



República Federativa do Brasil
Ministério da Indústria, Comércio Exterior
e Serviços
Instituto Nacional da Propriedade Industrial

(11) PI 0401894-0 B1



(22) Data do Depósito: 02/06/2004

(45) Data de Concessão: 13/11/2018

(54) Título: ARQUITETURA DE BARREIRA DE PROTEÇÃO EM MÚLTIPLAS CAMADAS

(51) Int.Cl.: H04L 29/06.

(52) CPC: H04L 63/0218; H04L 63/0227; H04L 63/164.

(30) Prioridade Unionista: 06/06/2003 US 10/456,766.

(73) Titular(es): MICROSOFT TECHNOLOGY LICENSING, LLC.

(72) Inventor(es): BRIAN D. SWANDER; PAUL G. MAYFIELD.

(57) Resumo: "ARQUITETURA DE PORTAL DE PROTEÇÃO EM MÚLTIPLAS CAMADAS". Um método e um sistema são providos para implementação de uma arquitetura de portal de proteção em um dispositivo de rede. A arquitetura de portal de proteção inclui uma pluralidade de camadas de rede, um primeiro motor de portal de proteção e um ou mais módulos de chamada. As camadas enviam pacotes e informação de pacote ao primeiro motor de portal de proteção, mantém e passam o contexto de pacote às camadas subsequentes e processam os pacotes, o primeiro motor de portal de proteção compara a informação de pacote a um ou mais filtros instalados e retorna uma ação às camadas, indicando como tratar o pacote. As camadas provêm funcionalidade adicional, tal como características de detecção de intrusão, cadastramento e controle pelos pais.

"ARQUITETURA DE BARREIRA DE PROTEÇÃO EM MÚLTIPLAS CAMADAS"

APLICAÇÕES RELACIONADAS

O presente pedido contém matéria pertinente relacionada ao do Pedido de Patente "Multy-Layer Based Method for Implementing Network Firewalls", Número de Documento 221039; "A Method and Framework for Integrating a Plurality of Network Policies", número de Documento 221041; e "Method for Managing Network Filter Based Policy", Número de Documento 221037; registradas no mesmo dia do presente pedido e as divulgações das mesmas estando expressamente incorporadas a este caso, à guisa de referência.

ÁREA TÉCNICA

Esta invenção, de forma geral, se refere a sistemas de computadores e à segurança de de rede. Mais particularmente, esta invenção se refere à arquitetura de barreira de proteção ("firewall"), implementada em um ou mais dispositivos de rede.

ANTECEDENTES DA INVENÇÃO

Protocolos de rede se designam a facilitar a comunicação entre dispositivos de rede através de uma permuta aberta de dados. Enquanto a permuta aberta de dados aumenta altamente a utilização de dispositivos de rede para concretização de tarefas, esta também gera problemas devido aos protocolos do rede não estarem desenhados para, e de forma geral não provêm, segurança de rede. Computadores acoplados tanto a redes públicas quanto privadas, tais como Redes de Área Local (LANs), Redes de Área Ampla (WANs), Intranet e



Internet ficam suscetíveis a ataques maliciosos perpetrados por outros dispositivos de rede acoplados direta ou indiretamente à rede. Tais ataques maliciosos incluem furto de dados, ataques de negação de serviço (DOS), proliferação de vírus de computador e similares. Outros itens relacionados aparecem quando se acoplam computadores a redes, tais como acesso controlado a sites indesejáveis e não apropriados por crianças.

Uma barreira de proteção trata-se de uma ferramenta utilizada para proteger usuários individuais, dispositivos de rede e redes de forma geral de ataques maliciosos, enquanto também adiciona a capacidade de controle de permuta de dados através do rede, por meio de implementação de uma diretriz. A barreira de proteção implementa a diretriz por meio do exame dos pacotes de rede e determinação, com base neste exame, se estes pacotes devem ser permitidos ou ao contrário bloqueados contra passagem adicional pela rede.

A diretriz implementado através da barreira de proteção se define por um ou mais filtros. Cada filtro inclui parâmetros de filtração e uma ação associada. Os parâmetros de filtração são utilizados para identificação de pacotes de rede que se sujeitam à diretriz da barreira de proteção e incluem informações tais como endereços de hardware, por exemplo, endereços de Controle de Acesso de Meios (MAC), endereços de rede, por exemplo, endereços de Protocolos de Internet (IP), tipos de protocolos, por exemplo, Protocolos de Controle de Transporte (TCP), números de porta e similares. A ação define como os pacotes com parâmetros que se



conjugam aos parâmetros de filtros devem ser tratados. Como exemplo específico, o filtro inclui como seu parâmetro um endereço de Localizador de Fonte Uniforme - Uniform Resource Locator (URL), por exemplo "http://www.foo.com". O filtro
5 ainda associa a ação de bloqueio, isto é descontinuação do pacote, com a do endereço URL. Sempre que os portais de proteção examinam um pacote e através deste exame identificam o endereço URL "http://www.foo.com" encaixado neste pacote, a barreira de proteção descontinua o pacote, com isso impedindo o mesmo de atravessar a rede.
10

Dispositivos de rede permutam dados por meio e envio e recebimento de pacotes através de um empilhamento de rede, compreendendo uma arquitetura de rede em camadas. Embora existam diferentes modelos de arquitetura de rede, a
15 maioria inclui pelo menos uma camada de aplicação, uma camada de transporte, uma camada de rede e uma camada de ligação. Os pacotes de rede atravessam cada camada em sequência, e à medida que cada camada é atravessada, este pacote fica sujeito ao processamento. Para pacotes para o exterior, a
20 camada de aplicação processa os dados de acordo com protocolos de aplicação, tais como Protocolo de Transferência Hiper Texto (HTTP), Protocolo de Transferência de Arquivos (FTP) e Protocolo de Transferência de Correspondência Simples (SMTP), para designar apenas alguns. Outras camadas, tais
25 como camada de rede e camada de transporte embalam estes dados por meio de encaixe dos mesmos em cabeçotes TCP e IP. As camadas efetuam um processamento recíproco para pacotes restritos através de, por exemplo, ferramentas de análise, de-



sempalagem de dados etc. O "empilhamento" de arquitetura em camadas e a função de processamento concretizado por estas camadas resulta em uma estrutura em pacote dinâmica, fato pelo qual o conteúdo do pacote incluindo os parâmetros de
5 pacote se modifica à medida que este pacote atravessa o empilhamento de protocolo de rede.

Portais de proteção examinam pacotes em um ponto de inspeção, localizado no interior do empilhamento de rede em camadas. Em um extremo, o ponto de inspeção se trata da
10 camada de aplicação. Por exemplo, a barreira de proteção se apresenta como um Provedor de Serviço em Camadas (LSP). Os pacotes na camada de aplicação incluem os dados subjacentes que serão transmitidos a outro dispositivo de rede ou que tenham sido recebidos a partir de outro dispositivo de rede.
15 O exame do pacote na camada de aplicação permite que a barreira de proteção identifique os parâmetros de camada de aplicação, tais como endereços URL e compare estes parâmetros de camada de aplicação aos parâmetros de filtro. No entanto, outros parâmetros de pacote, tais como endereços IP, números
20 de portais, endereços MAC e similares não ficam disponíveis devido ao fato de não terem sido adicionados aos pacotes para o exterior ou terem sido analisados além dos pacotes internos.

No outro extremo, o ponto de inspeção da barreira
25 de proteção se encontra implementado em níveis inferiores do empilhamento de rede, como um acionador intermediário, interposto entre a camada de ligação e a camada de rede. Os pacotes de níveis inferiores do empilhamento de rede incluem



uma quantidade máxima de parâmetros, por exemplo, números de identificação, endereços MAC, endereços IP, tipos de protocolos, portas e dados de carga de retribuição. Embora os pacotes incluam tais parâmetros, isso não significa que estes parâmetros possam ser prontamente identificados. Após a barreira de proteção receber o pacote, esta barreira de proteção necessita analisar e interpretar os parâmetros de pacote relevantes, para comparação com os parâmetros de filtro. Assim sendo, tanto as camadas no empilhamento de rede quanto à do barreira de proteção efetuam funções de análise e de interpretação de pacote redundantes.

SUMÁRIO DA INVENÇÃO

A presente invenção se refere a uma arquitetura de barreira de proteção que permita a aplicação de filtros de barreira de proteção a pacotes de rede em todas as camadas no interior do empilhamento de rede. A arquitetura inclui um conjunto de processadores de camadas, capaz de processar parâmetros de camadas a partir do pacote de rede. Os parâmetros de camada tratam-se de parâmetros associados com o processador de camada e incluem parâmetros que o processador de camadas analisa a partir de e adiciona a ou deriva de alguma forma a partir do pacote de rede. Os processadores de camada ainda são adicionalmente capazes de emitir requisições de classificação que incluem os parâmetros de camada.

A arquitetura ainda inclui um primeiro motor de barreira de proteção, compreendendo uma interface de camada, um conjunto de filtros instalados e um componente de consulta. A interface de camada recebe os parâmetros de camada en-



viados como parte da requisição de classificação. O primeiro componente de consulta de motor de barreira de proteção faz uso dos parâmetros de camada pra identificar um ou mais filtros que se conjuguem, a partir do conjunto de filtros instalados. Cada filtro que se conjuga inclui uma ação que instrui o processador de camada a permitir que o pacote atravessasse adicionalmente a rede ou a bloquear o pacote. O primeiro motor de barreira de proteção retorna, pelo menos uma ação ao processador de camada através da interface de camada.

Em uma modalidade da invenção, os processadores de camada mantêm o contexto do pacote. O processador de camada recebe o contexto de pacote a partir de um processador de camada prévio e envia o contexto de pacote ao primeiro motor de barreira de proteção. O componente de consulta no primeiro motor de barreira de proteção faz uso do contexto de pacote em combinação com os parâmetros de camada, para identificar um ou mais filtros que se conjugam. O processador de camada também modifica o contexto de pacote por meio da adição de parâmetros de camada e envia então o contexto de pacote modificado a um próximo processador de camada.

Em outra modalidade da invenção, uma ou mais chamadas são incluídas como parte da arquitetura de barreira de proteção. Cada um dentre os um ou mais módulos de chamada providos adiciona capacidade de funcionamento (além da permissão e bloqueio), tais como funções de cadastro de pacotes, verificação de Segurança de Protocolo de Internet (IP-Sec) e execução de controle pelos pais. O módulo de chamada



e executado no pacote quando este é identificado como ação em um ou mais filtros que se conjugam.

Em outra modalidade da invenção, se provê um segundo motor de barreira de proteção. A segundo motor de barreira de proteção adiciona novos filtros ao conjunto de filtros instalados, através da administração API. O segundo modo de barreira de proteção também inclui uma instância do primeiro motor de barreira de proteção, denominado de módulo de filtro que replica os serviços do primeiro motor de barreira de proteção no interior do segundo motor de barreira de proteção.

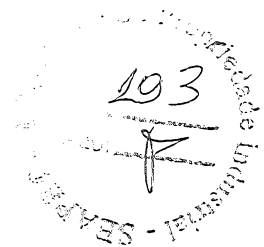
Características e vantagens adicionais da invenção tornar-se-ão aparentes a partir da descrição detalhada a seguir de modalidades ilustrativas, que procede com referência às Figuras em anexo.

BREVE DESCRIÇÃO DOS DESENHOS

Embora as reivindicações em anexo apresentem as características da presente invenção de forma particular, a invenção, em conjunto com seus objetivos e vantagens pode ser mais bem entendida a partir da descrição detalhada a seguir, tomada em conjunto com os desenhos em anexo, dos quais:

A Figura 1 é um diagrama de blocos, ilustrando, de forma geral, um exemplo de sistema de computador, no qual reside a presente invenção;

A Figura 2 é um diagrama de blocos, ilustrando, de forma geral, um exemplo de ambiente de rede pelo qual a presente invenção é aplicada;



A Figura 3 é um diagrama de blocos, ilustrando, de forma geral, a arquitetura de barreira de proteção da presente invenção;

5 A Figura 4 é um diagrama de blocos, ilustrando um exemplo de filtro utilizado com a presente invenção;

A Figura 5 é um diagrama de blocos, ilustrando um exemplo de estrutura de dados para contexto de pacote, utilizado com a presente invenção;

10 A Figura 6 é um diagrama de blocos, ilustrando um exemplo de conjunto de interfaces de aplicação de programação, utilizado com a presente invenção;

A Figura 7 é um diagrama de blocos, ilustrando um exemplo de conjunto de interfaces de aplicação de programação, utilizado com a presente invenção;

15 A Figura 8 é um diagrama de blocos, ilustrando funções executadas por camadas de refere de acordo com a presente invenção; e

A Figura 9 é um diagrama de blocos, ilustrando um exemplo de conjunto de chamadas utilizado com a presente invenção.
20

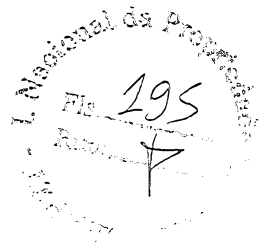
DESCRIÇÃO DETALHADA DA INVENÇÃO

Divulga-se uma arquitetura de barreira de proteção ou de proteção que permite a filtração de processos em múltiplas camadas, denominadas neste caso como camadas. A arquitetura inclui um processo de modo para usuário e um processo de modo núcleo sendo executados em um sistema de operação. Alternativamente, a arquitetura de barreira de proteção é executada no exterior do processo de operação em um ou
25



mais módulos de programa ou no interior de um único modo de sistema de operação.

O processo de modo núcleo inclui uma pluralidade de camadas compreendendo um empilhamento de protocolos, um motor de barreira de proteção núcleo e uma ou mais chamadas. O empilhamento de protocolo inclui uma camada de aplicação, uma camada de transporte, uma camada de rede e uma camada de ligação. Camadas adicionais são adicionadas ou deletadas ou excluídas da arquitetura, conforme desejado. As camadas, cada uma configura uma camada de requisição, capaz de receber um pacote de rede e dados de contexto de pacote correspondentes a partir de uma camada ou processo prévios. A camada de requisição lança então uma requisição de classificação ao motor de barreira de proteção núcleo através da camada API. A requisição de classificação inclui o pacote recebido pela camada de requisição, o contexto de pacote e um conjunto de parâmetros de camada, associados com a camada de requisição. O motor de barreira de proteção núcleo processa a requisição e retorna uma ação. A ação, à guisa de exemplo, instrui a camada de requisição como tratar o pacote (por exemplo, permitir ou bloquear o mesmo). Caso a ação seja de permissão, a camada de requisição processa o pacote de acordo com o protocolo de camada, modifica o contexto de pacote par incluir os parâmetros de camada e passa o pacote e o contexto de camada para a próxima camada. Caso a ação seja de bloqueio, a camada de requisição descontinua o pacote e não passa este pacote para a próxima camada. A camada de requisição pode efetuar funções adicionais como resultado da ação de blo-



queio, tal como retirada de uma conexão TCP, onde o pacote TCP é descontinuado.

