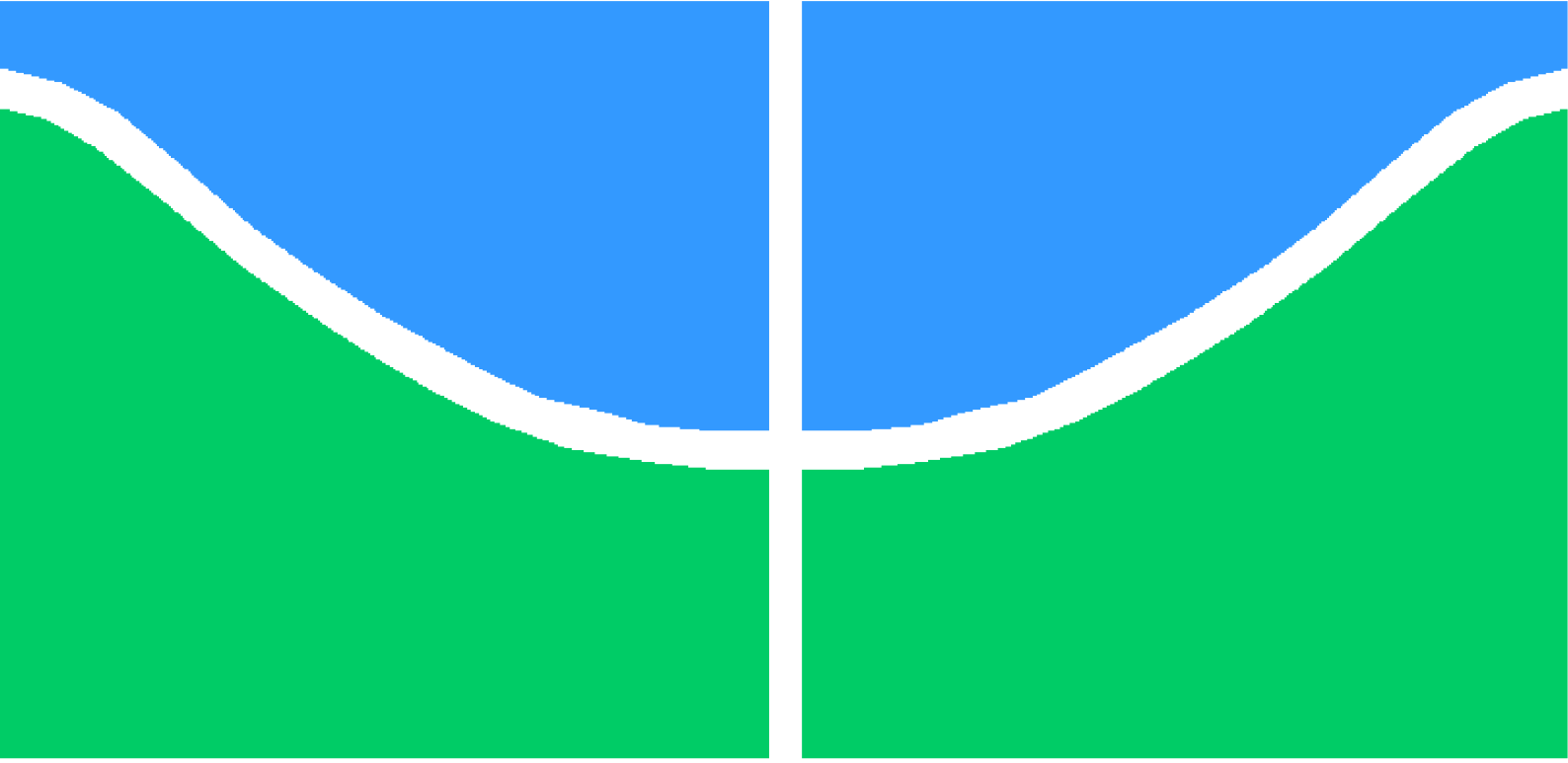


UM MÉTODO DE DEFESA CONTRA ATAQUES DoS BASEADO EM ASSINATURAS DIGITAIS

MARCONE PEREIRA DE ALMEIDA

TESE DE DOUTORADO EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

UM MÉTODO DE DEFESA CONTRA ATAQUES DoS BASEADO EM ASSINATURAS DIGITAIS

MARCONE PEREIRA DE ALMEIDA

ORIENTADOR: RAFAEL TIMÓTEO DE SOUSA JÚNIOR

TESE DE DOUTORADO EM ENGENHARIA ELÉTRICA

PUBLICAÇÃO:PPGENE.TD - Nº 133/2018
BRASÍLIA, DF, 02 DE OUTUBRO DE 2018

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

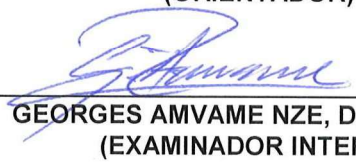
UM MÉTODO DE DEFESA CONTRA ATAQUES DOS BASEADO EM ASSINATURAS DIGITAIS

MARCONE PEREIRA DE ALMEIDA

TESE DE DOUTORADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR.

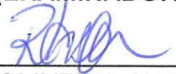
APROVADA POR:

RAFAEL TIMÓTEO DE SOUSA JÚNIOR, Dr., ENE/UNB
(ORIENTADOR)



GEORGES AMVAME NZE, Dr., ENE/UNB
(EXAMINADOR INTERNO)

FÁBIO BORGES DE OLIVEIRA, Dr., LNCC
(EXAMINADOR EXTERNO)



ROBSON DE OLIVEIRA ALBUQUERQUE, Dr., GSI/PR
(EXAMINADOR EXTERNO)

Brasília, 02 de OUTUBRO de 2018.

FICHA CATALOGRÁFICA

ALMEIDA, MARCONE PERERA DE
Um Método de Defesa Contra Ataques DoS Baseado em Assinaturas Digitais [Distrito Federal] 2018.
xxvii, 111p., 210 x 297 mm (ENE/FT/UnB, Doutor, Tese de Doutorado – Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica
1.Ataques de negação de serviço 2.Autenticação de pacotes
