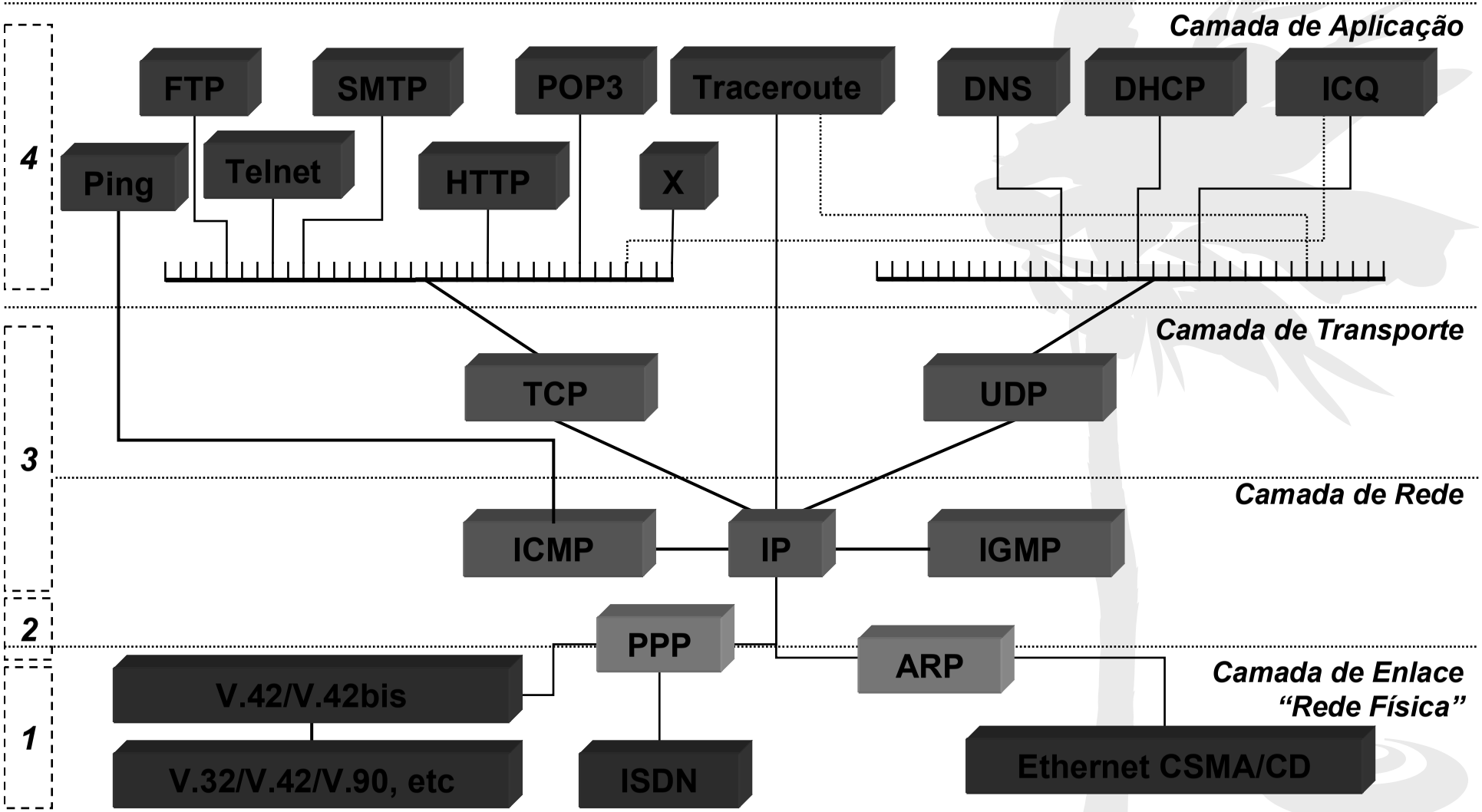
Bloquear por que?

- Porque os serviços de rede têm vulnerabilidades que o administrador de rede não tem como consertar
  - Seja porque ele não tem o código fonte da aplicação...
  - Seja porque ele não sabe consertar
  - Seja porque o conserto pode ser muito difícil (protocolos ou serviços intrinsecamente inseguros por *design*)
  - Seja porque o criador do programa nega veementemente que o problema existe em absoluto
- Se os engenheiros de software fizessem softwares realmente resistentes a ataques, a necessidade de bloqueio seria bem menor

Arquitetura Cliente-Servidor

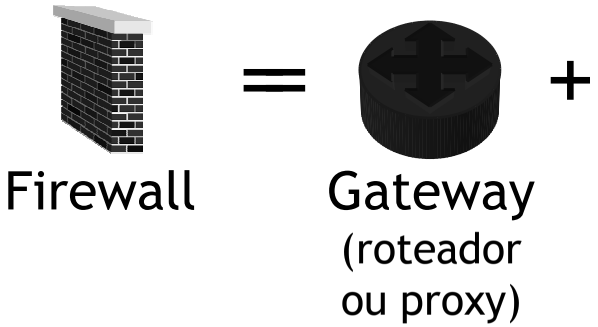


Camadas da Arquitetura de Rede



O que é um firewall?

- Gateway restritivo:



No.	Source	Destination	Service	Action	Track	Install On	Time	Comment
-	Trusted hosts	FW1 host	FireWall1	accept		Gateways	Any	Enable FW1 control connections [essential]
-	ftp server	local client	expected data conn	accept		Gateways	Any	Enable Response of FTP Data Connection
-	Any	Any	rip	accept		Gateways	Any	Enable RIP (Common)
-	Any	Any	domain-udp	accept		Gateways	Any	Enable Domain Name Queries (UDP) [Esse
-	Any	Any	domain-tcp	accept		Gateways	Any	Enable Domain Name Download (TCP)
-	Any	Any	rpc control	accept		Gateways	Any	Enable RPC Control
1	Any	Web_server_pool	http	accept	Short	Gateways	Any	Allow web traffic to the logical web serve Load Balance web traffic to the Web Serv
2	Local_Net Remote_Net	Any	http->URL_Filter	drop	Short	Gateways	Any	Filter against bad URLs
3	Local_Net Remote_Net	local_router remote_router	Any	drop	SnmpTrap	local_router remote_router	Any	Disallow any traffic not from the local or remote network to the routers. Send a trap if anyone trys.
4	Local_Net Remote_Net	Any	Any	accept	Short	Gateways	Any	Allow the local and remote network to access the Internet
-	FW1 Host	Any	Any	accept		Gateways	Any	Enable outgoing packets [common]
-	Any	Any	ICMP	accept		Gateways	Any	Enable ICMP [Common]
5	Any	Any	Any	drop	Long	Gateways	Any	Cleanup Rule

Base de Regras (de restrições)

- Base de regras: lista ordenada de condições
  - Condições geralmente baseadas na tupla de conexão
    - Mas muitos firewalls hoje em dia têm critérios adicionais, como data/hora, etc.
  - Resultam em ações, tipicamente: PERMITIR ou NEGAR o repasse do tráfego

Tuplas de conexão

- Todo diálogo via TCP/IP pode ser representado por pela tupla dos endereços de origem e destino:

(protocolo, endpoint de origem, endpoint de destino)

```
 1=icmp
 6=tcp
17=udp
27=rdp
  ...
```

Dependente do protocolo:

para ICMP:

endpoint = IP

para TCP e UDP:

endpoint = (IP : porta)

- Utilitários tais como o netstat mostram o estado atual dessas tuplas em uma dada pilha TCP
- Estende o conceito de “conexão” para protocolos tais como UDP, que não têm esse conceito.

Índice de Conectividade

- Definição “mais ou menos formal”:
  - Quantidade de possíveis tuplas de conexão *permitidas*
  - Por se tratar de um número muito grande, é conveniente representá-lo em forma de logaritmo base 2.
- Exemplo:
  - Qual o índice de conectividade TCP de um roteador tradicional?
  - Intuitivamente:

```
   32 bits do endereço IP de origem
   32 bits do endereço IP de destino
   16 bits da porta de origem
+  16 bits da porta de destino
-------------------------------
   96
```

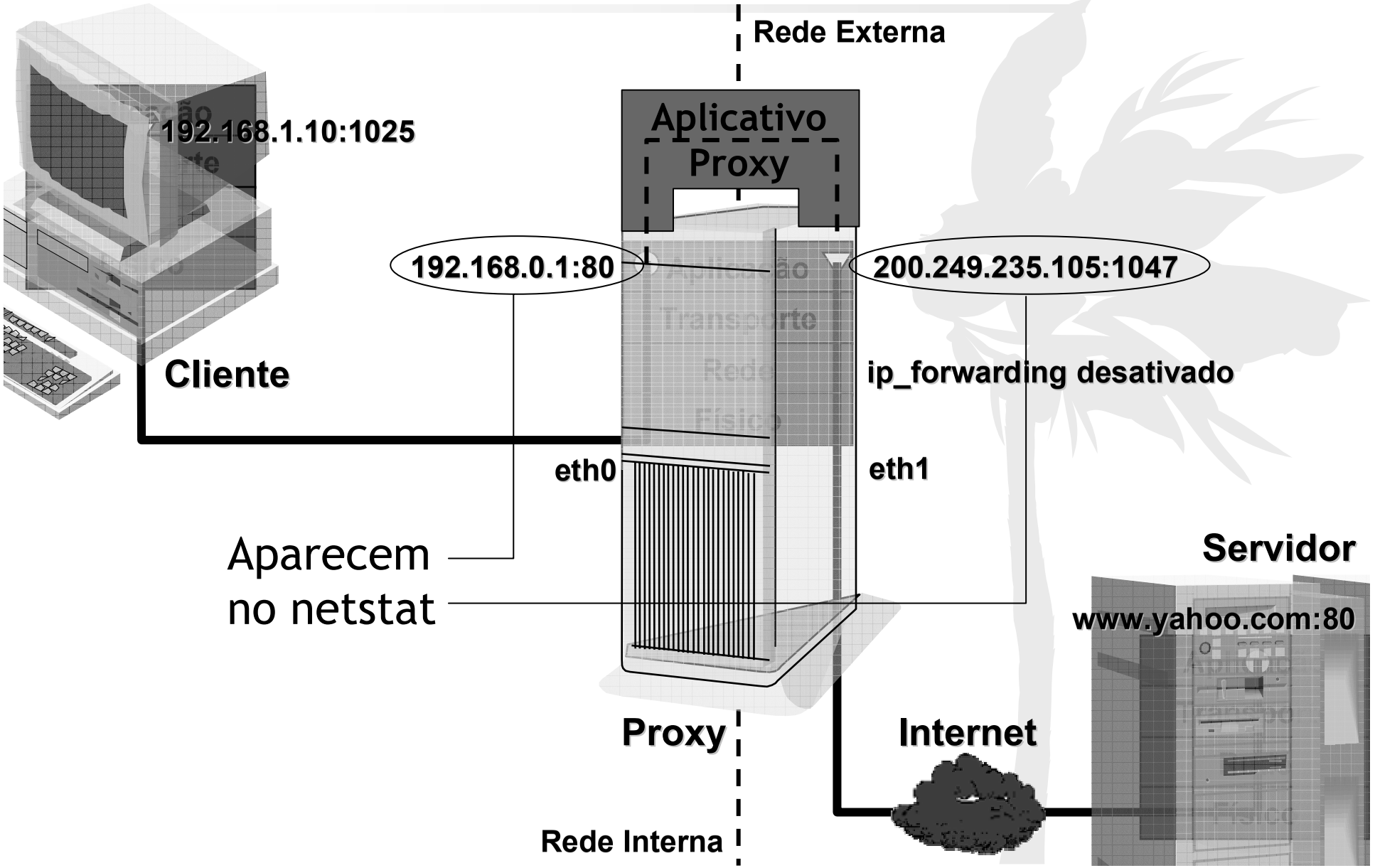
Índice de Conectividade

- Na prática, um pouco mais complicado de calcular
  - Há vários protocolos (256 possíveis), cada um com seus próprios conceitos de “endpoint”
  - Para TCP, UDP e ICMP segundo a definição anterior, o índice seria: 97.000000000167952
  - As bases de regras podem conter outros critérios que não somente comparações baseadas nos endpoints de origem e destino.
- Em outras palavras, mede a cardinalidade do conjunto de tuplas permitidas
- Conceito de “permitido” pode ser em relação apenas à base de regras ou à base de regras + roteamento.