O motor núcleo de barreira de proteção inclui a camada API, um conjunto de filtros instalados e uma camada API. Cada um dos conjuntos de filtros instalados inclui um conjunto de condições de filtro e uma ação associada. O motor núcleo de barreira de proteção processa a requisição de classificação enviada a partir da camada de requisição por meio de identificação de um ou mais filtros que se conjugam.

Os filtros que se conjugam apresentam condições de filtro que conjugam os parâmetros de camada e o contexto de pacote. Uma vez que os filtros que se conjugam foram identificados, estes são então aplicados em ordem de prioridade de filtro. Caso a ação do filtro sendo aplicado seja permitir ou bloquear, a ação é retornada à camada de requisição. Caso a ação seja chamada, a requisição de classificação lançada pela camada de requisição, em conjunto com a identificação do filtro conjugado, é passada a um dos módulos de chamada. O módulo de chamada efetua sua função programada e retorna uma ação ao motor núcleo de barreira de proteção.

O motor núcleo de barreira de proteção aplica filtros que se conjugam na ordem de prioridade do pacote, até pelo menos ser designada uma ação de término. Uma vez que a ação de término tenha sido designada, esta é retornada à camada de requisição e não é mais aplicado nenhum filtro conjugado adicional. Caso não seja identificado nenhum filtro conjugado para um pacote, a camada de requisição é notificada do fato de não terem sido encontrados filtros conjugados



e a camada de requisição decide então como tratar o pacote. Tipicamente, onde não tenha sido identificado nenhum filtro conjugado, a camada de requisição trata o pacote como se tivesse sido retornada uma ação de permissão.

5 Um exemplo de modo de usuário inclui um motor de modo de usuário de barreira de proteção e um ou mais provedores de diretrizes. Os provedores de diretrizes obtêm a diretriz a partir de qualquer fonte apropriada, tal como memória volátil ou não volátil ou são acessados pelos usuários
10 através de uma interface gráfica de usuário ou ferramenta similar. A diretriz se trata de uma fonte de informação para fornecimento de um novo filtro, inclusão de um conjunto de condições de filtro e ações associadas.

O modo de usuário também inclui uma instância do
15 motor de núcleo de barreira de proteção no interior do motor de usuário de barreira de proteção, com isso permitindo que sejam criadas as camadas de modo de usuário. As camadas de modo de usuário fazem uso então da instância de modo de usuário do motor núcleo de barreira de proteção para instalação
20 de filtros e identificação de filtros que se conjuguem a um conjunto de parâmetros, permitindo a aplicação de filtração no interior do modo de usuário.

Em uma modalidade da invenção, uma interface de chamada a partir do motor de barreira de proteção a um conjunto de módulos de chamada possibilita uma extensão virtualmente ilimitada das capacidades de barreira de proteção. À
25 guisa de exemplo, uma chamada de contexto HTTP provê uma característica de controle pelos pais por meio da identifica-



ção aceitável e não aceitável de endereços URL. Uma chamada IPSec verifica que os pacotes que se supõe estarem fazendo uso de IPSec tenham sido apropriadamente sujeitos ao processamento IPSec. Uma chamada de cadastro cadastra pacotes que
5 atendem a critérios estabelecidos, com isso facilitando um exame posterior dos pacotes, com base em algoritmos conhecidos.

Voltando aos desenhos, em que numerais de referência iguais se referem a elementos similares, a invenção encontra-se ilustrada como sendo implementada em um ambiente de computação apropriado. Embora não se trate de exigência, a invenção será descrita no contexto geral de instruções executáveis por computador, tais como módulos de programas sendo executados por um computador pessoal. De forma geral,
10 módulos de programas incluem rotinas, programas, objetos, componentes, estruturas de dados, etc., que executam tarefas específicas ou implementam tipos de dados abstratos específicos. A invenção também pode ser praticada em ambientes de computação distribuídos, onde as tarefas são executadas por
15 dispositivos de processamento remotos, que são ligados através de um rede de comunicações. Em um ambiente de computação distribuído, os módulos de programa podem ser localizados tanto em dispositivos de armazenamento de memória local, quanto remota.

25 A Figura 1 ilustra um exemplo de um ambiente de computação apropriado 100, no qual a invenção pode ser implementada. o ambiente de computação 100 trata-se apenas de um exemplo de um ambiente de computação apropriado e não su-



gere nenhuma limitação ao escopo de uso nem à funcionalidade da invenção. Tampouco o ambiente de computação 100 deveria ser interpretado como possuindo quaisquer dependências ou exigências relativas a quaisquer um ou combinação dentre os

5 componentes ilustrados no exemplo de ambiente de operação 100.

A invenção pode ser operada com numerosas outros ambientes de sistemas ou configurações com finalidades gerais ou finalidades especiais. Exemplos de sistemas de computação, ambientes e/ou configurações bem conhecidos que podem ser apropriados para uso com a invenção incluem, contudo

10 não se limitam a, computadores pessoais, computadores de servidor, dispositivos de manuseio manual ou laptops, sistemas de multi - processamento, sistemas com base em micro processamento, aparelho para acessar Internet pela TV, dispositivos eletrônicos programáveis para consumidores, PCs de

15 rede, mini computadores, computadores mainframe ou de estrutura principal. ambientes de computação distribuídos que incluam quaisquer dentre os sistemas ou dispositivos acima e similares.

A invenção pode ser descrita no contexto geral de instruções executáveis por computador, tais como módulos de programa que podem ser executados por computador. De forma geral, módulos de programa incluem rotinas, programas, objetos, componentes, estruturas de dados, etc. que efetuam tarefas específicas ou que implementam tipos de dados abstratos

20 específicos. A invenção também pode ser praticada em ambientes de computação distribuídos, em que as tarefas são efetuadas por dispositivos de processamento remoto que são



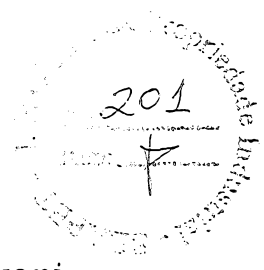
ligados através de rede de comunicações. em um ambiente de
computação distribuído, os módulos de programa podem ser lo-
calizados tanto em meios de armazenamento de dados local,
quanto remoto, incluindo dispositivos de armazenamento de
5 memória.

Com referência à Figura 1, um exemplo de sistema
para implementação da invenção inclui um dispositivo de com-
putação de finalidade geral em forma de um computador 110. os
componentes do computador 110 podem incluir, mas não se li-
10 mitam a uma unidade de processamento 120, um sistema de me-
mória 130 um barramento de sistema 121, que acopla vários
componentes de sistema, incluindo o sistema de memória à u-
nidade de processamento 120. O barramento de sistema 121 po-
de ser qualquer um dentre diversos tipos de estruturas de
15 barramento, incluindo um barramento de memória ou controla-
dor de memória, um barramento periférico, e um barramento
local, fazendo uso de quaisquer variedades de arquiteturas
de barramento. À guisa de exemplo e não de limitação, tais
arquiteturas incluem Barramento Arquitetura de Indústria Pa-
20 drão (ISA), Barramento de Arquitetura de Micro Canal (MCA),
barramento ISA aperfeiçoado (EISA) barramento local de Pa-
drões Associados de Vídeo Eletrônicos (VESA) e barramento de
Interconexões de Componentes Periféricos (PCI), também co-
nhecido como barramento Mezzanine.

25 O computador 110, tipicamente inclui uma variedade
de meios que podem ser lidos por computador. Os meios que
podem ser lidos por computador podem ser quaisquer meios
disponíveis que possam ser acessados através do computador



110 e incluem tanto meios voláteis, quanto meios não volá-
teis, meios removíveis e meios não removíveis. À guisa de
exemplo e não de limitação, meios que podem ser lidos por
computador podem compreender meios de armazenamento de com-
putador e meios de comunicação. Meios de armazenamento de
computador incluem tanto meios voláteis, quanto não volá-
teis, removíveis e não removíveis, implementados por qual-
quer método ou tecnologia de armazenamento de informação tal
como instruções que podem ser lidas por computador, estrutu-
ras de dados, módulos de programas e outros dados. Os meios
de armazenamento de computador incluem, mas não se limitam a
RAM, ROM, EEPROM, Memória Flash ou outras tecnologias de me-
mória, CD-ROM, DVD (disco digital versátil) ou outros arma-
zenamentos óticos em disco ou quaisquer outros meios, os
quais podem ser utilizados para armazenamento de informações
desejadas e que possam ser acessados por meio do computador
110. O meio de comunicação concretiza tipicamente instruções
que podem ser lidas por computador, estruturas de dados, mó-
dulos de programas ou outros dados em um sinal de dados mo-
dulados, tal como uma onda transportadora ou outros mecanis-
mos de transporte e inclui qualquer meio de fornecimento de
informação. O termo "sinal de dados modulado" significa um
sinal que possui uma ou mais de suas características ajusta-
das ou modificadas de tal maneira a codificar informações em
um sinal. À guisa de exemplo e não de limitação, o meio de
comunicação inclui meios em fio, tal como um rede conectado
ou uma conexão com fiação direta e meios de fio, tal como
acústicos, RF, infravermelho e outros meios sem fio. As com-



binações de quaisquer dos meios citados acima também deveriam ser incluídas no escopo dos meios que podem ser lidos por computador.

A memória de sistema 130 inclui meios de armazenamento de computador, em forma de memória volátil e/ou não volátil, tal como memória apenas de leitura (ROM) 131 e memória de acesso randômico (RAM) 132. Um sistema de alimentação de entrada /saída básico (BIOS), contendo as rotinas básicas que auxilia na a transferência de informações entre elementos contidos no computador 110, tal como durante a partida, ficam tipicamente armazenados em ROM 131. A RAM 132 contém, tipicamente, dados e/ou módulos de programas que podem ser imediatamente acessados e/ou se encontrem presente-mente em operação pela unidade de processamento 120. À guisa de exemplo e não de limitação, a Figura 1 ilustra um sistema de operação de dados 134, programas de aplicação 135. outros módulos de programa 136 e dados de programa 137.

O computador 110 também pode incluir outros meios de armazenamento de computador removíveis /não removíveis, voláteis / não voláteis. Apenas à guisa de exemplo, a Figura 1 ilustra um acionador de disco rígido 141, que lê a partir de ou escreve a um meio magnético não removível, não volátil, um acionador de disco magnético 151, que lê a partir de e escreve a um disco ótico removível, não volátil 156, tal como um CD ROM ou outro meio ótico. Outros meios de armazenamento de computador removíveis / não removíveis, voláteis / não voláteis que podem ser utilizados no exemplo de ambiente de operação incluem, mas não se limitam a fitas casse-



tes magnéticas, cartões de memória flash, discos digitais versáteis, fitas de vídeo digitais, RAM em estado sólido, ROM em estado sólido e similares. O acionador de disco rígido 141, tipicamente, fica conectado ao barramento de sistema 121, através de uma interface de memória não removível, tal como a interface 140 e o disco de memória magnético 151 e o acionador de disco ótico 155 ficam tipicamente conectados ao barramento de sistema 121, através de uma interface de memória removível, tal como a interface 150.

Os acionadores 150 e seus meios de armazenamento de computador associados, discutidos acima e ilustrados na Figura 1, provêm armazenamento de instruções que podem ser lidas por computador, estruturas de dados, módulos de programas e outros dados para o computador 110. Na Figura 1, por exemplo, o acionador de disco rígido 141 encontra-se ilustrado como sistema de operação de armazenamento 144, programas de aplicação 145, outros módulos de programas 146 e dados de programas 147. Observe-se que estes componentes podem tratar-se dos mesmos ou serem diferentes do sistema de operação 134, programas de aplicação 135, outros módulos de programas 136 e de dados de programa 137. Ao sistema de operação 144, aos programas de aplicação 145, aos outros módulos de programa 146 e aos dados de programa 147 são dados números diferentes aos ilustrados neste caso, no mínimo, estes sendo cópias diferentes. Um usuário pode entrar com comandos e informações no computador 110 através de dispositivos de alimentação de entrada tal como um teclado 162 e um dispositivo de apontamento 161, comumente denominado de mouse, es-



fera de rastreamento ou tecla de toque. Outros dispositivos de alimentação de entrada (não ilustrados) podem incluir microfone, joystick, game pad (teclas de jogos), disco de satélite, scanner ou similares. Estes e outros dispositivos de alimentação de entrada, muitas das vezes, ficam conectados à unidade de processamento 120 através de uma interface de alimentação de entrada de usuário 160, a qual fica acoplada ao barramento de sistema, mas pode ser conectada por outras estruturas de interface e barramento, tais como portais paralelos, portais de jogos ou um barramento universal em série. Um monitor 191 ou outros dispositivo de apresentação também fica conectado ao barramento de sistema 121 através de uma interface, tal como uma interface de vídeo 190. em adição ao monitor, os computadores também podem incluir outros dispositivos de alimentação de saída periféricos tais como alto-falantes 197 e impressora 196, os quais podem ser conectados através de uma interface de alimentação de saída de periféricos 195.

O computador 110 pode operar em um ambiente de rede, fazendo uso de conexões lógicas a um ou mais computadores remotos, tal como o computador remoto 180. O computador remoto 180 pode se tratar de outro computador pessoal, um servidor, um router, um PC de rede, um dispositivo peer ou outro nó comum de rede e, tipicamente, inclui muitos ou todos os elementos descritos acima com relação ao computador pessoal 110, embora tenha sido ilustrado apenas um dispositivo de armazenamento de memória 181 na Figura 1. As conexões lógicas ilustradas na Figura 1 incluem uma rede de área lo-



cal (LAN) 171 e uma rede de área ampla (WAN) 173, mas podem incluir também outros serviços de rede. Tais ambientes de rede tratam-se de lugar comum em escritórios, serviços de rede de computador de amplos empreendimentos, intranet e a Internet.

Quando utilizado em um ambiente de rede LAN, o computador pessoal 110 fica conectado ao LAN 171 através de uma interface de rede ou adaptador 170. Quando utilizado em um ambiente de rede WAN, o computador 110, tipicamente, inclui um modem 172 ou outro dispositivo para estabelecimento de comunicação através do WAN 173, tal como a Internet. O Modem 172, o qual pode ser interno ou externo, pode ficar conectado ao barramento de sistema 121 através de uma interface de alimentação de entrada de usuário 160 ou de outro mecanismo apropriado. Em um ambiente de rede, os módulos de programa ilustrados em relação ao computador pessoal 110 ou a partes do mesmo, podem ficar armazenados no dispositivo de armazenamento de memória remoto. À guisa de exemplo e não de limitação, a Figura 1 ilustra programas de aplicação remota 185 como residindo no dispositivo de memória 181. Deve ser observado que as conexões de rede tratam-se de exemplos e outros meios de estabelecimento de ligações de comunicação entre os computadores podem ser utilizados.