3.Identificação de tráfego 4.Assinaturas digitais
5.Criptografia 6.Segurança demonstrável
I. ENE/FT/UnB II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

ALMEIDA, M. P. (2018). Um Método de Defesa Contra Ataques DoS Baseado em Assinaturas Digitais. Tese de Doutorado em Engenharia Elétrica, Publicação PPGENE.TD-133/2018, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 111p.

CESSÃO DE DIREITOS

AUTOR: Marcone Pereira de Almeida.

TÍTULO: Um Método de Defesa Contra Ataques DoS Baseado em Assinaturas Digitais.

GRAU: Doutor ANO: 2018

Marcone Pereira de Almeida
SQN 314, BLOCO i, Apto 608.
70.767-090 Brasília – DF – Brasil.

Dedico a minha pequena Liz por ter me tirado e me dado tanto tempo necessário.

Agradecimentos

Acima de tudo, agradeço a Deus.

Especialmente, agradeço ao meu orientador Prof. Rafael Timóteo por sempre ter encontrado tempo entre os seus tantos compromissos, pela confiança, por também ter me desafiado a melhores resultados e a me ter dado ideias.

Agradeço também aos professores que compuseram a banca avaliadora no exame de qualificação, ao Prof. Georges, à Prof.ª Maristela e ao Prof. Robson, por terem proporcionado a melhoria deste trabalho.

Não posso também deixar de agradecer a todos que, ainda que involuntariamente, contribuiram para este sucesso.

"Vi descer do céu um anjo que trazia na mão
a chave do abismo e uma grande corrente."
(Bíblia Sagrada, Apocalipse 20, 1)