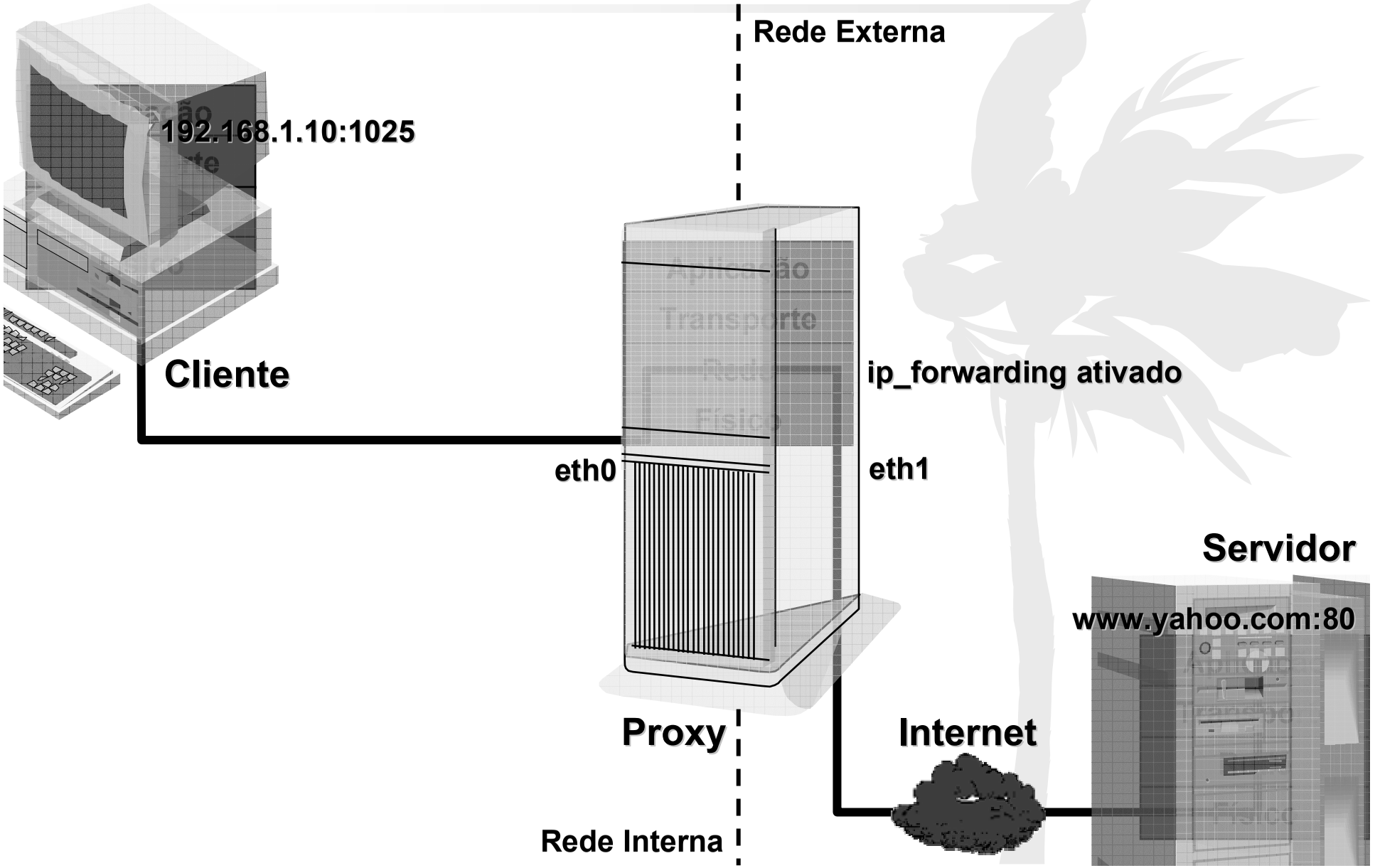
Objetivo de uma Firewall

- Permitir conectividade seletiva, com índice de conectividade menor que o índice máximo
- Prover controle preciso e granular do conjunto de conectividade → política de conectividade

Proxies (Intermediários)



Filtros de Pacotes



Stateless vs Stateful

- O que significa “sem estado” (stateless)?
  - Cada pacote, interação, transação, etc, é realizada de forma inteiramente independente das anteriores e futuras. Nenhuma lembrança é retida.
  - Particularizando para o nosso caso: não há tabelas em função do tráfego
- O que significa “com estado” (stateful)?
  - Existe alguma tabela ou cache (normalmente armazenada em memória) que lembre interações anteriores e que poderá influir em interações futuras, como parte das condições de aceitação/rejeição.

Parte II: Filtros de Pacotes com Inspeção de Estado

Ressalva

- Antes que os puristas/detalhistas me fritem... :)
- O modelo apresentado no aqui é uma generalização didática um tanto simplificada
- Apesar de inspirado nas implementações, não é idêntico a nenhuma delas:
  - Os campos exatos da tabela variam de implementação para implementação. Por exemplo, nem todas tem os campos “expected return”; alguns implementam a mesma semântica diretamente no Algoritmo de Inspeção de Estado
  - Cada implementação tem seus pormenores

A Tabela de Conexões

- Cada entrada na tabela consiste em uma versão estendida da tupla de conexões:

(proto, osep, odep, ersep, erdep, timer)

- proto: número do protocolo (1=icmp, 6=tcp, ...)
- osep: original source endpoint:
  - endpoint de origem original do pacote
- odep: original destination endpoint:
  - endpoint de destino original do pacote
- ersep: expected return source endpoint:
  - endpoint de origem esperado nos pacotes de retorno
- erdep: expected return destination endpoint:
  - endpoint de origem esperado nos pacotes de retorno
- timer: tempo em segundos para auto-remoção desta entrada

Expiração automática

- Cada entrada na tabela tem um campo chamado “timer” que é decrementado periodicamente
- Quando chega a zero, aquela entrada da tabela é automaticamente removida
- Por performance, na prática costuma ser implementado assim:
  - o campo em si armazena a data e hora da futura expiração
  - é indexado em ordem crescente, usando uma estrutura de dados tipo *heap* (onde é barato determinar o próximo-maior)
  - De tempos em tempos (2 seg, tipicamente), o próximo-maior é, menor que a data/hora atual, a entrada correspondente é removida da tabela

Algoritmo de Inspeção de Estado

- Regra do batimento direto (forward match):
  - Se alguma entrada na Tabela de Conexões tem endpoints de origem e destino originais exatamente iguais aos do pacote atual, PERMITA sua passagem.
- Regra do batimento reverso (reverse match):
  - Se alguma entrada na Tabela de Conexões tem endpoints esperados de retorno de origem e destino exatamente iguais aos do pacote atual, PERMITA sua passagem.
- Caso contrário, efetue a Avaliação da Base de Regras

Cache de Conexões Já Aprovadas

- Esse algoritmo efetivamente transforma a Tabela de Conexões em um Cache de Conexões Pré-Aceitas
- Implementado com tabelas de hash para efetuar o batimento de forma extremamente rápida
- Originalmente desenvolvido para conferir mais performance, para evitar ter de efetuar a Avaliação da Base de Regras (que é cara) para cada pacote
- Já leva diretamente em conta o fato que as “conexões” são bidirecionais, aceitando seus pacotes de retorno
  - Torna a base de regras mais simples, pois ela não precisará ter regras explícitas para fazer isso

Base de regras

- Lista ordenada de condições para aceitação/rejeição dos pacotes
- Cada implementação tem seu próprio jeito de especificar as condições
- Formato clássico de uma regra:
  - IP de origem [não] bate com IP/Netmask
  - IP de destino [não] bate com IP/Netmask
  - Porta de origem [não] bate com Lista ou Faixa de Portas
  - Porta de destino [não] bate com Lista ou Faixa de Portas
  - Ação em caso de batimento: ACEITA, DESCARTA, etc.

Base de Regras

- Cada uma tem suas particularidades
  - Alguns firewalls implementam usando outra arquitetura
    - Exemplo: No Firewall-1, cada regras é na realidade uma expressão em uma linguagem chamada INSPECT. O cliente gráfico aceita “regras” no sentido tradicional de lista de condições (mas muito mais expressivas, com objetos consistindo de grupos recursivos, etc) e outros fatores, como data/hora, autenticação, etc., e os converte em um programa nessa linguagem.
  - Em alguns firewalls, o Algoritmo de Inspeção de Estado é uma regra que deve ser inserida explicitamente.
    - Exemplo: IPTables do Linux
  - Alguns firewalls tratam logging como uma ação, enquanto outras tratam logging separadamente

Ações Clássicas

- ACEITAR (“ACCEPT”):
  - A tabela de estado é atualizada com os endpoints originais de origem e destino do pacote
  - Computa-se os endpoints de retorno esperados. Na ausência de tradução de endereço, eles nada mais são do que os endpoints originais, trocando-se origem e destino.
  - O pacote que desencadeou a avaliação é repassado normalmente.
  - Ação Terminal: quando executada, a avaliação da base de regras é dada por encerrada
- DESCARTAR (“DROP”):
  - O pacote que desencadeou a avaliação é descartado silenciosamente. Nenhuma atualização é feita na Tabela de Conexões
  - Ação Terminal: a avaliação da base de regras é dada por encerrada