Na descrição que se segue, a invenção será descrita com referência aos atos e representações simbólicas de operações que são executadas por um ou mais computadores, a não ser que de alguma forma indicado. Assim sendo, deve ser entendido que tais atos e operações, as quais algumas das



vezes são denominadas como sendo executada por computador incluem a manipulação pela unidade de processamento do computador de sinais elétricos, representando dados em forma estruturada. Esta manipulação transforma os dados ou mantém os mesmos em locais no sistema de memória do computador, o qual reconfigura ou altera de alguma maneira a operação do computador de uma maneira bem conhecida pelos indivíduos versados na técnica. As estruturas de dados, onde os dados são mantidos são locais físicos da memória, que apresentam propriedades específicas, definidas pelo formato dos dados. No entanto, embora a invenção esteja sendo descrita no contexto precedente, este não se designa a ser limitante, uma vez que os indivíduos versados na técnica irão observar que vários atos e operações descritas a seguir também podem ser implementados no hardware.

Um ambiente de rede em que a arquitetura do barreira de proteção da presente invenção é utilizado, será descrito agora com referência à Figura 2. O rede trata-se de exemplo em natureza, uma vez que a arquitetura de barreira de proteção da presente invenção pode ser implementada em qualquer dispositivo de rede que se encontre acoplado a qualquer configuração de rede. O ambiente de rede inclui um rede privado 200 e um rede público 202. O rede privado 200 e o rede público 202 são de qualquer tipo apropriado, tal como Rede de Área Local (LANs) Rede de Área Ampla (WANs), intranets, a Internet ou quaisquer combinações dos mesmos.

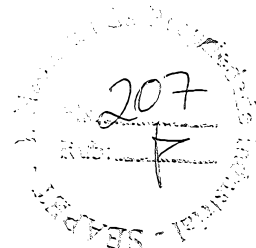
O ambiente de rede inclui uma pluralidade de dispositivos de rede 204, 206, 208, 210 e 212. Os dispositivos



de rede 204, 206 são acoplados ao rede privado 200. Os dispositivos de rede 210, 212 são acoplados ao rede público 202. O dispositivo de rede 208 é acoplado e provê uma interface tanto entre os serviços de rede privado 202, quanto público 202. Os dispositivos de rede são acoplados aos serviços de rede público e privado com uso de qualquer tecnologia apropriada, tal como Ethernet1394 ou 802.11(b). Os dispositivos de rede são adicionalmente implementados como qualquer dispositivo de computação apropriado, tal como computadores pessoais, servidores, dispositivos de manuseio manual, impressoras, comutadores, routers, pontes de ligação, repetidores ou similares.

O dispositivo de rede 208 inclui uma barreira de proteção 214 e um ou mais filtros 216. A barreira de proteção 214 trata-se de um módulo de programa ou de um conjunto de módulos de programa, implementados de acordo com a arquitetura da presente invenção, a qual examina pacotes de rede comutados entre os dispositivos de rede 204, 206, 208, acoplados ao rede privado 200 e os dispositivos de rede 210, 212, acoplados ao rede público 202. Segundo uma modalidade da invenção, a barreira de proteção 214 também examina pacotes de rede destinados de forma local enviados a partir dos e destinados aos dispositivos de rede no interior de um rede privado 200.

A barreira de proteção 214 é implementada no dispositivo de rede 208 com fins de proteção e controle do tráfego de rede, comutado entre o rede privado 200 e o rede público 202, denominado como uma barreira de proteção de bor-



da. Alternativamente, a barreira de proteção 214 é implemen-
tada e protege um único dispositivo de rede, tal como ilus-
trado no dispositivo de rede 210, denominado como barreira
de proteção hospedeiro. Uma barreira de proteção também é
5 capaz de ser implementada como um conjunto administrado de
forma central de barreiras de proteção hospedeiras e/ou bar-
reiras de proteção de borda, de maneira sincronizada, deno-
minado de barreira de proteção distribuída. O posicionamento
do dispositivo ou dos dispositivos de rede, implementando a
10 barreira de proteção 214, de preferência, são selecionados
de forma que a barreira de proteção 214 examine todo o trá-
fego de rede destinado aos dispositivos de rede que se supõe
que estes devam proteger.

Os filtros 216 são implementados como parte da
15 barreira de proteção 214. Alternativamente, os filtros 216
são implementados como uma parte de uma estrutura de dados
separada, que pode ser acessada pela barreira de proteção
214. A barreira de proteção 214 e os filtros 216 executam um
diretriz de barreira de proteção, designado a proteger os
20 dispositivos de rede 204, 206, 208 de ataques maliciosos o-
riginados a partir dos dispositivos de rede 210, 212, aco-
plados ao rede público. A barreira de proteção 214 ainda
provê uma funcionalidade adicional, tal como facilidade de
controle pelos pais, detecção de intrusão, cadastro de paco-
25 tes de rede e outras capacidades de funcionamento com base
em filtros adicionais.

Cada filtro 216 inclui um conjunto de condições de
filtro e uma ou mais ações associadas. As condições de fil-



tro incluem parâmetros e informações que podem ser analisadas ou obtidas de alguma forma a partir de pacotes de rede, tais como números de interfaces, endereços de hardware, endereços de serviços de rede, tipos de protocolo, números de portais e dados de carga de retribuição. Ações típicas incluem a permissão, isto é, permitem que os pacotes continue a travessia do rede e o bloqueio, isto é impedem a travessia adicional do rede pelos pacotes em descontinuidade.

A barreira de proteção 214 examina pacotes de rede que atravessam o rede quando recebidos no dispositivo de rede 208 e, por comparação dos parâmetros de pacote às condições de filtro, identifica um ou mais filtros que se conjugam. Um filtro que se conjuga resulta quando as condições de filtro se conjugam aos parâmetros de pacote. Os parâmetros de pacote, como as condições de filtro, incluem a informação que é analisada ou obtida de alguma outra maneira a partir dos pacotes. Quando a barreira de proteção identifica um filtro conjugado, as uma ou mais ações associadas às condições de filtro são executadas.

A Figura 3 ilustra uma vista geral de uma modalidade de arquitetura de barreira de proteção de acordo com a presente invenção. A arquitetura provê a capacidade de sujeitar pacotes a filtros em todas as camadas do empilhamento de rede. A barreira de proteção provê uma administração central das capacidades que permitem que filtros sejam adicionados e excluídos e que os conflitos de filtro sejam identificados e resolvidos. A arquitetura de barreira de proteção é extensível pelo fato das camadas de filtro são adicionadas



e excluídas conforme necessário, sendo expandidas para inclusão de capacidades de funcionamento especializadas além das ações de permissão e bloqueio. Embora a invenção esteja sendo descrita com referência específica a barreiras de proteção e a filtros de barreira de proteção, esta também é utilizada para facilitar e administrar outros filtros e diretrizes. Como exemplos específicos, a presente invenção é apropriada para facilitar e administrar filtros utilizados com conjunto de Qualidade de Serviço (QoS), Segurança de Protocolos de Internet (IPSec), bem como outros protocolos de codificação, autenticação e de administração chave.

A arquitetura de barreira de proteção inclui um processo de modo de usuário 250 e um processo de modo núcleo 252. O processo de modo de usuário 250 e o processo de modo núcleo 252 são executados como parte de um sistema de operação em um dispositivo de rede. Os indivíduos versados na técnica irão observar que o processo de modo de usuário 250 e o processo de modo núcleo 252 do sistema de operação incluem componentes adicionais que devido à simplificação não são ilustrados. Alternativamente, a arquitetura de barreira de proteção é executada como um todo ou em parte externamente ao sistema de operação como um ou mais módulos de programa ou diretrizes ou no interior de um único processo de sistema de operação.

O processo de modo núcleo 252 incluem um empilhamento de rede 254, um primeiro motor de barreira de proteção, denominado neste caso como motor de barreira de proteção núcleo 256 e chamadas opcionais 258. Coletivamente, o



processo de modo núcleo 252 implementa um diretriz de barreira de proteção estabelecido por meio da identificação de filtros conjugados para pacotes de rede, pacotes de processamento de acordo com protocolos conhecidos e execução de outras ações no pacote como designado pelos filtros conjugados.

O empilhamento de rede 254 compreende uma pluralidade de processos de camada, denominados neste caso como camadas, incluindo uma camada de jato de dados 268, uma camada de transporte 270, uma camada de rede 272 e uma camada de ligação 274. A arquitetura de barreira de proteção da presente invenção é extensível e camadas adicionais são dinamicamente adicionadas e excluídas quando necessário. Um exemplo de uma camada adicionada inclui uma camada de acesso de arquivo 276, implementada de acordo com um protocolo de Bloco Principal de Servidor (SMB). As camadas podem funcionar segundo atuação conjunta com outros módulos de programa, tal como o módulo de análise de Protocolo de Transferência Hiper Texto (HTTP) 278.

As camadas no empilhamento de rede 254 processam pacotes de rede internos e externos. Pacotes de rede externos são pacotes sendo transmitidos a partir do dispositivo de rede, que implementa a arquitetura de barreira de proteção sobre a rede. Pacotes internos são pacotes recebidos no dispositivo de rede, implementando a arquitetura de barreira de proteção. Como indicado pelas respectivas setas ilustradas na Figura 3, os pacotes internos atravessam o empilhamento de rede 254 a partir do fundo ao topo e os pacotes ex-

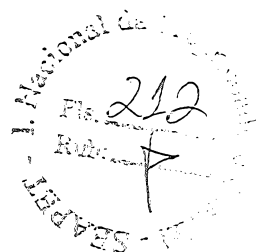


ternos atravessam o empilhamento de rede 254 a partir do topo ao fundo.

Pacotes de rede atravessam e são processados pelas camadas de rede em seqüência. De acordo com técnicas conhecidas, cada camada no empilhamento de rede 254 é capaz de receber pacotes a partir de uma camada ou processo prévio, processando o pacote de acordo com a especificação ou protocolo e enviando o pacote processo para a próxima camada ou processo. De acordo com a invenção, cada camada no empilhamento de rede 254 também mantém o contexto de pacote, passa este contexto de pacote à próxima camada, lança uma requisição de classificação ao motor de barreira de proteção núcleo 256 e efetua a ação no pacote de acordo com o diretriz do barreira de proteção.

O contexto de pacote trata-se de uma estrutura de dados que segue o pacote, partindo de camada a camada. Cada camada mantém o contexto por meio de adição, ao interior do contexto de estrutura de dados, de um conjunto de parâmetros, que a camada se encontra projetada para processar, por exemplo, a informação que a camada se encontra projetada a-para analisar a partir de pacotes internos, adicionar a pacotes externos ou derivar de alguma forma a partir dos conteúdos de pacote. Um exemplo de estrutura de dados utilizado para o contexto de pacote encontra-se descrito com referência à Figura 5.

Uma das operações executadas pelas camadas de empilhamento de rede 254 é invocar o motor da barreira de proteção núcleo por meio de emissão de requisições de classifi-



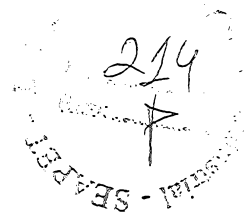
cação. A requisição de classificação trata-se de uma chamada por meio de uma camada no rede 254, requisitando que quaisquer filtros que se conjuguem ao pacote sejam identificados e quaisquer ações associadas, por exemplo, permissão ou bloqueio sejam retornadas à camada. A camada que lança a requisição de classificação é denominada neste caso de camada requisitante. Cada camada ainda efetua a ação no pacote que é retornado pelo motor de rede núcleo 256.

O motor de rede 256 inclui uma camada API 280, um conjunto de filtros instalados 282 e uma chamada API 284. O motor de barreira de proteção núcleo 256 efetua várias funções, incluindo (1) manutenção de um conjunto de filtros instalados 282, definindo a diretriz da barreira de proteção, (2) recebimento de requisições de classificação provenientes das camadas no empilhamento de rede 254, (3) identificação de um ou mais filtros que se conjugam, com base na requisição de classificação e (4) instrução da camada de requisição de quaisquer ações a serem efetuadas no pacote, com base nos filtros que se conjugam.

Cada filtro dentro o conjunto de filtros instalados inclui um conjunto de condições de filtro e uma ou mais ações associadas. Como descrito com referência à Figura 2, as condições de filtro identificam os pacotes de rede que se encontram sujeitos à ação de filtro associada. A ação especificada no conjunto de filtros instalados 282 inclui a permissão e o bloqueio. Uma funcionalidade adicional é adicionada através de chamadas opcionais 258. Um exemplo de forma dos filtros encontra-se descrito com referência à Figura 4.

Opcionalmente, as chamadas 258 são usadas para implementação de funcionalidade adicional além das ações de permissão e bloqueio. Uma chamada é executada quando o motor de barreira de proteção núcleo 256 identifica um filtro conjugado para o pacote que inclui, como ação associada, uma chamada a um dos módulos de chamada. O motor da barreira de

Opcionalmente, as chamadas 258 são usadas para implementação de funcionalidade adicional além das ações de permissão e bloqueio. Uma chamada é executada quando o motor de barreira de proteção núcleo 256 identifica um filtro conjugado para o pacote que inclui, como ação associada, uma chamada a um dos módulos de chamada. O motor da barreira de



proteção envia a requisição de classificação a um dentre os módulos de chamada, isto é pacote completo, parâmetros de camada e contexto de pacote, ao módulo de chamada, em conjunto com uma identificação do filtro conjugado, através da chamada API 284. Em uma modalidade da invenção, a arquitetura de barreira de proteção inclui um conjunto básico de chamadas 258. Chamadas adicionais, tais como camadas, são adicionadas conforme necessário à arquitetura de barreira de proteção extensível. Uma implementação específica da chamada API 284 encontra-se descrita com referência à Figura 6.

O processo de modo de usuário 250 inclui um segundo motor de barreira de proteção, denominado de motor de barreira de proteção de usuário 260 e um ou mais provedores de planos de ação 262, identificados como "PP1", "PP2" e "PP3". Os provedores de planos de ação 262 são processos que adicionam planos de ação de barreira de proteção, isto é, filtros instalados 282 à arquitetura de barreira de proteção. Qualquer processo pode ser utilizado para concretização desta tarefa. Um exemplo é um serviço de diretriz de legado IPSec (LIPS). O serviço de diretriz legado IPSec adiciona filtros definindo um tráfego de rede, que se supõe fazer uso de protocolos IPSec, tais como um Protocolo de Segurança por Encapsulamento (ESP) e Protocolo de Autenticação Superior (AH). Como exemplo específico, a diretriz de legado IPSec adiciona uma diretriz de barreira de proteção, indicando que todos pacotes internos não solicitados devem ser criptografados de acordo com o protocolo ESP. A diretriz ainda provê que quaisquer pacotes internos não solicitados em tecto ca-



lo, isto, um pacote não criptografado, deveriam ser bloqueados. Os provedores de diretriz 262 obtém o diretriz a partir de qualquer fonte apropriada, tal como dados em memória volátil e não volátil ou uma Interface Gráfica de Usuário (GUI), que permita que um administrador ou usuários de sistema façam uso direto do diretriz. O motor de barreira de proteção de usuário 260 converte o diretriz em um novo filtro, isto é, define o diretriz em termos de condições de filtro e ações associadas e adiciona o novo filtro ao conjunto de filtros instalados 282.