Resumo

Uma nova técnica de identificação de uma fonte de transmissão em uma rede baseada em interconexão de pacotes é apresentada neste trabalho. O modelo apresentado proporciona garantias de que uma fonte de tráfego é legítima, no sentido a permitir que o destino possa se certificar de que o fluxo na rede não tem a declaração de origem forjada. Este tipo de mecanismo tem aplicação fundamental na construção de esquemas de defesa e detecção contra ataques de negação de serviço *DoS* (*Denial of Serviçe*). O esquema proposto é baseado em assinaturas digitais designáveis, e argumenta-se que a proposta é sem precedentes na maneira que aquelas ferramentas criptográficas são aplicadas para construção de uma defesa contra ataques *DoS*. O esquema proposto não se baseia em um ferramental criptográfico de alta complexidade computacional, e se mostra apropriado para implementação de defesas de uma ampla gama de dispositivos, inclusive os de pequena capacidade. Uma análise da proposição é apresentada com uma argumentação da segurança do esquema, que é confrontada com conhecidas estratégias de ataque. Ainda mais, uma comparação da proposta às existentes na literatura é também defendida, destacando-se vantagens e desvantagem em relação ao que se encontra no atual estado da arte.

Palavras-chaves: Ataques de negação de serviço; autenticação de pacotes; identificação de tráfego; assinaturas digitais; criptografia; segurança demonstrável.

Abstract

This work presents a novel technique for source identification of a packet stream in a network, which intends to give guarantees that a specific network flow really comes from a claimed origin. This mechanism can be an essential tool for addressing Denial of Service (DoS) attacks. Based on designated verifier signature schemes, our proposal is an appropriate and unprecedented solution applying digital signatures for DoS prevention. Our scheme does not rely on an expensive public-key infrastructure and makes use of lightweight cryptography machinery that is suitable in the context of the small devices. We analyze our proposed scheme as a defense measure considering known DoS attacks and we discuss its resilience face to eventual adversaries. Furthermore, we compare our solution to already existent strategies, highlighting its advantages and drawbacks.

Key-words: Denial of Service (DoS) attacks; packet level authentication; traffic identification; digital signatures; criptography; provable security.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACK	Mensagem componente do protocolo TCP;
AS	(*Autonomous system*) Sistema autônomo;
Auth	Autenticado;
Bot	Código malicioso concebido para atender comandos remotos;
Botnet	Rede de computadore infectada e controlada por *bots*;
$D(\cdot)$	Algoritmos de deciframento;
DNS	(*Domain name system*) Sistema de domínio de nomes;
DoS	(*Denial of Service*) Ataque de negação de serviços;
DDoS	(*Distributed Denial of Service*) Versão distribuída dos ataques DoS;
DVS	(*Designated Verifyer Signature*) Assinaturas digitais designáveis;
$E(\cdot)$	Algoritmo de enciframento;
FIN	Mensagem componente do protocolo TCP;
HTTP	(*Hypertext Transfer Protocol*) Protocolo de transferência de hipertexto;
ICMP	(*Internet Control Message Protocol*) Protocolo de mensagens de controle de Internet;
IDS	(*Intrusion Detection System*) Sistema de detecção de intrusão;
IoT	(*Internet of Things*) Internet das coisas;
IP	(*Internet protocol*) Protocolo internet;
IPSec	(*IP Security Protocol*) Versão segura do protocolo IP;
IRC	(*Internet Relay Protocol*) Protocolo de aplicação de troca de mensagens de bate papo;
Hash	Função criptográfica de resumo de verificação de mensagens;
LDAP	(*Lightweight Directory Access Protocol*) Protocolo de acesso a serviço de diretórios;
ML	(*Machine Learning*) Aprendizado de máquina;

$\mathbb{N}$	Conjunto dos números naturais;
NA	Não autenticado;
NP	Classe de problemas cujas soluções conhecidas executam em tempo polinomial não determinístico;
NTP	(*Network Time Protocol*) Protocolo de gerenciamento de serviço de tempo;
OSI	(*Open System Interconnection*) Modelo de referência da ISO;
P	Classe de problemas que têm solução com um algoritmo determinístico em tempo polinomial;
PK	(*Public Key*) Chave pública;
PoS	(*Proof of Space*) Conceito de *prova de espaço de armazenamento*;
PoW	(*Proof of Work*) Conceito de *prova de trabalho*;
Pr	Função densidade probabilidade;
RST	Mensagem componente do protocolo TCP;
SDVS	(*Strong Designated Verifyer Signature*) Assinaturas digitais fortemente designáveis;
SK	(*Secret Key*) Chave privada;
SMTP	(*Simple Mail Transfer Protocol*) Protocolo de serviço de e-mail;
SYN	Mensagem componente do protocolo TCP;
SYN-ACK	Mensagem componente do protocolo TCP;
SSL	(*Secure Sockets Layer*) Protocolo seguro da camada de transporte;
SSDP	(*Simple Service Discovery Protocol*) Protocolo de descoberta de serviços;
TCP	(*Transmission control protocol*) Protocolo da camada de transporte orientado a conexão;
TTL	(*Time to Live*) Atributo de um datagrama no protocolo IP;
TLS	(*Transport Layer Security*) - Protocolo seguro da camada de transporte;
UDP	(*User Datagram Protocol*) Protocolo da camada de transporte não orientado a conexão;