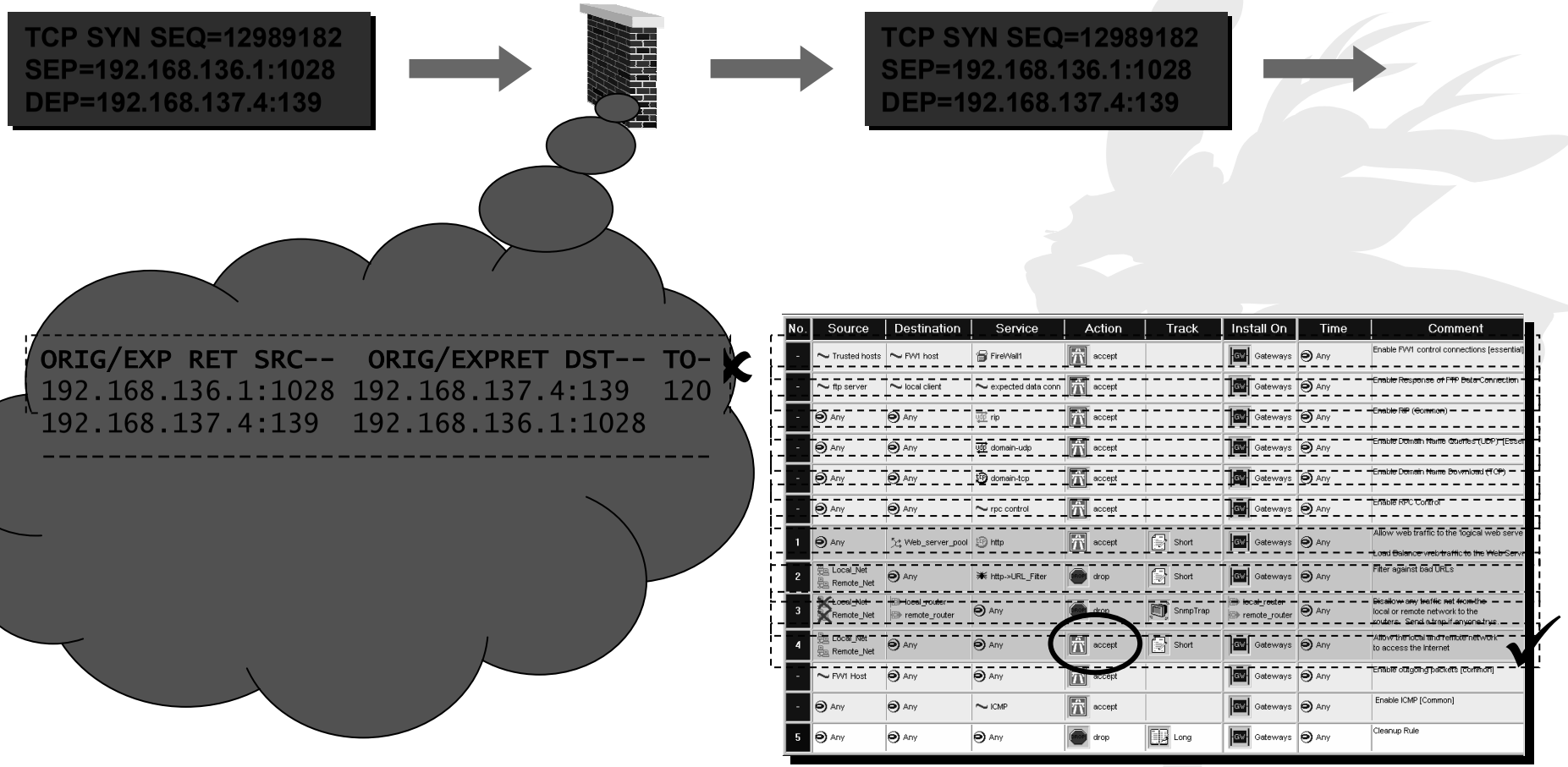
Timeouts

- Novas entradas inseridas na Tabela de Conexões recebem timeouts de acordo com algumas políticas
  - TCP: da ordem de horas
  - UDP: da ordem de dezenas de segundos
  - Primeiro pacote (“Unreplied”) normalmente recebe um tempo de vida curto (dezenas a centenas de segundos), os seguintes (“Replied/Assured”) recebem o tempo de vida “final”
  - Valores exatos configuráveis globalmente
    - No IPTables, pela linha de comando; no Firewall-1, via GUI

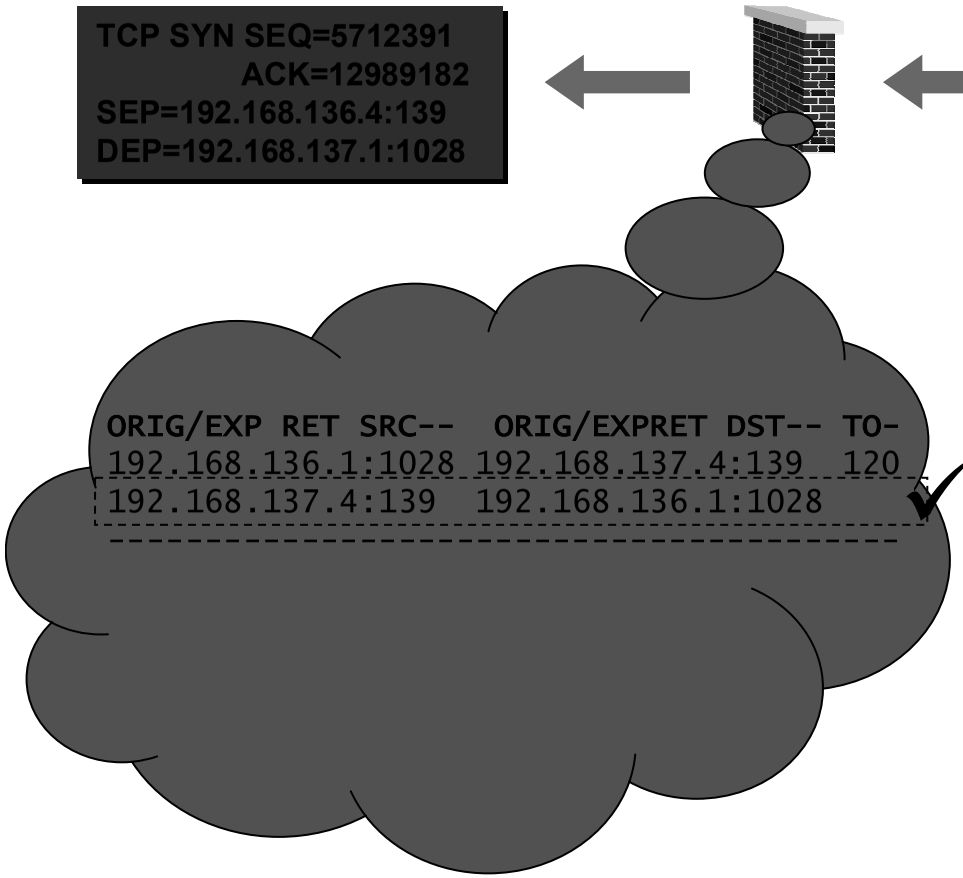
Outras Ações

- REJECT (“Rejeição Explícita”):
  - Tal como o DROP, descarta o pacote que desencadeou a avaliação da base de regras.
  - Contudo, manda uma resposta dependente do protocolo para explicitar que o pacote foi rejeitado:
    - Para TCP, manda um RST com um número de seqüência apropridado (tirado do pacote original)
    - Para UDP, manda uma mensagem ICMP Port Unreachable ou ICMP Administratively Prohibited
- LOG
  - Alguns firewalls consideram logging como uma ação
  - Registra a tupla e outras informações do pacote atual em um arquivo de log
  - Ação não-terminal: a avaliação da base de regras prossegue