O motor de barreira de proteção de usuário 260 também executa arbitragem de filtro e funções de resolução de conflitos. Quando o provedor de diretriz 262 provê uma nova diretriz ao motor de barreira de proteção de modo de usuário 260, o motor de barreira de proteção de usuário determina se o novo filtro, resultante da nova diretriz, entra em conflito com quaisquer dentre os filtros instalados 282. Um exemplo de um método de identificação e resolução de conflitos que é apropriado para uso na estrutura da presente invenção encontra-se descrito na Aplicação de Patente Norte Americana, intitulada "Method for Managing network Filter Based Policy".

A arquitetura ainda inclui um motor de filtro API 266, que configura uma interface entre o motor de barreira de proteção de modo de usuário 260 e o motor de barreira de proteção núcleo 256. O motor de filtro API 266 provê um mecanismo para o motor de barreira de proteção de usuário 260 para adição de novos filtros a ou para remoção de filtros

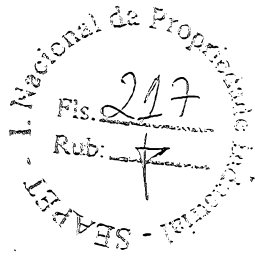


existentes a partir de um conjunto de filtros instalados 282 e para exame de filtros instalados 282, de forma que os conflitos de filtro possam ser detectados e resolvidos. Uma administração API 290, acessível pelos provedores de diretriz 5 262 também provê um mecanismo para adição e remoção de filtros da arquitetura.

O motor de barreira de proteção de modo de usuário 260 também inclui um módulo de filtro 294. O módulo de filtro 294 trata-se de uma instância do motor de barreira de 10 proteção núcleo 256 no modo de usuário 250. A instância do módulo de filtro 294 no motor de barreira de proteção de modo de usuário 260 permite que o motor de barreira de proteção de usuário 260 replique os serviços do motor de barreira de proteção núcleo 256 para um ou mais processo de modo de 15 usuário, denominado neste caso de camadas de modo de usuário 282. As camadas de modo de usuário 282 são adicionadas da mesma maneira que as camadas de modo de núcleo são criadas, devido ao fato de módulo de filtro 294 trata-se da instância de modo de usuário do motor de barreira de proteção núcleo 20 256, deve ser entendido que quaisquer funcionalidades descritas para o motor de barreira de proteção núcleo 256 também se aplicam ao módulo de filtro 294.

Um módulo chave API 288 provê uma interface entre o motor de diretriz de usuário 260 e o módulo chave 296. O 25 módulo chave API 288 é utilizado para sinalizar o módulo chave da necessidade de estabelecimento de SA.

A arquitetura de barreira de proteção inclui um diretriz de tempo de destituição 286 e uma diretriz seguro



contra falhas 264. A diretriz de tempo de boot 286 e a diretriz contra falhas 264 são implementados para prover um conjunto básico de filtros instalados 282, enquanto o sistema se encontra em estado de transição, tal como quando o dispositivo de rede é iniciado, por exemplo, ligado, reajustado ou é dada nova partida. Durante o estado de transição, existe um período latente durante o qual o modo de usuário 250 não está estabelecido e não é possível a instalação de filtros através de provedores de diretriz 262 e do motor de barreira de proteção de usuário 260. Como resultado, o dispositivo de rede fica vulnerável a ataques maliciosos a partir de, por exemplo, pacotes internos não solicitados. A diretriz de tempo de boot 286 reside no modo núcleo 252 e é carregado ao motor de barreira de proteção núcleo 256 anteriormente ao tempo de estabelecimento do modo de usuário 250. A diretriz de tempo de boot 296 provê o filtro básico instalado 282, que pode ser configurado para prover quaisquer condições de filtro e ações associadas para proteção do sistema, durante o estado de transição. Um exemplo de diretriz de boot 286 é um que bloqueia todos os pacotes de rede internos e externos.

Assim que o modo de usuário 250 foi estabelecido, o estado de transição continua, enquanto os provedores de diretriz 262 se encontram identificando planos de ação e tentando adicionar filtros instalados 282 no motor de barreira de proteção núcleo 256. Quando o modo de usuário 250 é estabelecido, o motor de barreira de proteção 260 carrega primeiro a diretriz seguro contra falhas 264 ao motor de



barreira de proteção núcleo 256, antes de qualquer diretriz ser aceito pelos provedores de planos de ação 262. A diretriz de tempo de boot 286 é então desativada. Do mesmo modo que a diretriz de tempo de boot 286, a diretriz segura contra falhas 264 trata-se de um filtro básico, projetado para proteger o sistema de ataques durante o estado de transição, quando os provedores de diretriz e o motor de barreira de proteção de usuário ainda se encontram no processo de instalação de filtros. Uma vez que os provedores de planos de ação 262 instalaram com sucesso seus respectivos filtros no dispositivo de rede, o plano seguro contra falhas 264 é desativado.

Com referência à Figura 4, o conjunto de filtros instalados 282 irá ser descrito agora. Cada filtro 31 do conjunto de filtros instalados 282 possui uma pluralidade de campos, incluindo uma ID (identidade) de filtro 312, um peso 314 e uma ou mais ações 316, um contexto de diretriz 328 e um conjunto de condições de filtro 318. A ID de filtro 312 provê uma identificação única do filtro. A ID de filtro é utilizada, por exemplo, como meio de um motor de barreira de proteção núcleo 256 retornar informações de filtro conjugado ao motor de barreira de proteção de usuário 260 e às chamadas 258. Em uma modalidade da invenção, o filtro se designa a uma dentre as camadas do empilhamento de rede 254. A ID de filtro 312 é utilizada pelo motor de barreira de proteção núcleo 256 para rastrear qual dentre os filtros se designa a qual camada.



O campo de peso 314 inclui um valor que identifica prioridade do filtro 310. Quanto mais alto o valor no campo de filtro 314, tanto mais alta a prioridade do filtro. A prioridade de filtro determina a ordem pela qual são aplicados filtros conjugados ao pacote por meio do motor de barreira de proteção núcleo 256.

Em uma modalidade da invenção, o filtro com a prioridade mais alta, isto é, o valor de peso mais alto, é aplicado em primeiro lugar, a seguir é aplicado o filtro de prioridade subsequente e assim por diante, até que um filtro conjugado com uma ação de término seja encontrado. As ações de término se encontram descritas com maiores detalhes abaixo. Uma vez que o filtro conjugado com a ação de término é aplicado, o motor de barreira de proteção núcleo 256 para com a aplicação de filtros conjugados. Assim sendo, a ação 316 especificada pelos filtros conjugados de prioridade mais baixa não são recebidos no pacote após a ação de término ter sido aplicada. Alternativamente, o motor de barreira de proteção 256 identifica um único filtro conjugado e retorna ao conjunto de ações a partir do único filtro conjugado.

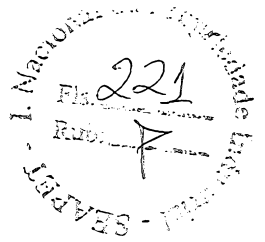
O conjunto de condições de filtro 318 determina se um pacote se conjuga ao filtro 310. Cada condição de filtro 318 inclui um tipo 320 e uma ID de camada: ID de campo 324. O tipo 320 define o comprimento e a quantidade de variáveis incluídas nos dados correspondentes 322. A arquitetura provê tipo de variáveis conhecidas previamente definidas, tais como Byte, Short, Long, 8 Bytes, String, Internet Protocol Address - versão 4 (IPv4), Internet Protocol Address - versão



6 (IPv6), IPv4 Address plus Mask, IPv6 Address plus Mask e Address Range.

O campo de dados inclui dados conjugando-se ao tipo. Por exemplo, caso o tipo seja IPv4 Address, um valor aceitável para o campo de dados 322 é um número de 32 - bit, na faixa de 00.00.00 a 255.255.255.255, como expresso na notação decimal tracejada. Em alguns exemplos, o tipo 320 provê valores múltiplos no campo de dados 322. Os tipos Address Range, IPv4 Address plus Mask e IPv6 Address plus Mask permitem dois valores de endereços IP, definindo um início e um término de faixa de endereços IP. Para uma máxima flexibilidade, a arquitetura ainda permite tipos definidos para usuários. Alternativamente, tipos adicionais são manualmente adicionados à arquitetura de sistema.

A ID de camada: ID de campo 324 é utilizado para identificação de uma camada originária e um parâmetro proveniente da camada originária, respectivamente. A camada originária e o parâmetro proveniente da camada originária definem parâmetros de pacote, isto é, parâmetros de camada e contexto de pacotes, aos quais os dados 322 são comparados quando da identificação de filtros conjugados. A camada originária identifica uma camada no empilhamento de rede. O parâmetro proveniente da camada originária identifica um parâmetro específico, associado à camada originária. Um exemplo específico encontra-se ilustrado pela condição de filtro 326. O tipo é IPv4, com isso indicando que os dados 322 se tratam de um endereço IP 32 - bit. A ID de camada é "IP", representando que o numero 32 bit trata-se de um IP, isto é



um parâmetros de camada, rede. O campo ID é "Src IP Addr", o qual no exemplo representa uma fonte de endereço IP. O endereço IP, provido no campo de dados 322 é "123.3.2.1", indicando que qualquer pacote com a fonte endereço IP atende a
5 condição de filtro, comisso conjugando o filtro. O filtro 310 pode incluir múltiplas condições de filtro318, e neste caso, o pacote conjuga o filtro 310 apenas em caso de todas as condições de filtro serem atendidas.

A ação 316 no filtro 310 é de permissão, bloqueio
10 ou chamada. Caso a ação 324 no filtro 310 seja de permissão ou de bloqueio e o pacote se conjugar ao filtro 310, a ação de permissão ou bloqueio é retornada à camada requisitante por meio do motor de barreira de proteção núcleo 256. Caso a ação seja de chamada, o motor de barreira de proteção núcleo
15 256 emite sua própria requisição de classificação que inclui um pacote completo, parâmetros de pacote, contexto e identificação do filtro conjugado ao módulo de chamada designado 258. O módulo de chamada 258 efetua sai função programada no pacote, por exemplo, detecção de intrusão. A chamada pode
20 retornar uma ação (permissão ou bloqueio) ao motor de barreira de proteção núcleo, o qual por sua vez, supre a ação à camada requisitante. A chamada também é capaz de manter o contexto de pacote que, do mesmo modo, é retornado à camada requisitante através do motor de barreira de proteção núcleo
25 256. A ação também pode trata-se de um valor, tal como um valor nulo, indicando que não existe nenhuma diretriz de barreira de proteção.



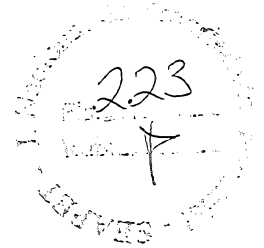
Ações são projetadas como sendo de término ou não de término. Caso a ação seja de término, assim que a ação é identificada para um pacote, a ação é retornada à camada requisitante e nenhum filtro adicional é aplicado ao pacote.

- 5 Caso a ação não seja de término, levantamentos adicionais de filtro continuam até uma ação de término para o pacote ser identificada. Como falha, permissão e bloqueio são designadas ações de término.

O contexto de diretriz 328 é utilizado para armazenamento de planos de ação diferentes da diretriz de barreira de proteção, tal como diretriz de segurança ou diretriz QOS. O contexto de diretriz trata-se de qualquer estrutura de dados. Por exemplo, o contexto de diretriz é um número de 64 bit, que é interpretado pelo processo que adiciona o contexto de diretriz. O contexto de diretriz e/ou a ação podem ser de valor nulo.

A Figura 5 ilustra um exemplo de estrutura de dados 330, utilizada para o contexto de pacote que é mantido por e passado às camadas no empilhamento de rede 254 e nos módulos de chamada 258. O contexto de pacote 330 segue o pacote de rede interno e externo à medida que atravessa as camadas e inclui uma ou mais entradas, sinalizadas 336 - 340. Cada entrada inclui uma ID de camada: ID de campo 322 e um valor correspondente 334.

25 A ID de camada: ID de campo 332 apresenta o mesmo significado da ID de camada; ID de campo 324, provido como parte das condições de filtro 318 nos filtros 310 (Figura 4). Especificamente, a ID de camada: ID de campo identifica



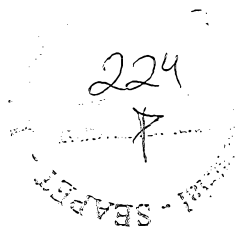
as camadas originárias e o parâmetro de camada a partir da camada originária para os dados no campo de valor 334. O campo de valor 334 inclui um parâmetro de camada específico.

Como exemplo específico, a entrada 336 inclui a ID de camada: ID de campo 332 "NDIS: Src. MAC Addr." "NDIS" representa uma implementação da Especificação de Interface de Acionador de Rede da camada de ligação 274 (Figura 1). "Src.MAC.addr" representa uma fonte de endereço MAC. Assim sendo, a ID de camada: ID de campo 332 indica que os dados no campo de valor 334 são uma fonte de endereços MAC, que foi processada pela camada (ligação) NDIS. O campo de valor 334 inclui a fonte real de endereços MAC, a qual no exemplo trata-se de "00.08.74.4F.22.E5", como expresso na notação hexadecimal.

Como um segundo exemplo, a entrada 338 possui uma ID de camada: ID de campo 332 de "NDIS:IF No". Novamente, isso identifica a camada como NDIS, mas neste caso indicando parâmetros como IF No" representando um número de interface como um parâmetro NDIS específico. O campo de valor 334 inclui o número de interface real, o qual nesse caso é de 2.

Como um terceiro exemplo, a entrada 340 possui uma ID de camada: ID de campo 332 de "IP:DstIP Addr". O "IP" representa a camada de rede utilizando o protocolo IP e "DdtIP Addr" representando o destino do endereço IP como parâmetro de camada IP. O campo de valor 334 inclui o destino real de endereço IP de "123.3.2.1".

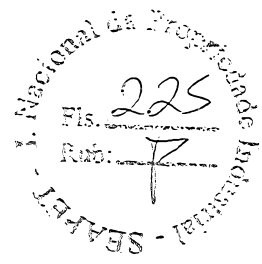
Tendo sido descrita a arquitetura de barreira de proteção subjacente da presente invenção, a atenção é volta-



da a interfaces funcionais do sistema e exemplos de métodos que são executados com a utilização da arquitetura de barreira de proteção subjacente, descrita neste caso. As interfaces funcionais os implementadas como uma pluralidade de interfaces de aplicação de programação (APIs). As APIs incluem a camada API 280, a chamada API 284, o motor de filtro API 266 e o módulo chave PAI 288, com ilustrado nas Figuras 6 e 7.

A camada API 280 facilita o intercâmbio de dados entre cada uma dentre as camadas no empilhamento de rede 254 e no motor de barreira de proteção núcleo 256. Como ilustrado, a camada API 280 inclui um método de Classificação 402, um método de adição de camada 404 e um método de exclusão de camada 406. A camada API 280 também pode ser executada dentro o módulo de filtro 294 para facilitar a comunicação entre as camadas de modo de usuário e o módulo de filtro 294.