WSN (*Wireless Sensor Networks*) Redes sem fio de sensores.

Lista de símbolos

ε	Palavra vazia;
$\mathcal{O}$	Oráculo aleatório;
$\lvert \cdot \rvert$	Valor absoluto;
$\Sigma_i a_i$	Somatório dos valores a indexados por i;
$\approx$	Valor aproximado;
$\neg$	Negação;
π_i	Valor de π indexado por i;
$\binom{a}{b}$	Combinação de a elementos tomados b a b;
If a Then b	Cláusula condicional significando: se a é verdade então b acontece;
$(p\|q)$	O evento p condicionado a ocorrência do evento q;
$< m, n >$	Variável composta formada pela relação entre os valores m e n;
$P(q)$	Predicado q;
[S]	Estado S;
[R,S]	Estado composto pelos estados R e S;
$\{a_i\}$	Conjunto dos elementos a indexados por i;
$P \to Q$	Transição do estado P para o Q;
$P \times \Sigma \to Q$	Função de transição entre os estados P e Q;
$a \leftarrow b$	Atribuição de valor: a recebe o valor de b;
$v \xleftarrow{\mathrm{r}} \{u\}$	Atribuição aleatória: v recebe um valor aleatorio em $\{u\}$;
$\equiv$	Indistinguibilidade incondicional;
$\overset{\mathrm{c}}{\equiv}$	Indistinguibilidade computacional;
$\overset{def}{=}$	Igual por definição;
$\overset{?}{=}$	Verificação de veracidade;

$\{X_n\}_{x\in\mathbb{N}}$	*Ensembles* indexados por $x \in \mathbb{N}$;
$\overset{P}{\rightleftharpoons}$	Transcrição da execução do protocolo P;
View^P	Visão da transcrição do protocolo P.

Sumário

1 Introdução

A disseminação de tecnologias baseadas na Internet como as em computação ponto a ponto — *P2P* (*Peer-to-Peer*), as em computação em nuvem e as de internet das coisas — *IoT* (*Internet-of-Things*) proporcionaram cenários em que uma quantidade até então não observada de dispositivos conectados que participam de múltiplos fluxos de informação suportados pelo protocolo Internet *IP* e sua estrutura de roteamento. Estes novos cenários têm apresentado desafios de segurança que vêm se mostrando distintos principalmente porque estabelecem condições para ataques em dimensões difíceis de se gerenciar, especialmente no caso de ataques de negação de serviço — *DoS* (*Denial-of-Service*), a qual se utilizou milhões de dispositivos *IoT*

Recentemente, ataques de negação de serviço têm sido desencadeados por cooptar um grande número de dispositivos *IoT*. Estas ameaças foram uma das principais causas de ataques lançados pelo vírus *Mirai* [1], o qual lançou recentemente um massivo ataque de negação de serviço tendo como alvo importantes provedores de serviço na Internet, perpetrando uma dimensão de tráfego até então não testemunhada. De fato, os dispositivos no ecossistema *IoT* sofrem pela falta de recursos de segurança, o que é explorado por atacantes para que os usem como um meio a partir do qual grandes ataques sejam desencadeados [2]. Por se constatar um crescimento contínuo de uma grande variedade de dispositivos *IoT*, no que diz respeito a energia, memória, computação e capacidade de comunicação, o desafio é o de se estabelecer medidas que combatam as ameaças originadas destes dispositivos [3].