Inspeção com Estado: Aceitação

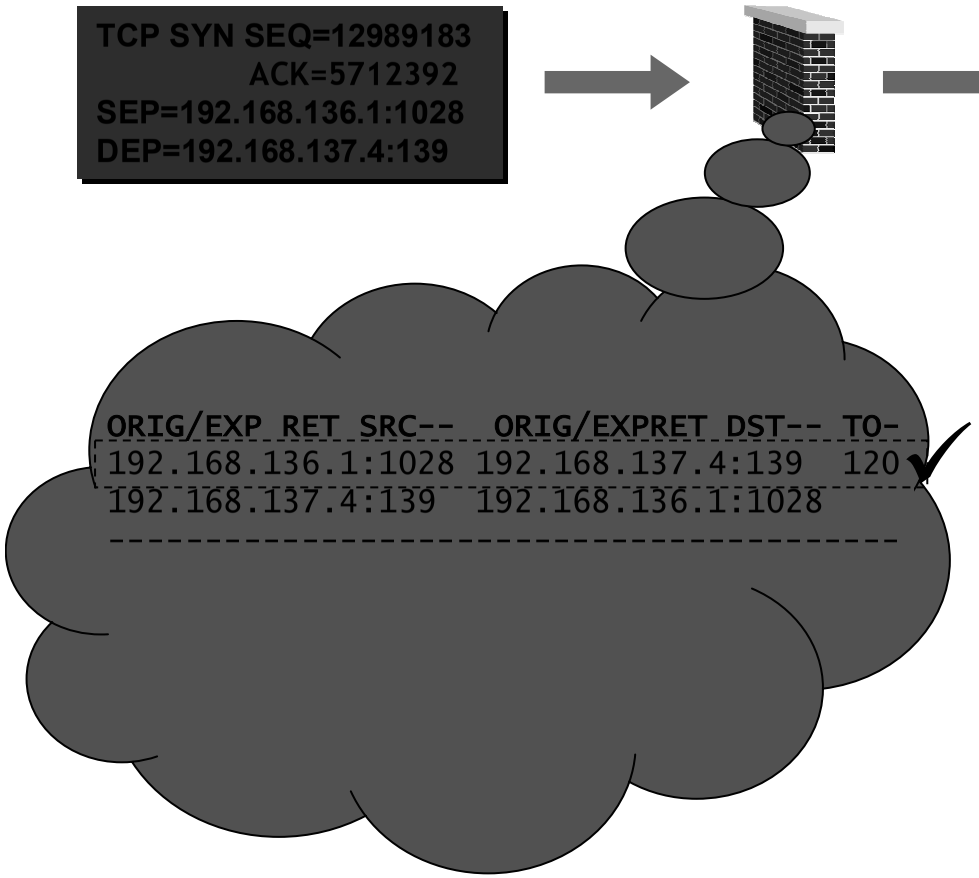


Insp. Estado: Aceitação Reversa



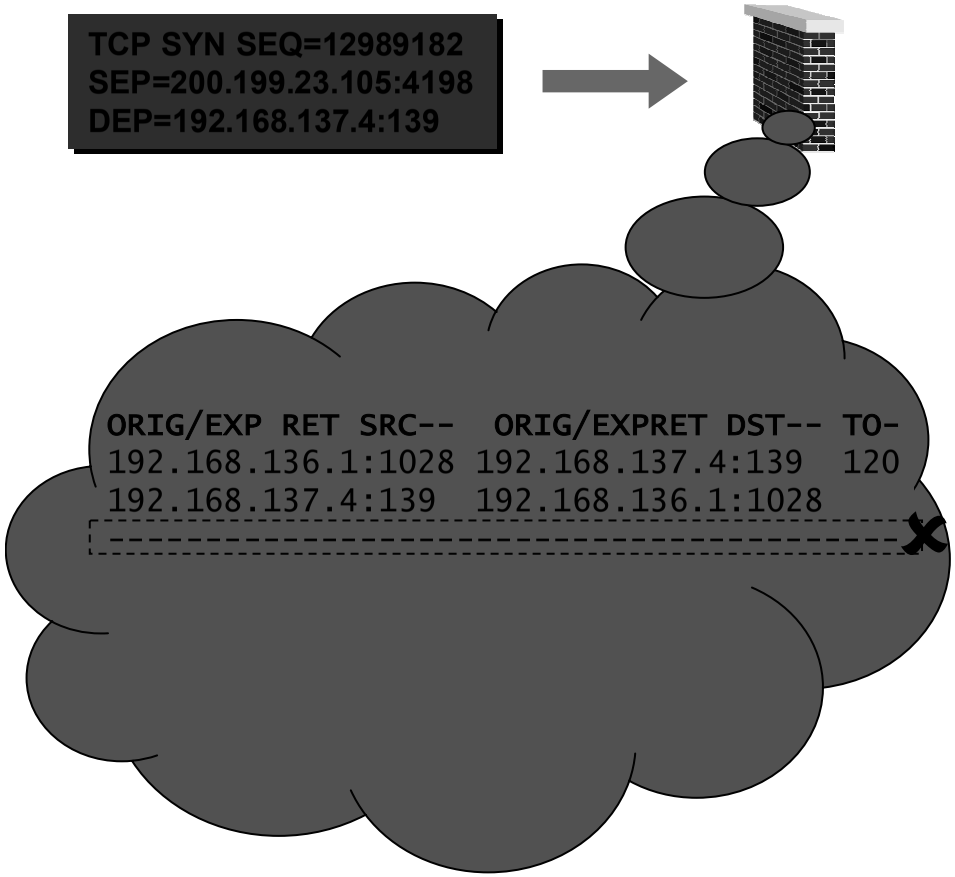
No.	Source	Destination	Service	Action	Track	Install On	Time	Comment
-	Trusted hosts	FW1 host	FireWall1	accept		Gateways	Any	Enable FW1 control connections [essential]
-	ftp server	local client	expected data conn	accept		Gateways	Any	Enable Response of FTP Data Connection
-	Any	Any	rip	accept		Gateways	Any	Enable RIP (Common)
-	Any	Any	domain-udp	accept		Gateways	Any	Enable Domain Name Queries (UDP) [Esse
-	Any	Any	domain-tcp	accept		Gateways	Any	Enable Domain Name Download (TCP)
-	Any	Any	rpc control	accept		Gateways	Any	Enable RPC Control
1	Any	Web_server_pool	http	accept	Short	Gateways	Any	Allow web traffic to the logical web serve Load Balance web traffic to the Web Serv
2	Local_Net Remote_Net	Any	http->URL_Filter	drop	Short	Gateways	Any	Filter against bad URLs
3	Local_Net Remote_Net	local_router remote_router	Any	drop	SnmpTrap	local_router remote_router	Any	Disallow any traffic not from the local or remote network to the routers. Send a trap if anyone trys.
4	Local_Net Remote_Net	Any	Any	accept	Short	Gateways	Any	Allow the local and remote network to access the Internet
-	FW1 Host	Any	Any	accept		Gateways	Any	Enable outgoing packets [common]
-	Any	Any	ICMP	accept		Gateways	Any	Enable ICMP [Common]
5	Any	Any	Any	drop	Long	Gateways	Any	Cleanup Rule

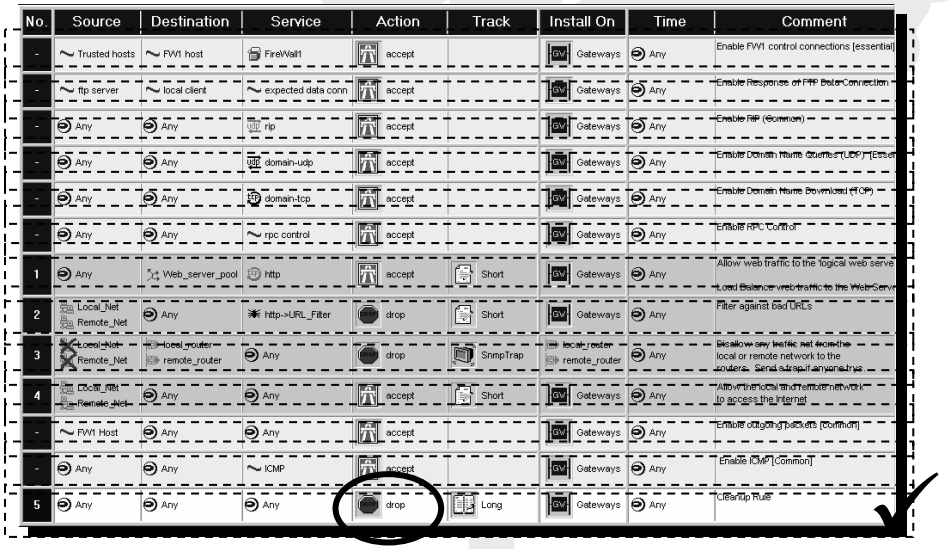
Insp. Estado: Aceitação Direta



No.	Source	Destination	Service	Action	Track	Install On	Time	Comment
-	Trusted hosts	FW1 host	FireWall1	accept		Gateways	Any	Enable FW1 control connections [essential]
-	ftp server	local client	expected data conn	accept		Gateways	Any	Enable Response of FTP Data Connection
-	Any	Any	rip	accept		Gateways	Any	Enable RIP (Common)
-	Any	Any	domain-udp	accept		Gateways	Any	Enable Domain Name Queries (UDP) [Esse
-	Any	Any	domain-tcp	accept		Gateways	Any	Enable Domain Name Download (TCP)
-	Any	Any	rpc control	accept		Gateways	Any	Enable RPC Control
1	Any	Web_server_pool	http	accept	Short	Gateways	Any	Allow web traffic to the logical web serve Load Balance web traffic to the Web Serv
2	Local_Net Remote_Net	Any	http->URL_Filter	drop	Short	Gateways	Any	Filter against bad URLs
3	Local_Net Remote_Net	local_router remote_router	Any	drop	SnmpTrap	local_router remote_router	Any	Disallow any traffic not from the local or remote network to the routers. Send a trap if anyone trys.
4	Local_Net Remote_Net	Any	Any	accept	Short	Gateways	Any	Allow the local and remote network to access the Internet
-	FW1 Host	Any	Any	accept		Gateways	Any	Enable outgoing packets [common]
-	Any	Any	ICMP	accept		Gateways	Any	Enable ICMP [Common]
5	Any	Any	Any	drop	Long	Gateways	Any	Cleanup Rule

Inspeção com Estado: Descarte





Sub-Bases de Regras e suas Ações

- Alguns firewalls permitem a criação de várias bases de regras além da principal
  - Normalmente referenciadas por nomes
  - Exemplo clássico: IPTables. Contém as tabelas principais (“filter”, “nat”, “mangle”) e várias “cadeias” (chains), tais como a “INPUT”, “OUTPUT”, “FORWARD”, “PREROUTING”, entre outras que podem ser criadas e nomeadas pelo próprio usuário.
- Ação JUMP: Executa uma sub-base de regras
  - Apesar do nome “JUMP”, que evoca a sintaxe do “GOTO”, na realidade trata-se mais de uma “chamada de subrotina”, pois a seqüência de chamadas de sub-bases é inserida em uma pilha
- Ação RETURN: retorna para a sub-base principal
  - Retira o ponto de retorno mais recente da pilha e prossegue a partir dele.

Avaliação da Base de Regras

- Cada regra é avaliada seqüencialmente, na ordem em que consta na Base de Regras
- Caso uma regra bata, sua ação é executada. Caso se trate de uma ação terminal, a avaliação da base de regras é dada como encerrada.
- Caso nenhuma regra bata, executa-se uma pseudo-regra chamada “política da base” ou “pseudo-regra final”
  - No IPTables, a política da base é configurável, podendo ser ACCEPT ou DROP
  - No Firewall-1, a política da base é fixa: DROP sem logging.

Término das Conexões

- Pode-se simplesmente deixar que suas entradas na tabela expirem
  - Em protocolos como UDP, que não têm término explícito, é a única opção, mesmo
- Em protocolos como TCP, que têm término explícito:
  - Observa-se os pacotes de finalização
  - A firewall segue o diagrama de estado do protocolo, retirando a entrada da tabela quando estiver convencida de que os dois lados encerraram

Instalação da Base de Regras

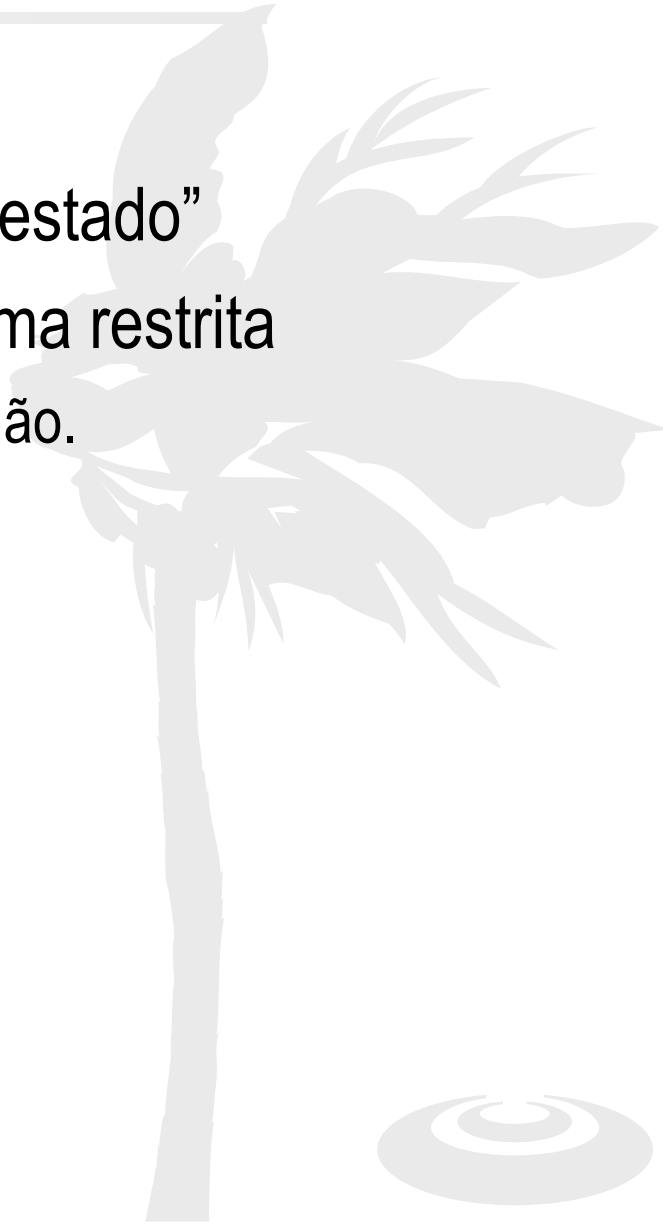
- Atômica/Com Preservação de Estado
  - A nova base de regras entre em efeito “imediatamente”
  - Preserva a tabela de estado, não quebra as conexões já estabelecida
  - Exemplos: Firewall-1, Ebtables, IPTables (possível mas pouco conveniente de fazer)
- Não-Atômica/Sem Preservação de Estado
  - Leva-se tipicamente vários segundos para a nova base de regras entrar em efeito
  - Tabelas de Conexão e demais estados são reiniciados
  - Quebra as conexões já estabelecidas
    - Irrita os usuários quando instalamos sucessivas vezes a política durante sessões de teste e depuração
  - Exemplos: Algumas versões do SonicWall (não sei se as mais novas consertam isso), alguns scripts para IPTables

Conexões TCP Ociosas

- Conexões TCP ociosas não geram tráfego
  - Podem ficar assim indefinidamente
  - É comum que uma conexão ociosa perfeitamente válida saia da tabela de estados devido à expiração automática
    - Especialmente se o timeout foi baixo
- Política de Tratamento de Conexões Ociosas
  - Alguns firewalls distinguem pacotes Não-SYN (i.e, do “meio da conexão”) na tabela de conexões
  - Nos que não distinguem, o pacote é simplesmente reavaliado na base de regras
    - Não quebra as conexões já estabelecidas
    - Ao custo de menos segurança: permite canais subversivos, à la AckCMD
  - Os que distinguem, podem:
    - Rejeitar os pacotes: quebra as conexões;
    - Aceita os pacotes: idem acima;
    - Desafia a conexão (Firewall-1): tira o conteúdo e muda o número de seqüência; se o outro lado retransmitir, a conexão é declarada como válida e reinserida na Tabela de Conexões.