O método de classificação 402 e utilizado pela camada requisitante para enviar parâmetros de camada, o pacote como recebido pela requisição e o contexto de pacote do motor de barreira de proteção núcleo 256. O motor de barreira de proteção núcleo 256 compara (1) os parâmetros de camada provenientes da camada de requisição e (2) as entradas de contexto de pacote nas condições de filtro 318 em cada filtro 310, designado à camada requisitante para identificação de filtros conjugados. O exemplo a seguir trata-se de um exemplo de implementação do método de classificação. Deve ser entendido que os seguintes métodos são descritos como recebendo ou retornando valores de dados. De acordo com técnicas



de programação conhecidas, os métodos podem usar apontadores para os valores de dados, ao invés de valores de dados reais.

NTSTATUS

WFPClassify

```

5  (
    IN ULONG                                LayerId,
    IN WFP_INCOMING_VALUES*                 pInFixedValues,
    IN WFP_INCOMING_CONTEXT_VALUE*         pInContext,
    PVOID                                    pPacket,
10  OUT WFP_ACTION_TYPE                     pActionType,
    OUT UINT64*                             pOutContext
    );

```

Onde a seguir se caracterizam os parâmetros mencionados.

15 LayerId identifica a camada de rede emitindo a requisição de classificação, isto é, a camada requisitante. Com referência à Figura 3, a ID de camada identifica a camada como a camada de fluxo de dados 268, a camada de transporte 270, a camada de rede 272 ou a camada de ligação 274.

20 Outras camadas 268 são válidas em caso de adição ao sistema. Por exemplo, caso a camada SMB 276 seja adicionada, esta possui sua identificação própria única. A arquitetura de barreira de proteção da presente invenção ainda permite implementações de múltiplos protocolos em uma camada no empilhamento de rede 254. Por exemplo, o empilhamento possui duas camadas de transporte 270, uma primeira camada de transporte fazendo uso do protocolo TCP e uma segunda camada de transporte utilizando o protocolo UDP. As camadas de modo de

usuário 282 podem também ser adicionadas, com isso configurando uma camada válida.

pInFixedValues inclui um subconjunto de parâmetros de camada, processados pela camada de requisição. Os pInFixedValues, em conjunto com as entradas de contexto de pacote são comparadas às condições de filtro para determinar se o pacote se conjuga ao filtro. Um exemplo de parâmetros de camada falhos incluídos nos pInFixedValues para cada camada se encontram identificados na Tabela A, abaixo. Deve ser entendido que os seguintes valores tratam-se de exemplos apenas, como uma vantagem da arquitetura sendo que esta permite que as camadas façam uso de quaisquer valores acessíveis à camada.

Tabela A

Camada	Parâmetros de Camada Falhos
Camada de ligação (link)	Fonte e destino de endereços MAC; Número de interface
Camada de rede	Fonte e destino de endereços IP; Tipo de protocolo; Tipo de endereço local
Camada de transporte	Fonte e destino de números de portas
Aplicação	Carga e retribuição de protocolo de camada de aplicação decifrada



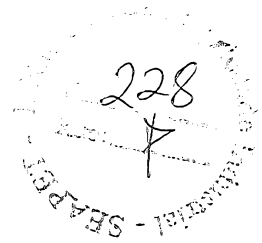
pInContext inclui a estrutura de contexto de dados 330 (fig5), como recebida pela camada requisitante. O contexto de pacote é utilizado pelo motor de barreira de proteção de núcleo 256, em conjunto com parâmetros de camada para
5 identificação de pacotes conjugados.

pPacket inclui a totalidade do pacote como recebido pela camada requisitante. O pPacket não é utilizado pelo motor de barreira de proteção núcleo 256 para identificação de filtros conjugados. Como previamente escrito, o motor de
10 barreira de proteção de núcleo 256 faz uso de pInFixedValues e pInContext para identificação de filtros conjugados. O pPacket é incluído no método e classificação, de forma que o motor de barreira de proteção de núcleo 256 possa enviar um ou mas módulos de chamada 258, identificados como ação 316
15 nos filtros conjugados.

pActionType inclui a ação 316 que é retornada à camada requisitante. A ação 316 retornada é de permitir ou bloquear como identificado no filtro conjugado ou de módulos de chamada, executados pelo filtro conjugado.

20 pOutContext inclui a diretriz de dados de contexto. Como previamente descrito, o contexto de plano de dados é utilizado para facilitar planos de ação de rede associados com IPSec, QOS e quaisquer outros planos de ação com base em filtros diferentes de barreira de proteção.

25 Os métodos de adição de camada 404 e a exclusão de camada 406 são utilizados para adição e remoção de uma camada da arquitetura de barreira de proteção, respectivamente.



A seguir se apresenta uma modalidade de exemplo de método de adição de camada 404.

NTSTATUS

AddExtensionLayer (OUT PULONG pLayerId);

5 Onde a seguir se caracterizam os parâmetros mencionados.

pLayered trata-se de um valor de identificação de camada único, retornado pela camada sendo adicionada, isto é, a camada que executa o método de adição de camada.

10 A seguir apresenta-se um exemplo de modalidade de método de exclusão de camada 406.

NTSTATUS

RemoveExtensionLayer (ULONG LayerId);

15 Onde a seguir se caracterizam os parâmetros mencionados.

LayerId identifica a camada sendo removida, isto é, a camada que executa o método de exclusão de camada.

20 A chamada API 284 facilita o intercâmbio de dados entre o motor de barreira de proteção de núcleo 256 e as chamadas 258. A chamada API 284 inclui um método de classificação 410, um método de notificação 412, um método de registro de chamada 414 e um método de retirada de registro de chamada 416. O método de classificação 410 da chamada API 284 é similar ao método de classificação 402 da camada API 280, com exceção de também incluir dados de filtro conjugados. A seguir se apresenta uma modalidade de exemplo do método de classificação 410, utilizado para executar uma chamada.

25



```
typedef NSTATUS (*WFP_CALLOUT_CLASSIFY_FN)
```

```
(
    IN const WFP_INCOMING-VALUES* fixedValues,
    IN WFP_INCOMING_CONTEXT_VALUE* wfpContext,
5    IN VOID* packet (pacote),
    IN WFP_FILTER* matchedFilter,
    OUT WFP_ACTION_TYPE* action (ação),
    OUT UINT64* outContext
);
```

10 Onde a seguir se caracterizam os parâmetros mencionados.

fixedValues inclui os parâmetros de camada enviados a partir da camada requisitante. Os fixedValues tratam-se dos mesmos dados providos pela camada requisitante em dados pInFixedValues, enviados como parte do método de classificação 402 na camada API 280.

wfpContext inclui a estrutura de contexto e dados 330 (Figura 5). Estes dados são os mesmos que os enviados pela camada de requisição no pInContext enviado como parte do método de classificação 402 na camada API 280.

packet inclui a totalidade de pacote, como recebido pela camada requisitante. Estes dados tratam-se dos mesmos enviados pela camada de requisição no pPacket, enviado como parte do método de classificação 410 da chamada API 284.

25 matchedFilter identifica o filtro requisitando da chamada. Tipicamente, o filtro conjugado se identifica pela ID de filtro 312 do filtro conjugado 310, iniciando o método de classificação 410 da chamada API 284.



PActionType inclui a ação que é retornada ao motor de barreira de proteção de núcleo 256, a partir da chamada 258. Caso a pActionType seja permitir ou bloquear, esta é retornada à camada de requisição como pActionType, retornada

5 pela camada API 280. A chamada também pode retornar uma ação contínua que instrui o motor de barreira de proteção de núcleo 256 a continuar aplicando filtros conjugados ao pacote.

pOutContext inclui os dados de contexto de diretriz, tais como segurança ou dados de diretriz QOS.

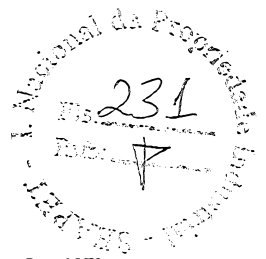
10 O método de notificação 412 é utilizado para notificar uma chamada, quando um filtro 310 é adicionado ou excluído do conjunto de filtros instalados 282, que identifica como uma de suas ações 316, o módulo de chamada 258. A notificação provê a chamada com uma oportunidade de execução de qualquer

15 ação necessária, tal como alocação ou retirada de alocação de amortecimentos que irão ser utilizados pela chamada 258, quando esta é executada pelo motor de barreira de proteção de núcleo 256. A seguir se apresenta uma modalidade de exemplo de método de notificação (Notify).

20 typedef NSTATUS (*WFP_CALOU_NOTIFY_FN)
(
 IN WFP_NOTIFY_ENUM notify (notificação),
 IN WFP_FILTER* filter (filtro)
);

25 Onde a seguir se caracterizam os parâmetros mencionados.

notify inclui um valor numérico que indica se o filtro está sendo adicionado ou excluído. Por exemplo, um



valor de 1 indica que o filtro está sendo adicionado e um valor de 2 indicando que o filtro está sendo excluído.

filter identificando o filtro sendo adicionado ou excluído através de um valor único. Isso pode ser concreti-
 5 zado pelo provimento da ID de filtro 312, incluída como parte do filtro 310.

O método de registro de chamada 414 e o método de retirada de registro de chamada 416 os utilizados para adicionar e remover módulos de chamada, respectivamente. Um exemplo de modalidade para concretização do método de registro 414 sendo apresentado a seguir.

NSTATUS WfpRegistreCallout

(

IN const GUI* calloutId,
 15 IN const WFP_CALLOUT* callout (chamada),
 IN const SECURITY_DESCRIPTOR* sd
);

Onde a seguir se caracterizam os parâmetros mencionados.

20 callout ID provê uma identificação única para o módulo de registro de chamada.

callout provê qualquer informação específica de chamada, tal como um nome de serviço de acionador, nome de dispositivo e apontadores para as funções de classificação e
 25 notificação de chamadas.

sd provê uma descrição de segurança para a chamada. A descrição de segurança provê uma lista de controle de



acesso (ACL), que especifica qual processos podem modificar e excluir chamadas.

Segue um exemplo de modalidade de método de retirada de registro de chamada 416.

5 NSTATUS WfpDeregistrationCallout

(

IN const GUID* calloutId

);

10 Onde a seguir se caracterizam os parâmetros mencionados.

calloutId trata-se da identidade única da chamada a ser removida.

O motor de filtro API 266 facilita intercâmbio de dados entre o motor de barreira de proteção de modo de usuário 260 e o motor de barreira de proteção de núcleo 256. Como ilustrado, a administração API 266 inclui um método de
15 adição de filtro 418, um método de exclusão de filtro 420 e um método de camada de enumeração 422. Os métodos de motor de filtro API 266 também são providos como parte da administração
20 ção API 290.

Os métodos de adição de filtro 418 e a exclusão de filtro 418 são utilizados para adicionar um novo filtro a um conjunto de filtros instalados 282 e para exclusão de um filtro existente do conjunto de filtros instalados 282, respectivamente. Se segue um exemplo de modalidade de método de
25 adição de filtro 418.

NSTATUS

AddFilterToLayer



```
(
    ULONG                LayerId,
    WFO_FILTER*          pFilter
);
```

5 Onde a seguir se caracterizam os parâmetros mencionados.

LayerId identifica a camada à qual o filtro se encontra designado.

pFilter trata-se de um filtro 310 sendo adicionado
10 ao conjunto de filtros instalados 282.

A seguir, um exemplo de modalidade para concretização do método de exclusão de filtro 420.

NSTATUS

DeleteFilterFromLayer

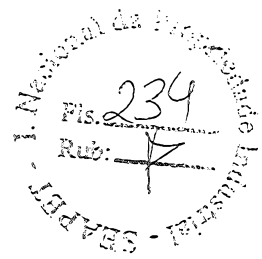
```
15 (
    ULONG                LayerId,
    ULONG                FilterId
):
```

20 Onde a seguir se caracterizam os parâmetros mencionados.

LayerId identifica a camada à qual o filtro se encontra designado.

FilterId é o filtro sendo excluído a partir do conjunto de filtros instalados.

25 O método de enumeração de camada 422 provê um mecanismo para o motor de barreira de proteção de usuário 260 identificar todos os filtros conjugados a u conjunto de critérios. Isso permite que o motor de filtro API identifique



filtros conflitantes para arbitragem de filtros e resolução de conflitos. A seguir, um exemplo de modalidade de método de enumeração de camada 422.

LayerStartEnum

```

5  (
        PWFP_ENUM_TEMPLATE          pEnumTemplate,
        OUT PULONG                  pMatchCount,
        OUT PWFP_ENUM_HANDLE        pEnumHandle
    );

```

10 Em que a seguir se caracterizam os parâmetros mencionados.

 pEnumTemplate inclui uma estrutura de dados, definindo os filtros que deveriam ser retornados. Por exemplo, este inclui parâmetros aos quais as condições de filtro devem se conjugar para que o filtro seja retornado.

 pMatchCount inclui o número de conjugações de filtro com base no pEnumTemplate especificado.

 pEnumHandle inclui um referência às entradas de filtros conjugados.

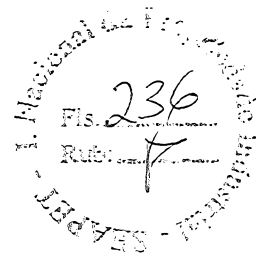
20 o módulo chave API 288 provê uma interface entre os módulos chave e o motor de barreira de proteção de usuário 260. O módulo chave API 288 inclui um método IPsec AS Acquire 430, um método Expire Notify 432 e uma método IPSEC AS Acquire Complete 434, um método Keying Module Register 436, um método Keying Module Deregister 438, um método IPSEC Inbound Get SPI 440, um método ADD Inbound AS 444, um método Add Outbound SA 446, um método Inbound SA Expire 448 e um método Keying Module Initiate 350. O módulo chave API 288



também pode incluir os métodos previamente descritos para a camada API, tais como método de classificação 402, um método de adição de camada 404 e um método de exclusão de camada 406.

O módulo chave API é utilizado para facilitar o uso de protocolos de segurança, tais como os definidos por IPSec, utilizados por um computador em início e um computador em resposta. Ipsec inclui protocolos tais como Ah e ESP. O protocolo ESP trata-se de um protocolo de autenticação e criptografia, o qual faz uso de mecanismos criptográficos para prover integridade, autenticação de fonte e confidencialidade de dados. O protocolo AH trata-se de um protocolo de autenticação, que faz uso de uma assinatura remodelada no pacote encabeçador para validar a integridade dos dados de pacote e a autenticidade de quem enviou.

Um protocolo IKE provê um método para um computador em início e um computador em resposta para negociar ajustes de segurança utilizados com os protocolos AH e ESP. Os ajustes de segurança negociados configuram uma estrutura de dados, denominada uma associação de segurança (SA). A SA define parâmetros tais como um algoritmo de autenticação, algoritmo de criptografia, chaves e vida útil das chaves, utilizadas pelo ESP ou AH para proteção dos conteúdos de um pacote IP. Devido ao ESP e AH exigirem uma SA estabelecida, uma negociação IKE é executada antes que os protocolos ESP ou AH sejam utilizados pelos computador em inicio e em resposta. Uma determinada SA se identifica pelo valor conhecido como Índice de Parâmetro de Segurança (SPI).