A proposição desta tese faz aplicação de esquemas de assinatura digitial *DSS* (*Digital Signature Schemes*). Nestes últimos anos, tem-se constatado na literatura muitas aplicações baseadas em assinaturas digitais, os cenários de aplicação inclui problemas como os de roteamento seguro, gerenciamento de dados com privacidade, publicação oficial de documentação, entre outros [4, 5, 6]. O interesse pela aplicação de assinaturas digitais surge também com a crescente busca por tecnologias que proporcionam uma opção à centralização, ou seja, por tecnologias que fogem da dependência de uma terceira parte confiável [7, 8, 9].

Portanto, a proposição apresentada nesta tese é de um mecanismo de defesa contra ataques de negação de serviço baseado na aplicação de assinaturas digitais. O mecanismo proporciona que um receptor de tráfego possa saber distinguir um receptor por uma identificação baseada naquilo que ele transmite, criando um credenciamento em que o receptor o identifique, mas sem a necessidade de se estabelecer uma autenticação entre eles. O argumento nesta tese é que o mecanismo proposto se mostra resistente a adversários

que lançam das estratégias sejam elas de tentar copiar as credenciais de um transmissor legítimo e se passar por ele, seja de repetir o tráfego, ou até mesmo de tentar frustrar o esquema descartando ou modificando tráfego.

No texto, é adotada a expressão *prova* com diversos significados. Procurou-se ter o devido cuidado para que não fosse aplicada no sentido que não correspondesse a sua definição tradicional da matemática. Contudo, fazendo-se referência a alguns conceitos na literatura, houve também a devida cautela de não querer ferir a ideia de *prova* trazida nas suas definições originais. Deste modo, procurou-se preservar a expressão *prova* nos conceitos como *prova de trabalho* (*proof-of-work*), *provas de conhecimento zero* (*zero-knowledge proofs*), prova interativa, por exemplo. Uma discussão sobre referida conotação da palavra pode ser encontrada no Capítulo 4 de [10].

O texto está dividido em seis capítulos. No primeiro, é descrito o problema atacado por esta tese, apresentando-o na sua relevância como um objeto de pesquisa em aberto, mostrando sua importância para a contribuição do resultado alcançado.

No capítulo dois, são descritos o paradigma de segurança de informação e o modelo de adversário a ser adotado, os quais serão considerados na argumentação da validade do resultado apresentado.

O capítulo três disserta sobre as ferramentas de criptografia a serem adotadas, demonstrando-se em qual contexto e sob quais configurações a combinação delas se mantém segura e proporcionará a construção do modelo de defesa proposto.

No capítulo quarto, é descrito o resultado desta tese. Lá também é apresentado como a solução combina os elementos dos capítulos anteriores para se construir a proposta, reunindo as ferramentas criptográficas introduzidas e considerando o modelo de adversário descrito. Também naquele capítulo, é apresentado como a solução pode ser aplicada a um cenário prático.

O capítulo cinco apresenta os argumentos de que a solução proposta alcança a segurança pretendida por confrontá-la com o modelo de adversário adotado. Uma discussão e conclusão fecham o texto no capítulo seis.

2 Especificação do problema

2.1 Negação de serviço e a questão de identificação de tráfego

Métodos de análise de tráfego em redes de computadores têm se apresentado como um dos grandes desafios [11, 12], seja devido à velocidade com que novas tecnologias aparecem, mas também pela contínua mudança das necessidades impostas pela indústria com o consequente surgimento de novos contextos como o de computação em nuvem e a popularização de tecnologias associadas à Internet das Coisas — *IoT*.

Análise de tráfego é um conceito amplo, engloba as metodologias que têm como objetivos, entre outros: identificação de tráfego [13], sua classificação [14], detecção [12] ou rastreamento [11]. Entre esses tais objetivos, a identificação de tráfego é central, pois está sempre relacionada a algum aspecto da segurança de dados. Por exemplo, a questão da mensuração no estabelecimento de políticas de qualidade e garantia de serviço, e também na engenharia de tráfego e na contabilidade do uso de recursos de rede. Ainda mais, os métodos de classificação ou identificação de tráfego são comumente aplicados ao problema de negação de serviço — *DoS* (*Denial of Service*).