Tratamento do ICMP

- Tratamento inconsistente
  - Alguns firewalls tratam de forma “sem estado”
  - Alguns tratam com estado, mas de forma restrita
    - A tabela de estados ora reconhece, ora não.



Logging

- Só costuma ser gerado por uma ação terminal
  - Tabela de Conexões não geram log
  - Só são registrados, portanto, os pacotes:
    - Que inserem ou reinserem as conexões na Tabela:
      - normalmente, o pacote inicial; mas o caso das conexões ociosas é uma exceção
    - Que sejam rejeitados (pois sempre forçam reavaliação da base)
  - Tratamento inconsistente dos ICMPs torna-se bem evidente nos logs
  - Em alguns Firewalls, registrar no log é uma ação não-terminal
- Referência às regras
  - Regras costumam ser numeradas de alguma forma que possibilite referênica
  - Para fins de depuração, é útil que cada entrada no log registre o número ou referência da regra que a causou

Logging

- Subsistema de logging deve ser eficiente
  - Deveria ser possível registrar *tudo* em log, para fins de depuração, pós-processamento, estatísticas, IDS, etc.
  - Subsistema separado que registre sem empatar o repasse dos pacotes, que é muito mais crítico
    - Deve ser capaz de falhar graciosamente, descartando entradas se o volume de registros de log excede sua capacidade de atendimento... e notificar o administrador sobre o que houve!
  - Alguns firewalls (Firewall-1) geram, além do log cru, índices para busca rápida
- Syslog clássico do Unix não satisfaz esses requisitos
  - Além de ter suas próprias vulnerabilidades

Logging

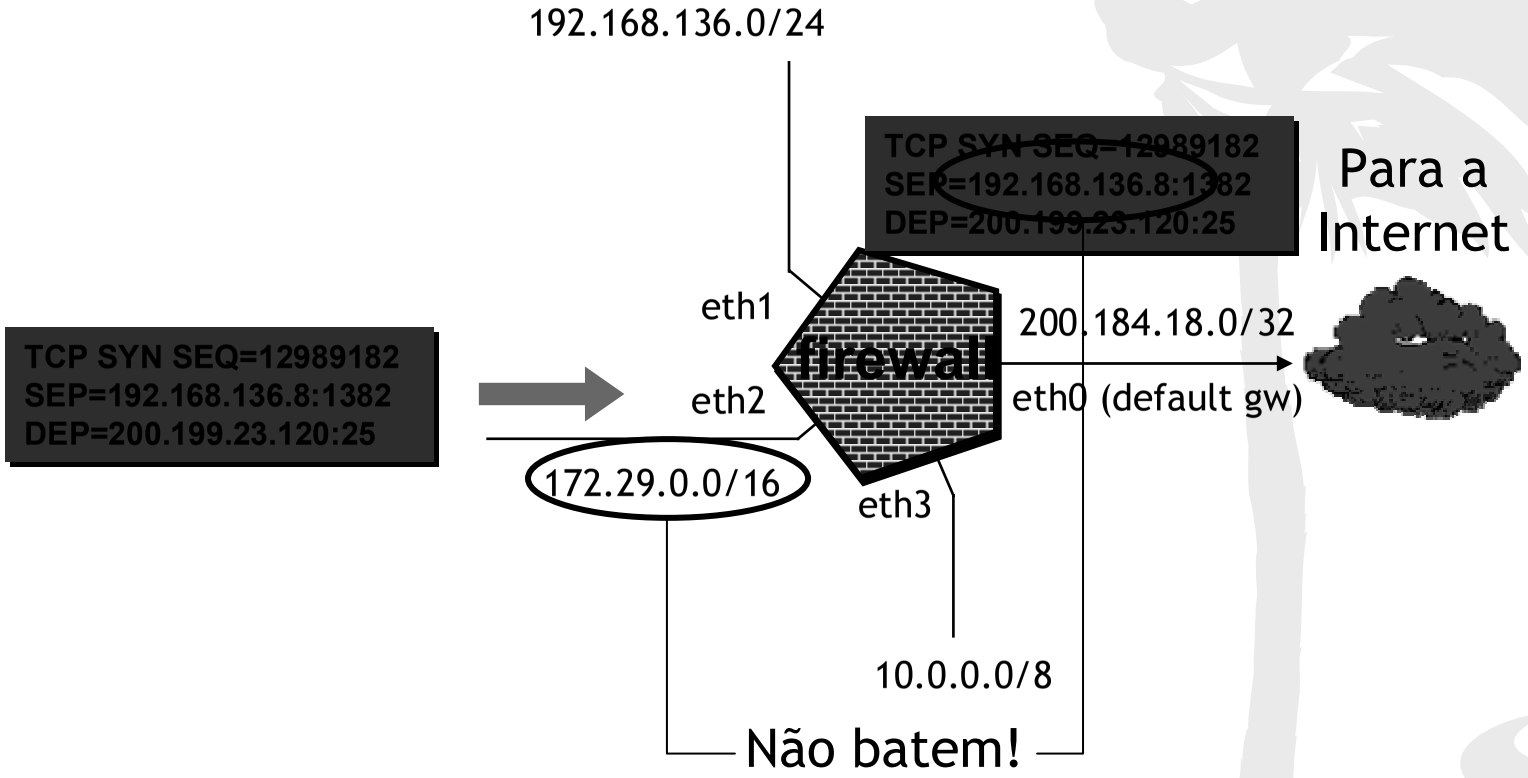
- Gerenciamento de excesso:
  - descarte automático de conexões “parecidas” dentro de um intervalo de tempo
  - Útil para evitar que pings de teste e de NMSs tornem o log ainda maior
  - Mas confunde os novatos: nada mais confuso do que você mandar seu ping de teste ele não aparecer no log
- Contenção de Tamanho:
  - Logs tendem a ficar muito grandes
    - Muito maiores do que os bancos de dados típicos conseguem lidar
  - Política de rotacionamento e compressão são fundamentais
  - Ter quem leia e analise o log regularmente também
    - Senão, o log só servirá para ocupar espaço em disco

Anti-Spoofing

- Verificação extra segundo endereços configurados *em cada interface*.
- Tenta prevenir o ataque de IP Spoofing *entre redes diferentes.*
  - Naturalmente, não previne IP spoofing em uma mesma rede, dado que o tráfego não passa pelo firewall.
- Descarta o pacote se:
  - O endereço de origem do pacote não está dentro da faixa configurada para a interface por onde está entrando
  - O endereço de saída do pacote não está dentro da faixa configurada para a interface por onde ele está saindo
    - O roteamento IP clássico deveria tornar isso impossível de acontecer... mas NATs e VPNs às vezes geram isso.
- Pode ser implementado na base de regras se a linguagem de especificação das regras permitir condições baseadas nas interfaces de entrada e saída

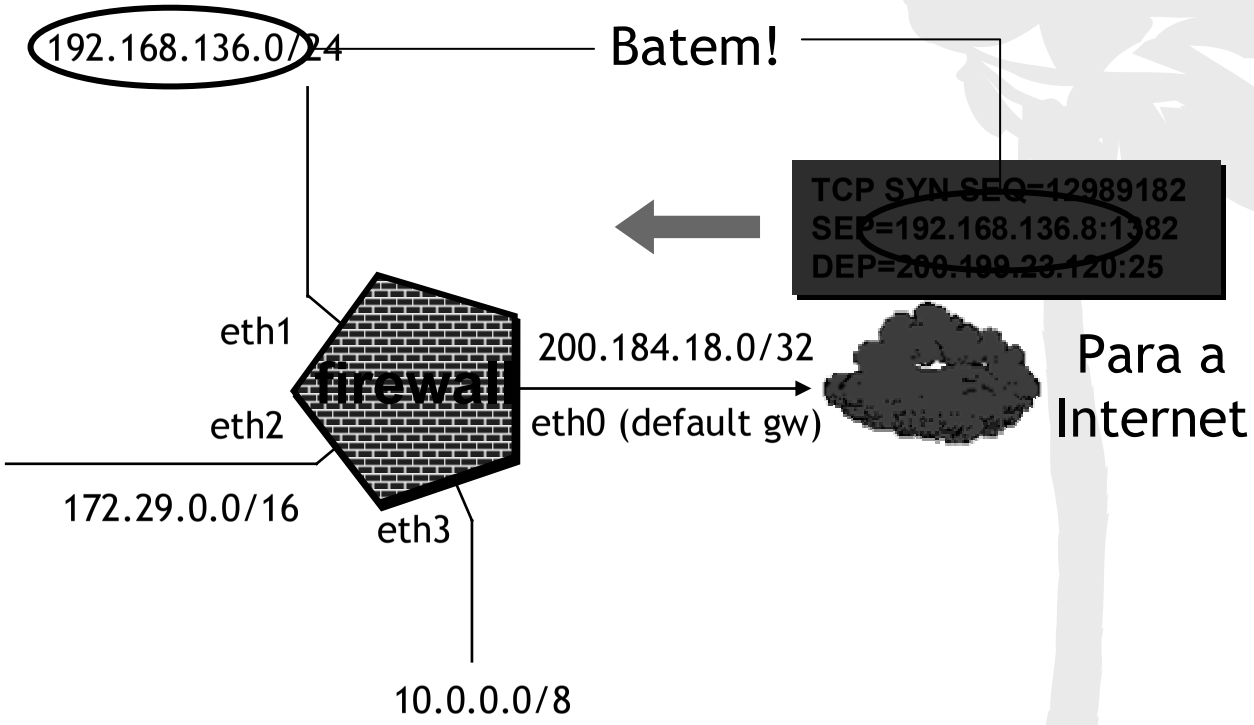
Anti-Spoofing: Exemplo

- Pacotes vindo das redes internas devem necessariamente ter seu endereço de origem dentro da faixa de IPs válidos para o IP/Netmask da sua interface



Anti-Spoofing: Exemplo

- Um pacote vindo da Internet não pode ter como endereço de origem dentro da faixa de nenhuma das interfaces internas



Outros Aspectos

- Acompanhamento/Tradução do Número de Seqüência
  - Detecta/diminui chance de IP Spoofing por previsão do ISN
  - Não há um consenso sobre se é realmente necessário
- Descarte de pacotes com IP Options
  - Quase nunca necessários
- OS fingerprinting passivo embutido
- Proteção contra DoSs comuns, tais como SYN Floods
- Resistência a exaustão da Tabela de Conexões
  - Tipicamente dimensionada para ~ 20000 conexões
  - Política de descarte pode ajudar