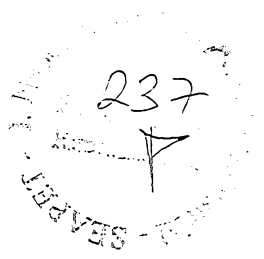
Cada um dentre os computadores em início e em resposta incluem um acionador IPsec, que determina, com base na diretriz IPsec, se os dados enviados entre os computadores em início e em resposta necessitam ser criptografados ou autenticados. A diretriz IPsec é um conjunto de filtros, os quais definem como o dispositivo de rede faz uso do IPsec e inclui listas de filtros, métodos de autenticação e outras informações. Em uma modalidade da invenção, a diretriz IPsec se define por meio de filtros incluídos em um conjunto de filtros instalados no motor de barreira de proteção núcleo ou no módulo de filtro 294. A diretriz que se aplica a um pacote se identifica por meio de invocação da camada de módulo chave, a qual utiliza o método de classificação para identificar filtros conjugados.

O método IPsec SA Acquire 430 é chamado pelo motor de barreira de proteção 260 (através de um proxy de cliente) para passar uma aquisição de acionador ou uma requisição de início externa à camada de modo de usuário 282, implementada como camada de módulo chave. A camada de módulo chave retorna esta chamada e efetua a negociação de forma não sincronizada. Assim que a camada de módulo de chave completou a negociação, a camada de módulo chave chama o método IPsec SA Acquire Complete 343 para notificar o motor de diretriz de usuário 260 que a negociação foi completada. A seguir, uma modalidade de exemplo do método IPsec SA Acquire.

```
typedef WIN32_ERR (*PROCESS_IPSEC_SA_ACQUIRE0)
```

```
(
```

```
IN FWP_IPSEC_ACQUIRE_CONTEXT0 ipsecContext,
```



```
IN const FWP_IPSEC_ACQUIRE0* acquire,  
IN FWP_IPSEC_SPI                inboundSAspi  
);
```

Em que, a seguir os parâmetros mencionados se en-
5 contram caracterizados.

ipsecContext trata-se de um manuseio para ligação
do acquire com a SA sendo adicionada.

acquire inclui a informação necessária para nego-
ciação da SA, de acordo com protocolos conhecidos, tais como
10 IKE.

inboundSAspi inclui uma SPI que é utilizada para
uma SA interna.

O método de notificação de expiração (Expire No-
tify) 432 é chamado para passar uma notificação de expiração
15 à camada de módulo chave que adicionou a SA interna. A se-
guir, um exemplo de modalidade para concretização do método
de notificação de expiração (Expire Notify).

```
typedef VOID (*PROCESS_IPSEC_SA_EXPIRE0)  
(  
20 IN const FW_IPSEC_SA_EXPIRE_NOTIFY0* expireNotify  
);
```

Onde a seguir se caracterizam os parâmetros men-
cionados.

expireNotify contém informação de identificação da
25 expiração da SA. Por exemplo, no caso de uma SA externa, se
provê SPI.

O método IPsec SA Acquire Complete 434 é chamado
por uma camada de módulo chave para fechamento do contexto



do motor de barreira de proteção de usuário, após este ter terminado a negociação e adicionado todas as SAs ou após este ter encontrado um erro. Após este método ter sido executado, a camada de módulo chave não reutiliza o IPSEC Context para qualquer outro método API. A seguir, um exemplo de modalidade de método IPsec SA Acquire Complete.

WIN32_ERR

FwpISecSAAcquireComplete0

(

```

10         IN FWPM_ENGINE_HANDLE          engineHandle,
           IN FWP_IPSEC_ACQUIRE_CONTEXT0 ipsecContext,
           IN const FWP_IPSEC_NEGOTIATION-STATUS0* status
    );

```

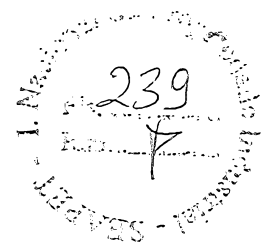
Onde a seguir se caracterizam os parâmetros mencionados.

engineHandle provê um manuseio ao motor de barreira de proteção de usuário 260.

ipsecContext é o contexto passado pelo motor de barreira de proteção de usuário com o método IPsec Acquire.

status provê estado e outros detalhes da negociação SA. O estado é retornado pelo motor de barreira de proteção de usuário 260, caso a aquisição foi iniciada externamente, através de FwpKeyingModelInitiated0.

O método de registro de módulo chave 436 é chamado por uma camada de módulo chave para registra com o motor de barreira de proteção de usuário 260 e passar seus apontadores de função. A seguir, um exemplo de modalidade para concretização do método de registro de módulo chave.



WIN32_ERR

FwpKeyingModuleRegister0

```
(  
    IN FWPM_ENGINE_HANDLE      engineHandle,  
5    IN const GUID*            keyingModuleID,  
    IN const FWP_KEYING_MODULE_INFO* keymodInfo  
);
```

Onde a seguir se caracterizam os parâmetros mencionados.

10 engineHandle provê o manuseio para motor de barreira de proteção de usuário 260.

keyingModuleID é uma ID única para a camada de módulo chave.

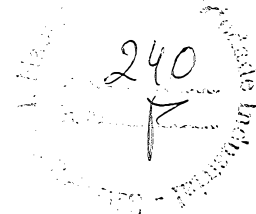
keymodInfo inclui uma informação de registro a
15 respeito da camada de módulo chave, tal como apontadores para as processo das funções IPsec SA Acquire e processo das funções IPsec SA Expire functions.

O método de retirada de registro de módulo chave
438 é chamado pela camada de módulo chave para retirada de
20 registro do módulo chave a partir do motor de barreira de proteção de usuário 260. A seguir, um exemplo de modalidade para concretização do método de retirada de registro de módulo chave.

WIN32_ERR

25 FwpKeyingModuleDeregister0

```
(  
    IN FWPM_ENGINE_HANDLE      engineHandle,  
    IN const GUID*            keyingModuleID
```



);

Onde a seguir se caracterizam os parâmetros mencionados.

engineHandle é o manuseio do motor de barreira de proteção de usuário 260.

keyingModuleID é uma ID única para a camada de módulo chave.

O método IPsec Inbound Get SPI 440 é chamado pela camada de módulo chave para obter SPI para uma nova SA interna. O método IPsec Inbound Get SPI 440 é, de forma geral, utilizado quando a camada de módulo chave é executada em um dispositivo de resposta de rede. A seguir, um exemplo de modalidade para concretização do método IPsec Inbound Get SPI.

WIN32_ERR
15 FwpIPsecSAInboundGetSpi0

(

IN FWPM_ENGINE_HANDLE engineHandle,

IN const FWP_IPSEC_TRAFFIC0* ipsecTrafficDescription,

20 IN const FWP_IPSEC_UDP_ENCAP0* udpEncapInfo,

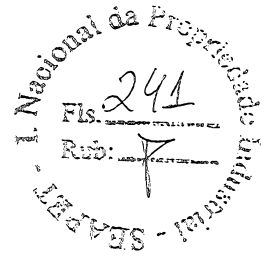
OUT FWP_IPSEC_SPI* inboundSpi

);

Onde a seguir se caracterizam os parâmetros mencionados.

25 engineHandle é o manuseio do motor de barreira de proteção de usuário 260.

ipsecTrafficDescription é uma descrição 5 - tuple para criação de uma SA interna, dissimulada. 5 -tuple inclui



endereços de fonte e destino, portais de fonte e destino e tipo de protocolo de camada de transporte.

udpEncapInfo trata-se de dados encapsulados UDP para criação de SA dissimulada. Um encapsulamento UDP trata-se de um método conhecido de encaixe de um pacote formatado de acordo com um protocolo de segurança em um pacote UDP não criptografado.

inboundSpi é um SPI para uma SA interna.

O método de adição de SA interna 444 é chamado pela camada de módulo chave para adição de uma SA interna, isto é, atualização da dissimulação SA. O motor de barreira de proteção de usuário 260 faz uso do SPI na SA para mapear esta chamada a seu estado interno e para baixar da SA ao acionador IPsec. A seguir, um exemplo de modalidade para concretização do método de adição de SA interna.

WI32_ERR

FwpIPSecSAInboundAdd0

(

IN FWPM_ENGINE_HANDLE engineHandle,

IN const FWP_IPSEC_SA_STRUCT0* inboundSA

);

Onde a seguir se caracterizam os parâmetros mencionados.

engineHandle é o manuseio do motor de barreira de proteção de usuário 260.

inboundSA inclui a SA interna.

O método de adição de SA externo 446 é chamado pela camada de módulo chave para adição de um SA externo. O



motor de barreira de proteção de usuário faz uso de um parâmetro SPI interno para mapear esta chamada a seu estado interno e baixar a SA ao acionador IPsec. A seguir, um exemplo de modalidade para constituição método de adição de SA

5 externo.

WIN32_ERR

FwpIPSecSAOutboundAdd0

(

IN FWPM_ENGINE_HANDLE engineHandle,

10 IN FWP_IPSEC_SPI inboundSpi,

IN const FWP_IPSEC_SA_STUCT0* outboundSA

Onde a seguir se caracterizam os parâmetros mencionados.

engineHandle é o manuseio do motor de barreira de

15 proteção de usuário 260.

inboundSpi é um SPI para uma SA interna, com a qual uma SA externa forma um par.

outboundSA inclui a SA externa.

O método de expiração de SA interna 448 é chamado

20 pela camada de módulo chave para expirar a SA interna, que foi previamente adicionada. A seguir, um exemplo de modalidade para formação do método de expiração de SA interna 448.

WIN32_ERR

FwpIPSecSAInboundExpire0

25 (

IN FWPM_ENGINE_HANDLE engineHandle

IN const FWP_IPSEC_SA_EXPIRE0* expire



Onde a seguir se caracterizam os parâmetros mencionados.

engineHandle é o manuseio do motor de barreira de proteção de usuário 260.

5 expire inclui dados para a expiração da SA.

O método de início do módulo chave 450 é chamado pela aplicação externa conhecida, similar a RAS, Winstock API e similares para dar início à camada de módulo chave e ajustar as SAs, antes do início de uma aplicação dar início
 10 ao envio de seu tráfego de rede. O motor de barreira de proteção 260 pendente a chamada RCP de forma não sincronizada, obtém o SPI a partir do acionador IPsec e passa a aquisição ao módulo chave apropriado. Assim que a camada de módulo chave chamar FwpIPsecAcquireComplete0, o motor de barreira
 15 de proteção de usuário completa o RPC não sincronizado com estado de negociação. A seguir, um exemplo de modalidade de concretização do método de início do módulo chave.

WIN32_ERR

FwpKeyingModuleInitiate0

```

20 (
    IN FWPM_ENGINE_HANDLE          engineHandle,
    IN const FWP_IPSEC_SA_ACQUIRE0* acquire,
    IN HANDLE                      waitEvent,
    OUT FWP_IPSEC_NEGOTIATION_STATUS0* negotiation-
25 Status
  
```

Onde a seguir se caracterizam os parâmetros mencionados.



engineHandle é o manuseio do motor de barreira de proteção de usuário 260.

acquire inclui dados necessários para negociação de uma SA.

5 waitEvent trata-se de um manuseio para um evento que é engatilhado quando o estado de negociação é disponibilizado. Caso um cliente, isto é, uma aplicação de chamada interna, não está interessada na espera da compleição de uma negociação, esta pode ajustar o parâmetro a ZERO. Interna-
10 mente, o proxy cliente passa opcionalmente este evento ao RPC e requisita o mesmo a ajustar o evento assim que o RPC não sincronizado se completar.

negotiationStatus inclui um resultado de negociação. O negotiationStatus é ZERO caso o waitEvent seja ZERO.
15 De outro modo, o negotiationStatus permanece válido até que o witEvent seja engatilhado.

A Figura 8 ilustra os métodos utilizados pelas camadas no empilhamento de rede 254, de acordo com a presente invenção. O método ilustrado na Figura 8 também pode ser u-
20 tilizado por uma ou mais camadas de modo de usuário 282 em conjunto com módulo de filtro 294 e ao motor de barreira de proteção de usuário 260.

Cada camada é capaz de uma pluralidade de funções, incluindo pacotes de processamento de rede, emissão de re-
25 quisições de classificação ao motor de barreira de proteção núcleo 256 e administração de contexto de pacote. Em uma modalidade da invenção, as funções são executadas em cada um dentre as camadas de um preenchimento 460, 462, 464, 466,



instalados no empilhamento do rede 254. Alternativamente, a funcionalidade é diretamente concretizada no interior das camadas individuais, sem necessidade de preenchimentos.

O empilhamento de rede 254 inclui a camada de fluxo de dados 268, a camada de transporte 270, a camada de rede 272 e a camada de ligação 274. Para fins de ilustração da presente invenção, a camada de ligação 274 encontra-se implementada como um acionadorr NDIS, a camada de rede 272 encontra-se implementada como uma camada IP, a camada de transporte 270 encontra-se implementada como uma camada TCPe a camada de fluxo de dados 268 encontra-se implementada como uma camada HTTP. Deve ser entendido que as camadas podem ser implementadas de acordo com qualquer protocolo. Por exemplo, a camada de transporte também acomoda um protocolo de diagrama de dados de usuário (UDP). A camada de aplicação suporta o protocolo de transferência de arquivo (FTP), Remote Procedure Call (RPC), protocolo de transferência de correspondência simples (SMTP), bloqueio principal de servidor (SMB), etc. Como previamente descrito, as camadas adicionais podem ser adicionadas à arquitetura e as camadas podem ser excluídas. Por exemplo, as camadas são adicionadas e excluídas com uso de métodos de adição de camada e de exclusão de camadas, descritos com referência à Figura 6.

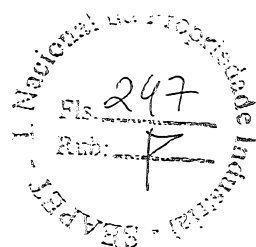
Um pacote de rede 468 (a) - (d), ilustra o pacote de rede, à medida que este atravessa é processado pelas camadas no empilhamento de rede 254. Caso o pacote 468 (a) - (d) seja um pacote interno, este atravessa o empilhamento de rede a partir do fundo até o topo. Caso o pacote 468 (a) -



(d) seja um pacote externo, este atravessa o empilhamento de rede a partir do topo até o fundo. Embora tais processamentos são bem conhecidos, estes encontram-se brevemente descritos para fins de ilustração da presente invenção.

5 Considerando-se que uma aplicação sendo executada em um dispositivo de rede, tal como um navegador de rede, dê início a uma requisição para conteúdos de uma página de rede, localizada em outros dispositivo de rede, a aplicação lança a requisição à camada de fluxo de dados 268. No exemplo,
10 o fluxo de dados 264 formata a requisição de acordo com o protocolo HTTP e envia a requisição à camada de transporte no pacote 468 (a). A camada de transporte 270 recebe o pacote 468 (a). a camada de transporte 270, a qual implementa o protocolo TCP, posiciona os dados em um ou mais pacotes, e
15 cada pacote fica provido com um encabeçador TCP. O encabeçador TCP inclui informações tais como portais de fonte e de destino, tipo de protocolo, isto é TCP, números de sequência, sinais e somatórios de exame. A camada de transporte envia então o pacote, sinalizado 468 (b) à camada de rede.