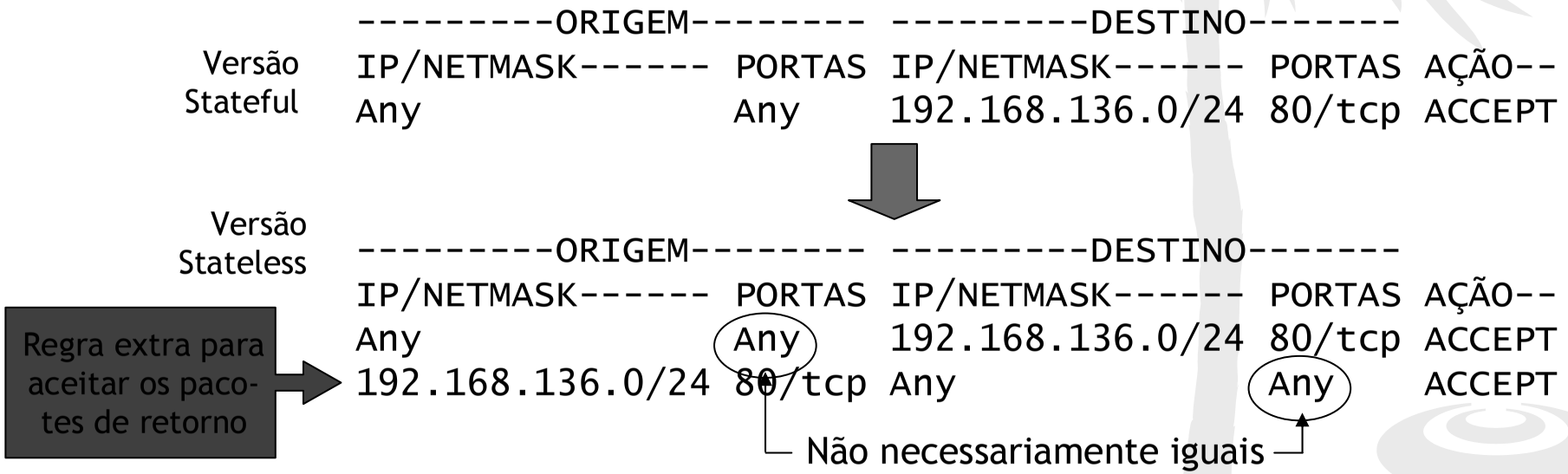
Parte III: Filtros de Pacotes Tradicionais

Filtros de Pacotes Tradicionais

- Não temos a Tabela de Conexões
- Força que tratemos os pacotes diretos e de retorno manualmente na base de regras
  - Bases de regras mais longas e mais complicadas
  - Sem o “efeito cache”, a avaliação da base de regras precisa ser feita para cada pacote
    - E tende a ficar tão mais cara quanto maior for a base de regras
  - Portas efêmeras forçam as regras a serem mais frouxas do que o desejado
- Por outro lado, não há tabela de estado para estourar: menos vulnerável a DoS por exaustão de recursos

Filtros de Pacotes Tradicionais

- A ausência da Tabela de Conexões faz com que cada regra de aceitação se torne duas em um filtro sem estado
  - Uma regra para cada direção (original e de retorno) da conexão
  - Porta efêmera tem de ficar em Any (ou em uma faixa ampla)
    - Mais permissivo do que deveria, pois não há garantia de que o número da porta de origem do pacote de retorno seja o mesmo do número da porta de origem do pacote direto.



Anti-Spoofing

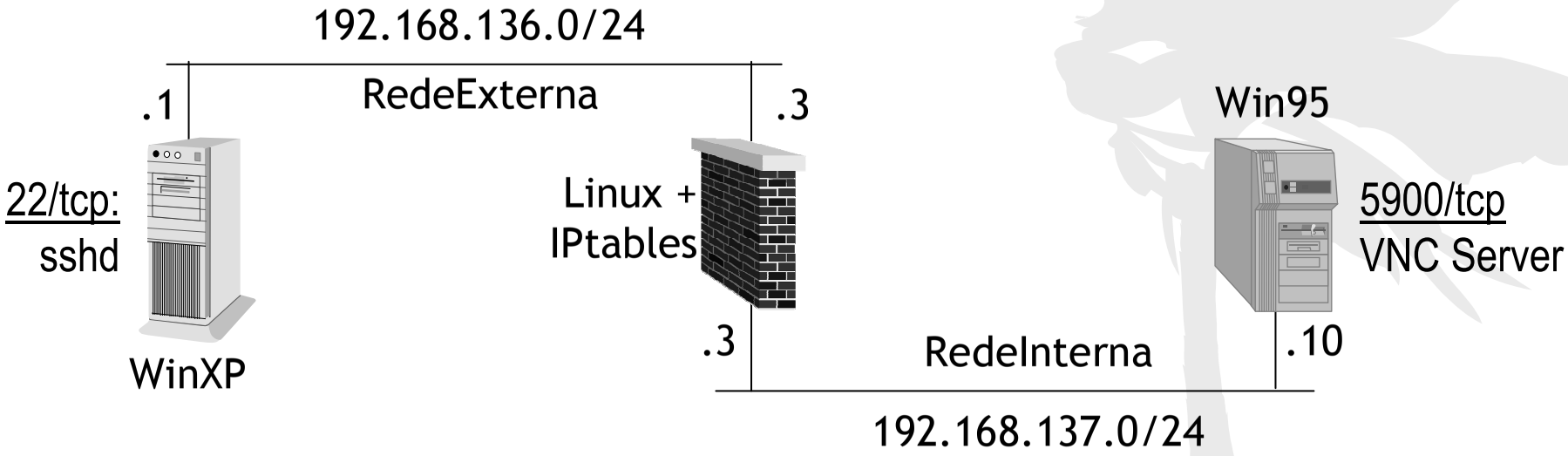
- Pode ser feito direto na base de regras, mas aumenta *muito* a quantidade de regras
  - Naturalmente, requer que as regras possas ser atreladas a interfaces específicas
- Alguns sistemas operacionais permitem que seja ativada juntamente com a configuração das interfaces, fora da base de regras

Checkpoint

- O termo “stateful” refere-se primariamente à existência da Tabela de Conexões, que atua como um cache de conexões já aceitas que melhora a performance e simplifica a base de regras. São normalmente usados nas “pontas” do backbone.
- Não obstante, os firewalls stateful requerem minucioso acompanhamento do protocolo e por vezes vários artifícios técnicos, dos quais os administradores deveriam estar cientes.
- Contudo, uma única porta aberta, bem utilizada (i.e., usando-se programas que permitem repasse de portas, proxying ou tunelamento) pode ser usada para burlar as regras.
- Filtros de pacotes clássicos, apesar de funcionalmente mais simples, tendem a gerar bases de regras mais longas, complicadas, lentas e difíceis de manter. Sem a tabela de conexões, contudo, têm menos um ponto de possível exaustão de recursos, tornando-as mais adequadas para ficarem “no meio do backbone”.

Demonstrações

- Arquitetura da rede simulada
  - Exemplo simplificado; na prática, tende a ser mais complicado



Firewall Piercing

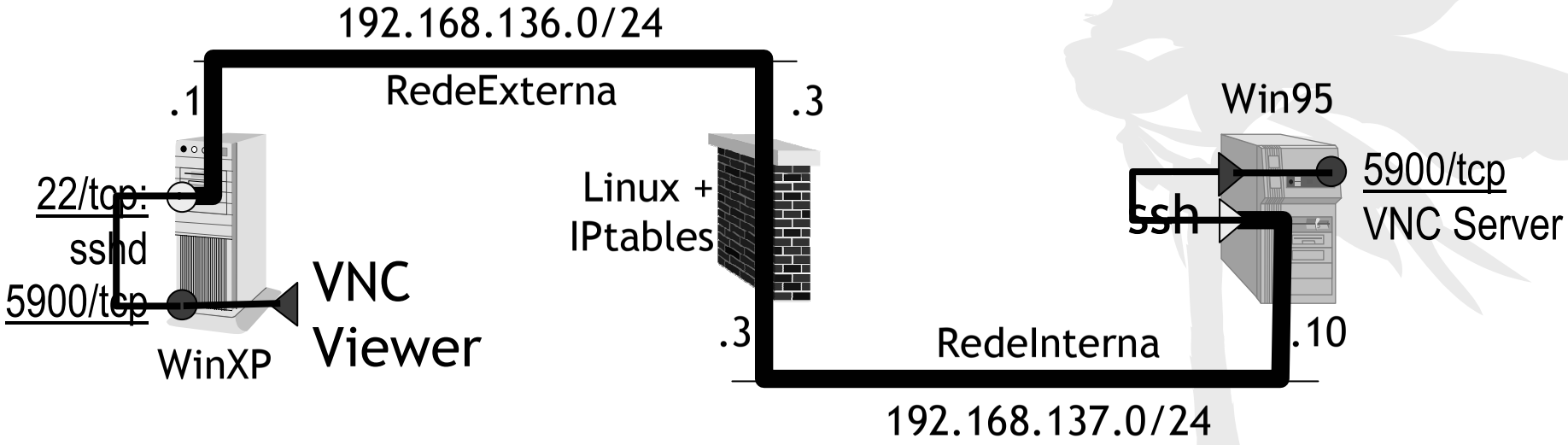
- Técnicas para burlar restrições de firewalls
- “Pierce” não é um bom nome
  - A técnica não “fura” as restrições. Ela consiste em canalizar o tráfego desejado de uma forma que não fira as restrições impostas pela base de regras
- Requer preparação prévia
  - O usuário tem de ter previamente instalado algum software dos dois lados para fazer o repasse
  - Pode ser um proxy, tunelador, etc.
  - No nosso exemplo, usaremos o SSH atuando como proxy (a docuementação do SSH chama esse recurso de “port forwarding”)

Firewall Piercing

- Não é intrinsecamente errado do ponto de vista ético
  - Pode ser usado para o bem ou para o mal
- Furando proxies HTTP excessivamente restritivos
  - HTTPTunnel: cria um canal bidirecional sob o protocolo HTTP usando POSTs
- Princípio generalizável
  - Basta um único canal bidirecional ser permitido e escrever um programa que formate os dados segundo o protocolo desejado (DNSTunnel? ICMPTunnel?)
- Não é realmente um conceito novo
  - Citado desde os livros clássicos sobre firewalling
  - Surpreendentemente pouco conhecido na prática
- Lição de humildade
  - Não leve seu firewall tão a sério.

Firewall Piercing

- Exemplo com o SSH atuando como proxy (port forwarding)
  - Passando tráfego não permitido “por dentro” de tráfego permitido



Parte IV: Tradução de Endereços

Tradução de Endereço

- Costumam ter bases de regras próprias
  - Separadas da base de regras que decide aceitação/rejeição
  - Ações, que descrevem o destino dos pacotes, têm uma forma bem diferente
    - Requerem parâmetros para descrever como os pacotes deverão ser traduzidos
  - Requerem uma tabela de estado própria
- Posicionamento em relação ao roteamento
  - Antes da decisão de roteamento
  - Após a decisão de roteamento
  - Alguns firewalls permitem somente de uma das formas

Tradução de Endereço Estática

- Consiste em alterar
  - O endereço IP de origem dos pacotes
  - O endereço IP de destino dos pacotes
- Traduz uma faixa de endereços IP para outra
  - Normalmente, um endereço ou endereços contíguos
  - Mapeamento um-a-um: quantidade de IPs originais e traduzidos é exatamente a mesma
  - Pode ser feto de forma totalmente sem estado (“stateless”)
    - Por isso um tanto inapropriadamente chamado de “estático”
  - Mas, se há uma tabela de estado, interage com ela

Tradução de Endereço Dinâmica

- Consiste em alterar
  - O endereço IP e porta de origem dos pacotes
  - O endereço IP e porta de destino dos pacotes
  - Não altera a unicidade da tupla de conexão
- Traduz uma faixa de endereços IP para outra de tamanho menor
  - Freqüentemente um só
- Altera a porta (de origem ou destino)
  - Imprescindível para manter a unicidade da tupla de conexão
  - Torna-o necessariamente stateful, pois é preciso manter um mapeamento (efêmero e altamente dinâmico) sobre que portas/endereços traduzidos “escondem” que portas e endereços originais

NAT HIDE

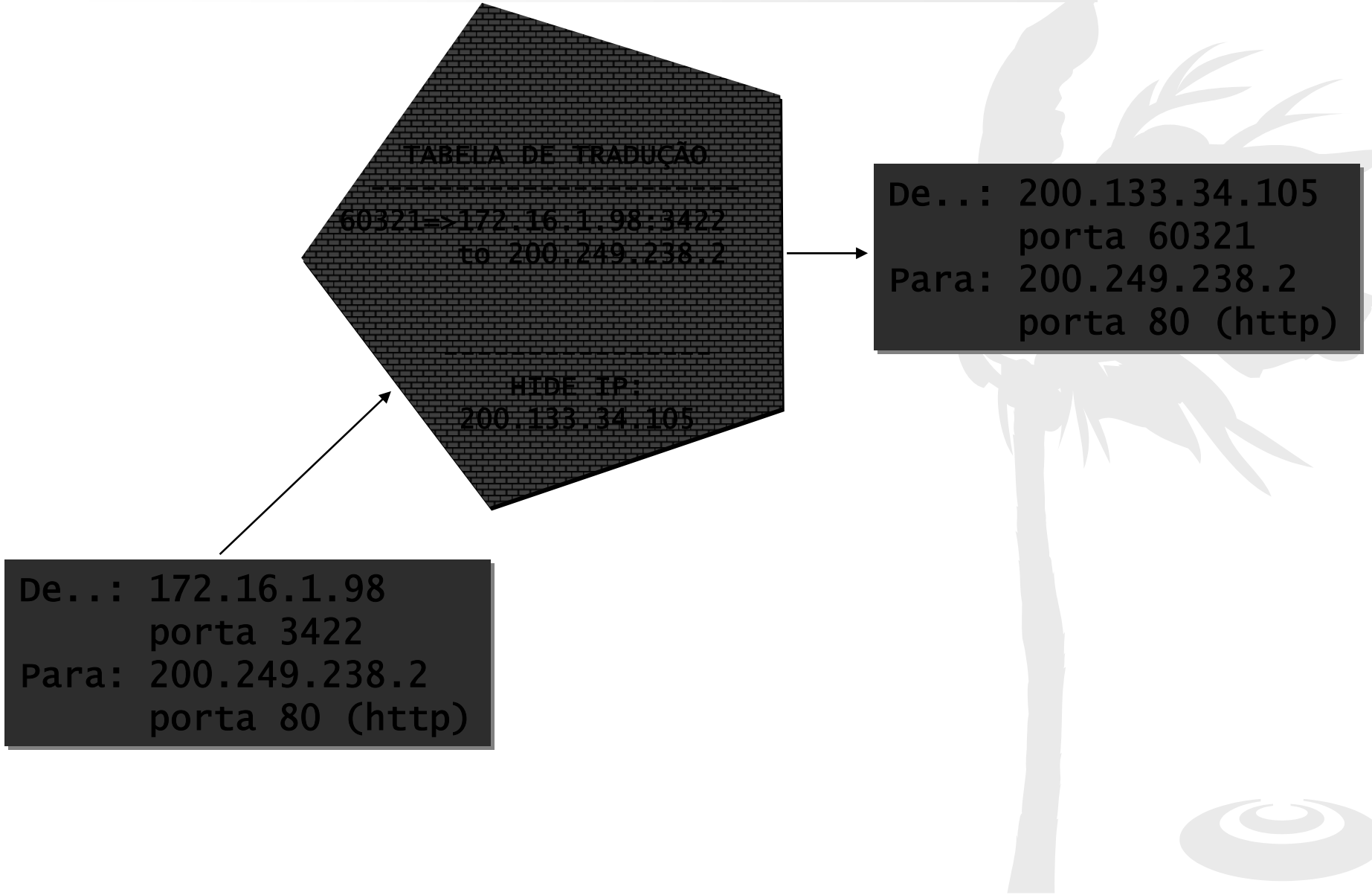


TABELA DE TRADUÇÃO
60321=>172.16.1.98:3422
to 200.249.238.2
HIDE IP:
200.133.34.105
De..: 200.133.34.105
porta 60321
Para: 200.249.238.2
porta 80 (http)
De..: 172.16.1.98
porta 3422
Para: 200.249.238.2
porta 80 (http)

NAT HIDE

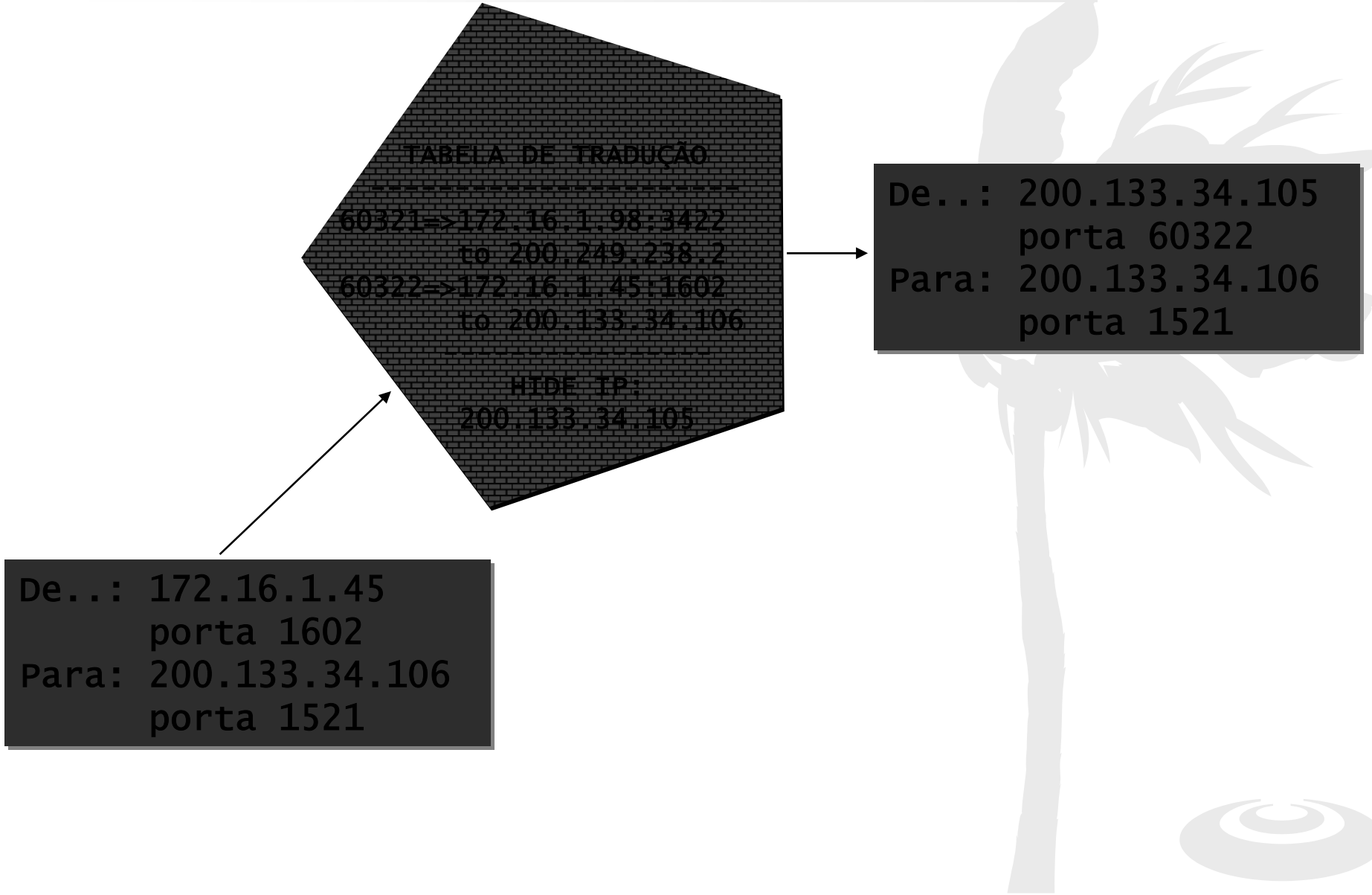


TABELA DE TRADUÇÃO
60321=>172.16.1.98:3422
to 200.249.238.2
60322=>172.16.1.45:1602
to 200.133.34.106
HIDE IP:
200.133.34.105
De..: 172.16.1.45
porta 1602
Para: 200.133.34.106
porta 1521
De..: 200.133.34.105
porta 60322
Para: 200.133.34.106
porta 1521