20 A camada de rede implementa o protocolo IP e encapsula do dados no interior do encabeçador IP, que inclui os endereços de fonte e de destino IP, sinais, somatórios de exame e outras informações conhecidas. O encabeçador IP também indica se o pacote é fragmentado. Um pacote é fragmentado,
25 quando a dimensão do pacote IP excede uma dimensão de unidade máxima de transmissão (MTU) para a tecnologia de rede utilizada para transmitir o pacote. Por exemplo, a tecnologia Ethernet especifica que a MTU seja de 1500 bytes. Caso



o comprimento de pacote IP exceda a MTU, este é fragmentado em dois ou mais pacotes IP, cada um com seu próprio encabeçador IP, todos de comprimento igual ou menor em comparação ao MTU.

5 Em uma modalidade da invenção, a camada de rede é dividida em primeira e segunda camadas. A camada final, denominada de camada fragmentada, processa os fragmentos de pacote IP. A segunda camada, denominada de camada completamente constituída processa os pacotes IP completos, por exemplo, antes dos pacotes IP externos serem fragmentados e
10 após os pacotes IP internos serem novamente estruturados em um único pacote IP. Seguindo-se um processamento de camada de rede e possível fragmentação, o pacote 468 © é enviado à camada de ligação 274. A camada de ligação 274 ainda empaco-
15 te os dados por meio de provimento de um encabeçador MAC, com endereços de fonte e destino MAC, bem como outras informações. O pacote é então enviado a um cartão de interface de rede (NIC), onde é fisicamente transmitido ao rede.

 Pacotes internos são processados de maneira recí-
20 proca. O pacote 468 (d) é recebido pelo NIC e enviado à camada de ligação 274. O encabeçador MAC é removido e o pacote 468 © é enviado à camada de rede, onde fragmentos de pacote IP são reestruturados, caso necessário e o encabeçador IP é analisado. A camada de rede envia então o pacote 468 (b) à
25 camada de transporte, onde o encabeçador TCP é removido e caso o fluxo de dados tenha sido enviado em uma pluralidade de pacotes TCP, o fluxo de dados é reestruturado. Finalmente, o fluxo de dados 468 (a) é enviado à camada de fluxo de



dados 268, onde os dados são decifrados pelo protocolo de aplicação, neste caso o protocolo HTTP.

Para cada pacote externo, as camadas do empilhamento de rede mantém o contexto de pacote 470 (a) - (c). para cada pacote interno, o empilhamento de rede mantém o contexto de pacote 470 (a) - (c). O contexto de pacote segue cada pacote, à medida que este atravessa as camadas de rede. O contexto de pacote também é passado, podendo ser modificado, às chamadas 258 (Figura 1).

10 O contexto de pacote é atualizado à medida que os pacotes são processados em cada camada. Cada camada adiciona seus parâmetros de camada ao contexto de pacote, com isso provendo esta informação à camadas ou processos subsequentes. Como ilustrado, a camada de ligação 274 adiciona endereços de fonte e destino MAC e um número de interface a um
15 pacote interno, como ilustrado pelo contexto 472 (a). Este contexto é recebido pela camada de rede 272, a qual adiciona endereços de fonte e destino IP, como ilustrado pelo contexto 472 (b). A camada de transporte 266 recebe o contexto e
20 adiciona os números de portais, como ilustrado pelo contexto 472 (a).

Um processo similar ocorre para o contexto 470 (a) - (c), associado aos pacotes externos. A camada de fluxo de dados 268 adiciona informações tais como um endereço URL
25 proveniente do pacote de carregamento de retribuição, como ilustrado pelo contexto 470 (a), a camada de transporte 270 adicionando ainda números de porta de fonte e destino, como ilustrado pelo contexto 470 (b) e a camada de rede adicio-



nando endereços de fonte e destino IP, como ilustrado pelo contexto 470 (c).

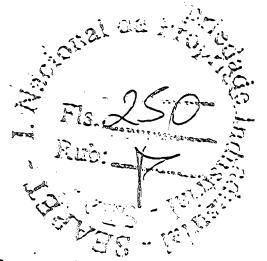
Deve ser entendido que cada camada pode adicionar qualquer informação de contexto, disponível àquela camada.

5 Tipicamente, isso inclui qualquer informação de designação de camada a processo, isto é, adição ou análise ou outra derivação dos pacotes.

Quando o pacote e seu contexto correspondente são recebidos em cada camada, as funções de camada funcionam como uma camada requisitante, por meio da identificação dos parâmetros de camada e envio de uma requisição de classificação, sinalizada 474. A requisição de classificação 474 inclui os parâmetros de camada 476, o contexto de pacote 478, recebido da camada anterior e o pacote completo 480. Um exemplo de método utilizado para emitir a requisição de classificação é o método de classificação 402, descrito com referência à camada.

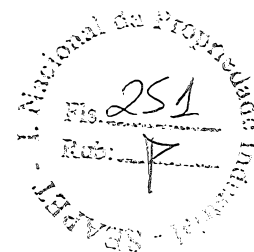
API 280 (Figura 7).

Em resposta a cada requisição de classificação, o motor de barreira de proteção 256 compara os parâmetros de camada 476 e o contexto de pacote 478 às condições de filtro 318 (Figura 4) dos filtros designados à camada requisitante. O motor de barreira de proteção 256 envia uma resposta sinalizada 482, com a ação 484 a partir do filtros conjugados 310, com o peso mais alto 314 à camada requisitante. O motor de barreira de proteção 256 também retorna o contexto de diretiva 482. Caso não sejam identificados filtros conjugados pelo motor de barreira de proteção 256, este motor de bar-



reira de proteção 256 notifica a camada requisitante da não existência de filtros conjugados. O motor de barreira de proteção 256 continua a identificar filtros conjugados até que um filtro conjugado designe uma ação de término, isto é, permissão ou bloqueio ou até que todos os filtros designados à camada requisitante tenham sido examinados, o que acontecer em primeiro lugar. Alternativamente, o motor de barreira de proteção 256 identifica todas as conjugações e retorna as ações à camada requisitante em uma única resposta.

10 Com ilustrado, de forma geral, a identificação de parâmetros de camada é efetuada com parte do processamento normal de camada, efetuado por meio de camadas do empilhamento de rede 254. Uma análise adicional do pacote não se faz necessária, com isso minimizando o impacto no desempenho do sistema. Além disso, devido ao fato das camadas atuarem em conjunto para manutenção do contexto de pacote, se torna possível ao motor de barreira de proteção 256 comparar as condições de filtro com os parâmetros de pacote base camadas que normalmente não apresentam acesso a estes parâmetros de pacote. Por exemplo, a camada de rede 268 recebe o contexto para pacote internos a partir da camada de ligação 274, que inclui endereços de fonte e de destino MAC. Devido ao fato da camada de rede 272 emitir requisições de classificação com parâmetros de camada de rede, por exemplo, endereços de fonte e de destino IP, bem como de contexto de pacote, o motor de barreira de proteção 256 pode filtrar em ambos os endereços IP e MAC na camada de rede 272, mesmo embora os endereços Mac, de forma geral, não se disponibilizem.



Com referência à Figura 9, será descrito agora um exemplo de conjunto de módulos de chamada 258, incluído na arquitetura de barreira de proteção da presente invenção. Os módulos de chamada 258 incluem chamadas de contexto HTTP 500, detecção de intrusão 502, chamada IPsec 504 e chamada de cadastro 506. A chamada API 284 forma a interface entre o motor de barreira de proteção de núcleo 256 e as chamadas 258.

A chamada de contexto HTTP 500 mantém um cache 508 de endereços URL aceitáveis ou alternativamente inaceitáveis. A chamada de contexto HTTP 508 acessa periodicamente um servidor 510 em interface com a rede pública, que mantém endereços URL e classifica os mesmos como aceitáveis ou inaceitáveis. Quando o motor de barreira de proteção de núcleo 256 executa a chamada de contexto HTTP, esta chamada examina o pacote, decifra os endereços URL, caso necessário e determina se são aceitáveis com base na informação oculta 508. A chamada HTTP retorna então a permissão, como ação 316 caso o endereço URL seja aceitável e bloqueia caso o endereço URL seja inaceitável ao motor de barreira de proteção núcleo 256, que por sua vez retorna a ação 316 à camada requisitante através da camada API 280. A chamada de contexto HTTP é útil na implementação de funções de controle pelos pais.

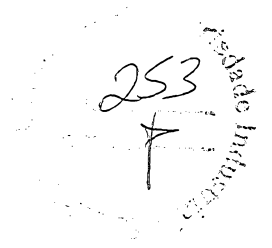
A chamada de detecção de intrusão 502 faz uso de algoritmos e técnicas disponíveis para exame do pacote, com fins de identificação de índices de vírus ou de outros pacotes de alguma forma suspeitos. Caso seja detectado um pacote suspeito, uma ação 316 do bloco é retornada. Como exemplo de



um pacote suspeito há o pacote em que os sinais nos encabeçadores IP e TCP encontram-se ajustadas a um valor de um. O pacote é suspeito devido ao mesmo nunca ser válido e poder indicar uma assinatura de ataque. Alternativamente, a chamada de intrusão de detecção 502 modifica o contexto de pacote para sinalizar a natureza suspeita do pacote, com isso deferindo a decisão de bloqueio do pacote às camadas subsequentes no empilhamento de rede.

A chamada IPsec 504 se designa a determinar se foi aplicado um protocolo de segurança apropriado ao pacote. A chamada IPsec 504 se comunica com um processo IPsec e determina, com base na diretriz Ipsec, se o pacote deveria se sujeitar ao processamento Ipsec. Caso positivo, a chamada IPsec 504 verifica, com base no contexto de pacote, se o pacote foi de fato sujeito ao processamento IPsec. Caso o pacote tenha devesse ser processado de acordo com IPsec, mas não o foi, por exemplo este pacote encontrando-se em texto claro, uma ação de bloqueio é retornada. Caso o pacote tenha se submetido ao processamento IPsec, a chamada IPsec verifica se foi aplicada uma ação de segurança SA apropriada. Caso a SA apropriada não tenha sido aplicada, a chamada retorna uma ação 316 de bloqueio, caso contrário uma ação 316 de permissão é retornada. Alternativamente, a chamada IPsec modifica o contexto de pacote, deferindo uma decisão de bloqueio ou de permissão para permissão a uma camada subsequente.

A chamada de cadastro 506 é utilizada para salvar informações pertinentes ao pacote, tais como pacote completo para utilização posterior. Tal utilização posterior, por e-



xemplo, pode se tratar de diagnósticos para determinado tráfego de rede, o qual não se encontra atravessando o rede inesperadamente devido ao fato de estar sendo bloqueado ou para rastreamento de ataques maliciosos no sistema.

5 Todas as referências citadas neste caso, encontram-se incluídas neste caso, incorporadas em sua totalidade, à guisa de referência.

 Em vista das muitas modalidades disponíveis, à
quais os princípios desta invenção podem ser aplicados, deve
10 ser reconhecido que a modalidade descrita neste caso em relação aos desenhos das Figuras, se propõe a ser apenas ilustrativo e não deveria ser tomado como limitação do escopo da invenção. por exemplo, os indivíduos versados na técnica irão reconhecer que os elementos da modalidade ilustrada em
15 software, pode ser implementada em hardware e vice-versa ou que a modalidade ilustrada pode ser modificada em disposição e detalhe, sem haver afastamento do espírito da invenção. assim sendo, a invenção como descrita neste caso, contempla todas estas modalidades, como incluídas no escopo das reivindicações a seguir e equivalentes das mesmas.
20

REIVINDICAÇÕES

1. Sistema de computador para implementar uma estrutura de firewall para proporcionar filtragem em camadas múltiplas de um pacote, compreendendo:

- 5 um conjunto de processadores de camada (268,270,272,274), cada processador de camada sendo capaz de processar parâmetros de camada (476) para o pacote associado ao processador de camada e cada processador de camada sendo ainda capaz de emitir uma requisição de
- 10 classificação (474) que inclua os parâmetros de camada;
- um primeiro mecanismo de firewall (256) de um processo no modo kernel (252);
- um segundo mecanismo de firewall (260) de um processo de modo de usuário (250); e
- 15 uma API de mecanismo de filtro (266);
- em que o primeiro mecanismo de firewall (256) inclui:
- uma interface de camada (280) para receber os parâmetros da primeira camada a partir de um processador
- 20 de camada requisitante e para retornar uma ação (484) à camada requisitante, o processador de camada requisitante sendo um dos conjuntos de processadores de camada;
- um conjunto de filtros instalados (282); e
- um componente de consulta para identificar pelo
- 25 menos um filtro conjugado a partir do conjunto de filtros instalados e identificar, a partir do filtro conjugado, a ação a ser retornada pela interface de camada;
- em que a API de mecanismo de filtro (266) é

adaptada para proporcionar um mecanismo para o segundo mecanismo de firewall para adicionar novos filtros ou remover filtros existentes do conjunto de filtros instalados e examinar os filtros instalados;

5 **caracterizado pelo fato de que** o segundo mecanismo de firewall (260) incluir um módulo de filtro (294) que é uma instância do primeiro mecanismo de firewall (256), o módulo de filtro (294) permitindo ao segundo mecanismo de firewall replicar serviços do primeiro mecanismo de firewall (256) para um ou mais processos de ca-
10 mada de modo de usuário; e

 em que a camada requisitante recebe o contexto de pacote (478) incluindo parâmetros de segunda camada a partir de um processador de camada anterior e a requisi-
15 ção de classificação inclui o contexto de pacote, em que o componente de pesquisa ainda utiliza o contexto de pacote para identificar pelo menos um filtro conjugado.

 2. Sistema, de acordo com a reivindicação 1, **ca-**
20 **racterizado pelo fato de que** o mecanismo de firewall ainda compreende uma interface de chamada para enviar os parâmetros da primeira camada, o contexto de pacote e o pacote de uma chamada, em que a chamada analisa o pacote e retorna a ação ao mecanismo de firewall.

 3. Sistema, de acordo com a reivindicação 1, **ca-**
25 **racterizado pelo fato de que** a camada requisitante passa o contexto de pacote para uma próxima camada.

 4. Sistema, de acordo com a reivindicação 1, **ca-**
 racterizado pelo fato de que ainda compreende um provedor

de diretriz para estabelecer um novo filtro, e

em que o segundo mecanismo de firewall é adaptado para adicionar o novo filtro ao conjunto de filtros instalados.

5 5. Sistema, de acordo com a reivindicação 1, **caracterizado pelo fato de que** cada um dos filtros instalados compreende:

uma condição de filtro incluindo um par de dados de tipo, o tipo definindo a dimensão dos dados e os dados
10 incluindo um parâmetro de filtro, e
a ação.

6. Sistema, de acordo com a reivindicação 5, **caracterizado pelo fato de que** o parâmetro de filtro inclui um intervalo de valores.

15 7. Sistema, de acordo com a reivindicação 5, **caracterizado pelo fato de que** o filtro compreende um fator de ponderação que define uma prioridade de filtro.

8. Sistema, de acordo com a reivindicação 1, **caracterizado pelo fato de que** o conjunto de processadores
20 de camada ainda compreende:

uma camada de ligação tendo parâmetros de camada e incluindo um número de interface e endereços MAC de origem e destino;

uma camada de rede tendo parâmetros de camada e
25 incluindo um endereço IP de origem e destino;

uma camada de transporte tendo parâmetros de camada e incluindo portas de origem e destino; e

uma camada de aplicação tendo parâmetros de ca-

mada e incluindo um fluxo de dados.

9. Sistema, de acordo com a reivindicação 8, **caracterizado pelo fato de que** a camada de rede é ainda dividida em uma camada fragmentada e em uma camada completamente estruturada, a camada fragmentada processando fragmentos de pacotes IP e a camada completamente estruturada processando pacotes IP completos, cada uma dentre a camada fragmentada e a camada completamente estruturada emitindo a requisição de classificação.

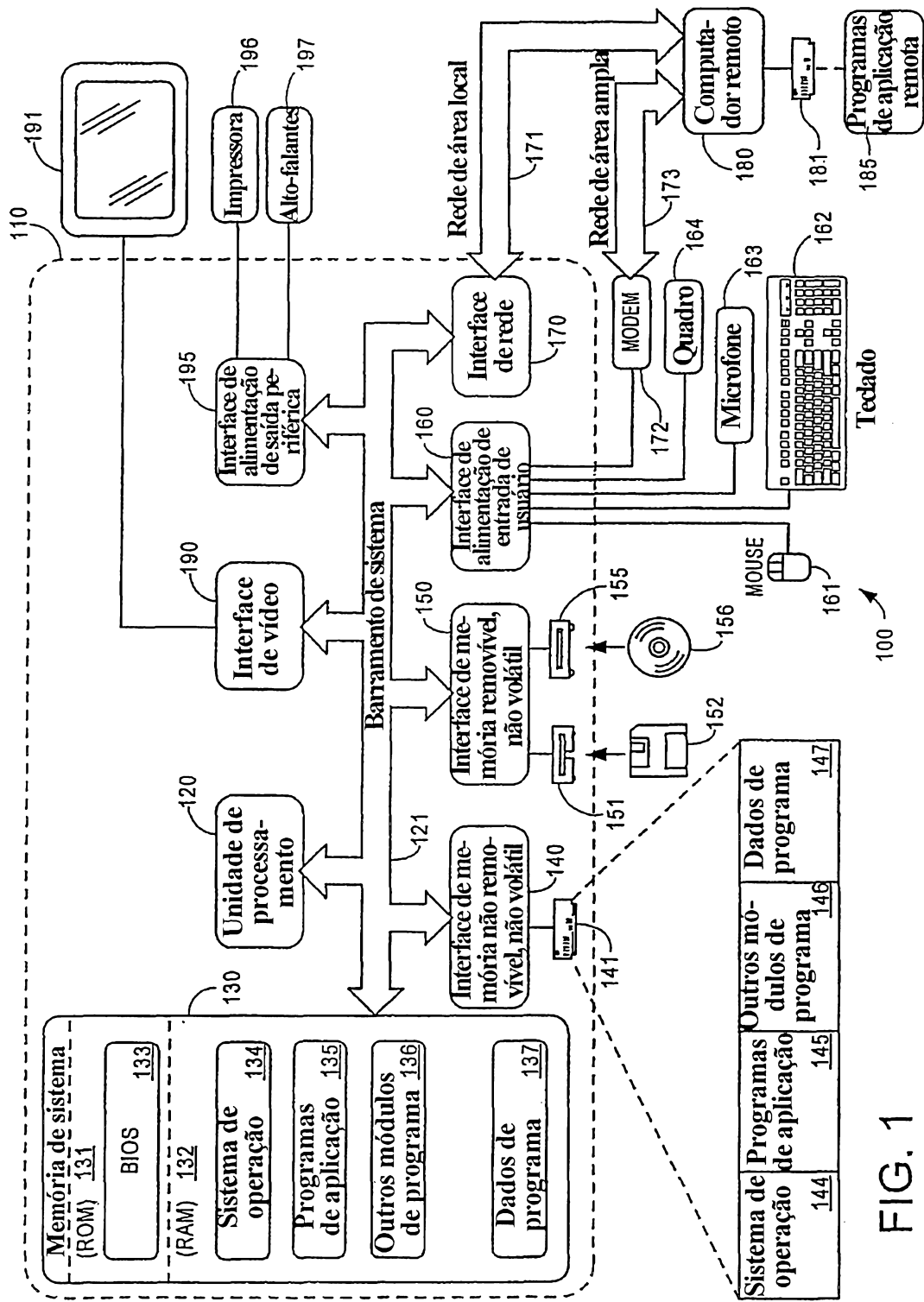


FIG. 1

165

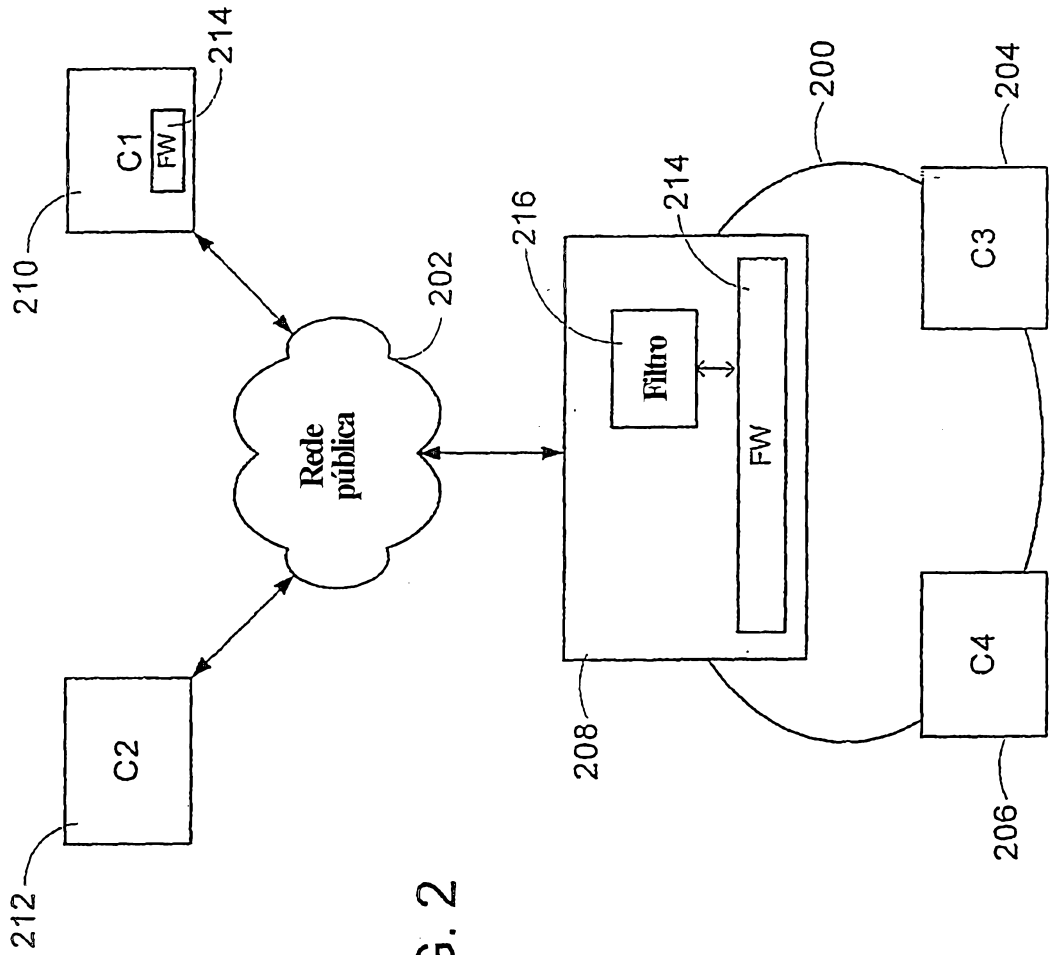


FIG. 2

FIG. 4

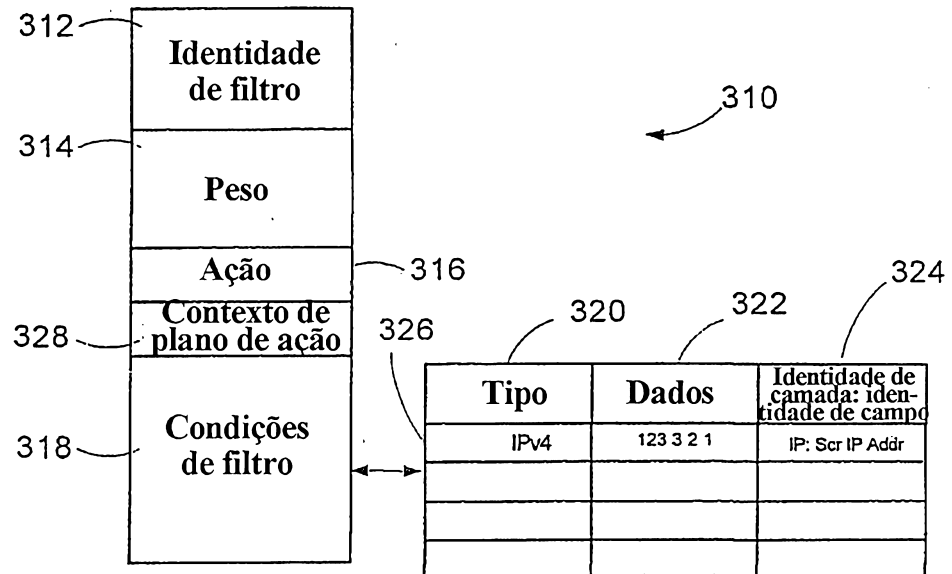


FIG. 5

332	334
Identidade de camada: identidade de campo	Valor
336 NDIS:Src MAC Addr.	00.80.74.4F.22.E5
338 NDIS:IF No.	2
340 IP:Dst IP Addr.	123.3.2.1

330

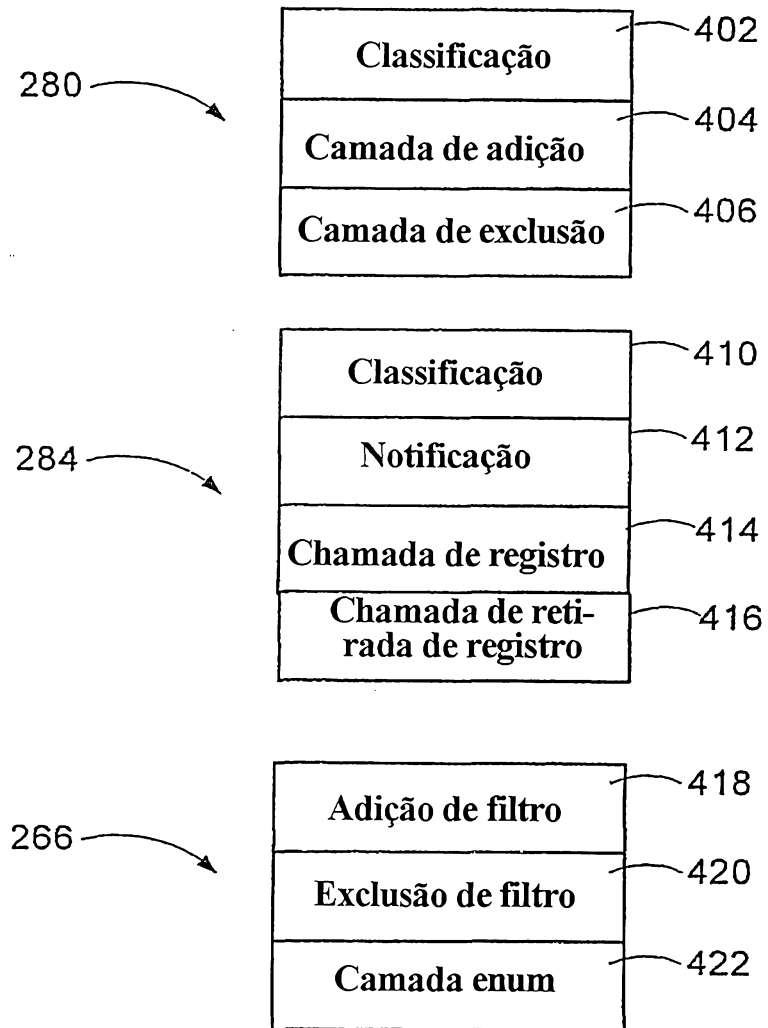


FIG. 6

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

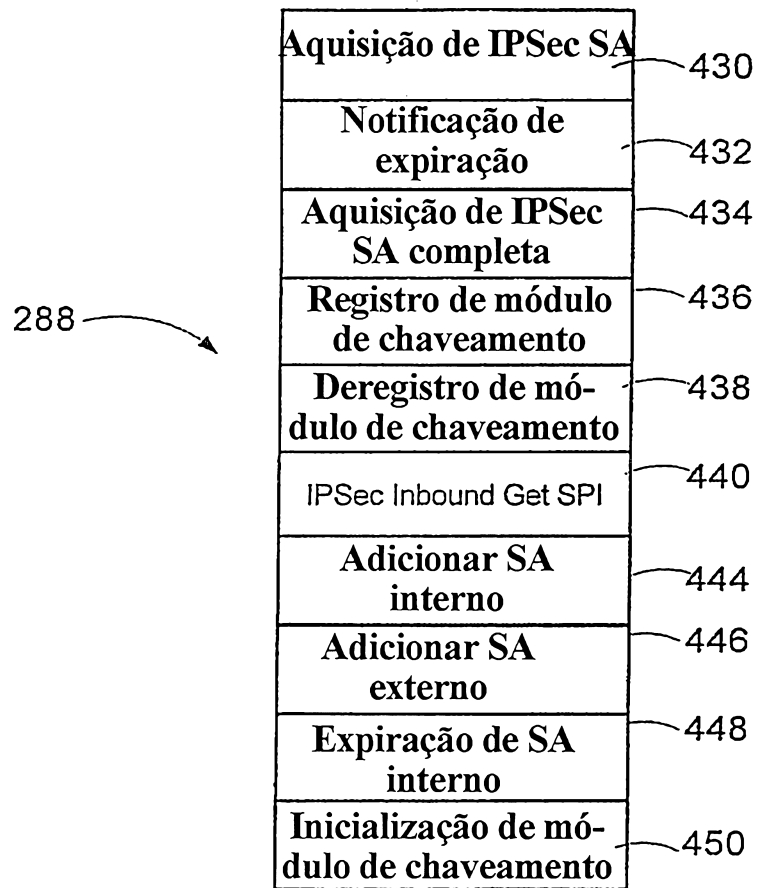


FIG. 7

FIG. 8