NAT HIDE

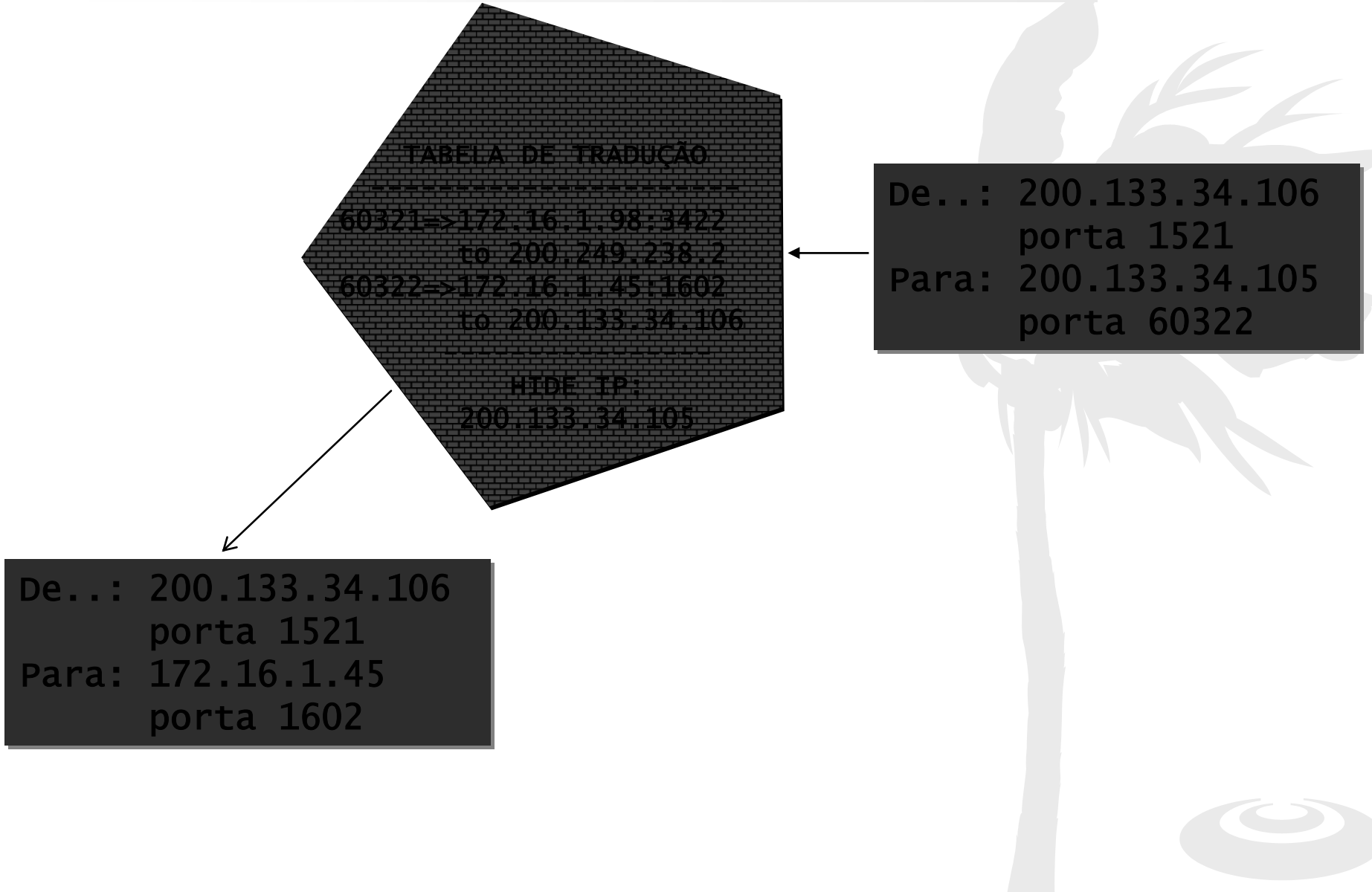


TABELA DE TRADUÇÃO
60321=>172.16.1.98:3422
to 200.249.238.2
60322=>172.16.1.45:1602
to 200.133.34.106
HIDE IP:
200.133.34.105
De..: 200.133.34.106
porta 1521
Para: 200.133.34.105
porta 60322
De..: 200.133.34.106
porta 1521
Para: 172.16.1.45
porta 1602

NAT HIDE

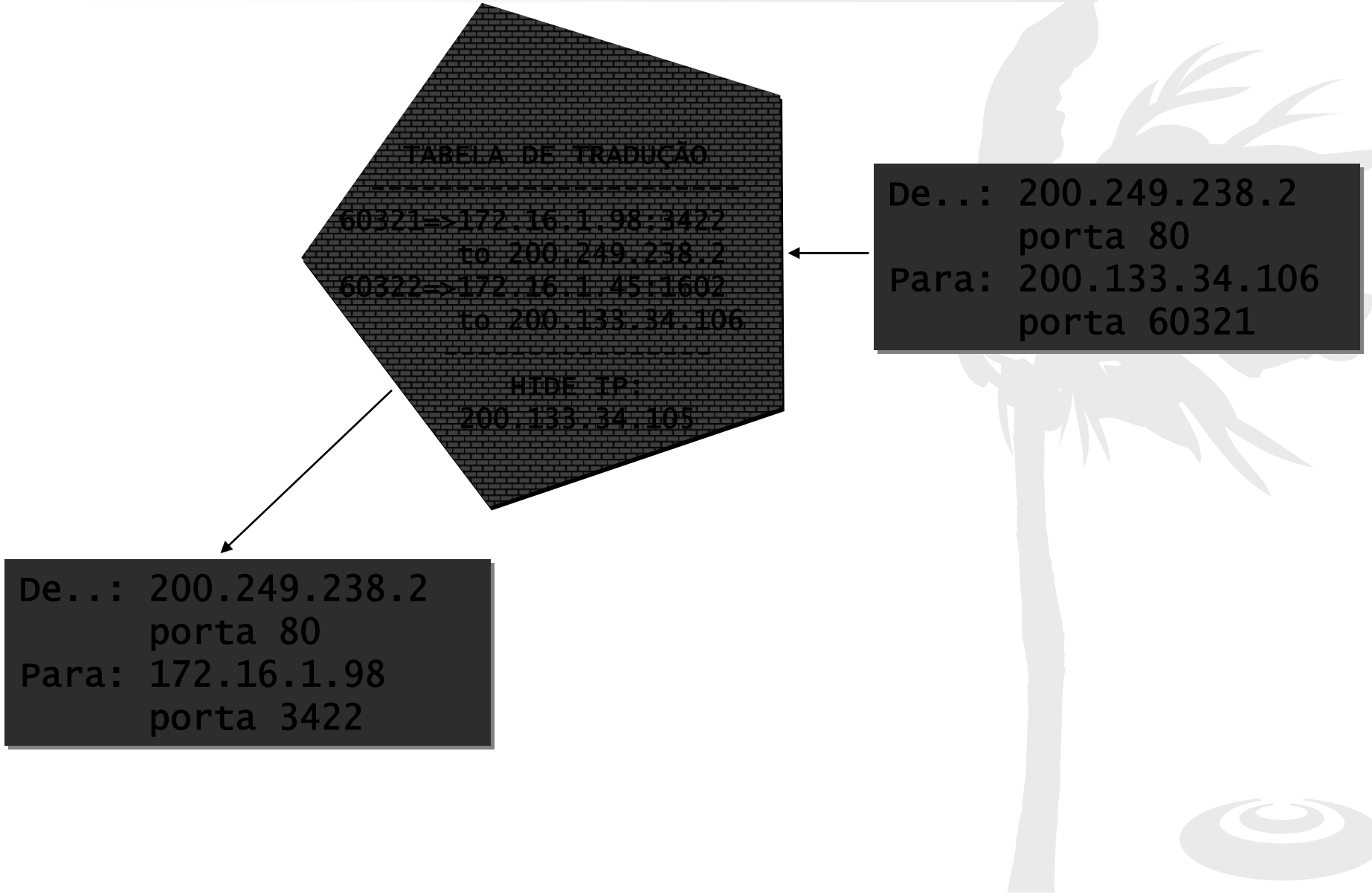


TABELA DE TRADUÇÃO
60321=>172.16.1.98:3422
to 200.249.238.2
60322=>172.16.1.45:1602
to 200.133.34.106
HIDE IP:
200.133.34.105
De..: 200.249.238.2
porta 80
Para: 200.133.34.106
porta 60321
De..: 200.249.238.2
porta 80
Para: 172.16.1.98
porta 3422

Tradução de Endereços

- Cada implementação tem suas firulas e limitações...
  - Nos roteadores Cisco, há um conceito/limitação de “inside” e “outside” completamente desnecessário
  - IP Pools versus um único IP
- ...e sua própria nomenclatura
  - SNAT/DNAT vs NAT HIDE
  - PAT

Parte V: Tendências, Necessidades, Idéias

Desenvolvimentos Recentes

- Bridge-Firewalls (Bridge-Level IP Packet Filtering)
  - Firewalling atua antes do roteamento IP
  - Dispensa mudar os default gateways das máquinas das redes vizinhas ao firewall
  - Firewall “invisível”: não aparece no traceroute
- Bridge-Level Firewalls
  - Fazem o mesmo que os filtros tradicionais, mas diretamente na camada 2
  - Filtragem por protocolo: já pensou em conter IPX, Appletalk, e outros legados/esquisitices?
  - Filtragem eficiente por endereços MAC:
    - Não confundir com possibilidade de filtragem por endereços MAC na base de regras de um filtro de pacotes IP
  - Tradução de endereços MAC
  - Contenção de broadcasts e multicasts
  - Exemplo emblemático: EBTables

Desenvolvimentos Recentes

- Halted Firewalls
  - Após a instalação da base de regras, efetua a parada (“halt”) do sistema operacional
  - *Todos* os processos de usuário são terminados (inclusive o init)
    - Por isso mesmo, menos vulnerável
  - Apenas o núcleo (kernel) do SO fica rodando, em loop infinito
  - Mas, como o serviço de roteamento e firewalling é no kernel, os pacotes continuam fluindo
  - Mudar algo → reboot: parada no serviço
  - Sem logging, nem acesso remoto
    - A menos que alguém escreva módulos de kernel para isso
  - Útil para instalações tipo “instale e esqueça” (ou appliances) onde não se antevê mudanças freqüentes nas bases de regras

Desenvolvimentos Recentes

- Integração Firewalls + NIDSs
  - Parece o caminho natural, mas por ora recomenda-se ceticismo e cautela
  - NIDSs ainda são imaturos, instáveis e ineficientes: excesso de falsos positivos tornam sua utilidade questionável
  - NIDSs são lentos, requerem longa lista de comparações em cada pacote: nega a vantagem da tabela de conexões como cache de aceitação.

Necessidades

- Infra-estrutura de logging decente
  - Registro confiável transacional, rotacionamento e arquivamento robusto, indexação para consulta rápida
  - Gerenciamento decente do espaço em disco
    - Minimamente, não parar de registrar log quando o disco enche; avisar antes e sobrescrever os registros antigos
  - Analisadores de logs mais inteligentes e que dêem resultados genuinamente úteis
- Lado humano: carência de técnicos que
  - Conheçam a fundo os protocolos de rede
  - Não tenham medo de usar sniffers: é imprescindível para depurar problemas reais

Continua ano que vem...

- Proxies
- Encapsulamento e Tunelamento
- Tunelamento + Criptografia → VPNs
- VPNs: Redes Virtuais Privadas
- Complexidade, Evolução no Tempo e Otimização das Bases de Regras

Referências

- William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, *Firewalls and Internet Security: Repelling the Wily Hacker, 2nd Edition*, 2003, Addison-Wesley Pub Co, ISBN 020163466X
- Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, *Building Internet Firewalls, 2nd Edition*, 2000, O'Reilly & Associates, ISBN 1565928717
- W. Richard Stevens, *TCP/IP Illustrated Volume 1: The Protocols*, 1994, Addison-Wesley Co., ISBN 0201633469
- Lance Spitzner, *How Stateful is Stateful Inspection: Understanding the Firewall-1 State Table*, http://www.spitzner.net/fwtable.html
- Rene Rideau, *Firewall Piercing Mini-HowTo*, http://secinf.net/firewalls_and_VPN/Firewall_Piercing_miniHOWTO/
- *GNU HTTPTunnel*, http://www.nocrew.org/software/httptunnel.html

Referências

- Marco Carnut, Cristiano Lincoln Mattos, Evandro Hora, Fábio Silva, *Improving Stateful Inspection Log Analysis*, 2001, Anais do 3o Simpósio Segurança em Informática, ITA/SJC
- Marcelo Lima, Paulo Lício de Geus, *Comparação entre Filtros de Pacotes com Estados e Tecnologias Tradicionais de Firewall*, 1999, Anais do 1o Simpósio Segurança em Informática, ITA/SJC
- Mike Murray, *SysAdmin: Halted Firewalls (Running Linux Firewalls at Run Level 0)*, http://linuxtoday.com/security/200202080420SCHL
- Ethernet Bridge Tables, http://ebtables.sourceforge.net/