A Figura 1 ilustra um cenário no qual o problema da identificação de tráfego é considerado. O ambiente é de uma rede baseada no protocolo *IP* com um servidor que tem serviços de rede que podem ser acessados através da Internet. São destacadas duas redes de clientes destes serviços, redes A e B, os quais geram tráfego endereçado a este servidor. A identificação de tráfego se apresenta como um desafio em um cenário como este, considerando que há um dispositivo comprometido por um adversário malicioso (1) na rede B e que pode injetar tráfego de modo a parecer que a origem deste fluxo de rede seja a rede A.

O adversário que comprometeu o dispositivo em (1) pode ordenar que ele forje tráfego malicioso de modo que ele percorra e chegue em (2) e em (3) seja encaminhado como se daquele outro segmento de Internet tivesse saído, até chegar ao servidor em (4). Este mesmo dispositivo controlado pelo adversário pode ordenar que outro dispositivo realmente conectado à rede A (também comprometido) transmita tráfego de ataque DoS. Neste caso, o tráfego foi de fato originado na rede A.

Em (4) o servidor, destinatário de todo o tráfego de rede, poderá reagir. Porém, a informação que tem a respeito de quem é originador do fluxo malicioso é equivocada, e consequentemente ele erroneamente bloqueia a rede A em (5).

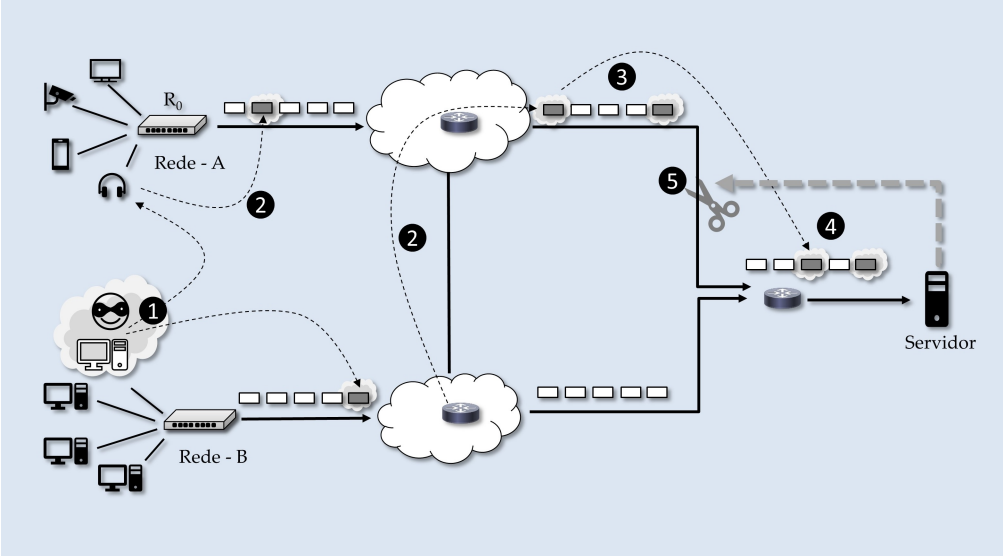


Figura 1 – Dinâmica de inserção de pacotes e a dificuldade de identificação.

Identificação de tráfego e sua relevância

O problema identificação de tráfego, e em especial na Internet, é central para que se estabeleça a solução de diversos outros problemas de segurança da informação relacionados a redes de computadores. A seguir, serão pontuadas as áreas em que identificação na rede exerce importante função ou com a qual mantém uma relação de dependência.

Privacidade do usuário — Análise de tráfego é um desafio quando a privacidade do usuário tem que ser preservada [14, 15]. A questão da anonimidade na rede e da neutralidade vêm de encontro aos objetivos de identificação e classificação, por um lado, e, por sua vez, sistemas baseados em informação sobre o usuário são necessários e vêm se popularizando: aplicações baseadas em posicionamento [16] ou baseadas em informação da rede pública de telefonia, por exemplo, estão se tornando cada vez mais comuns.

Engenharia de tráfego — Em redes, a análise de tráfego exerce um papel imprescindível proporcionando a aplicação de políticas estabelecidas por diferenciação de serviço e qualidade de tráfego. Aqui também a tarefa de identificação do tráfego é crítica, pois as soluções de engenharia de tráfego frequentemente prescindem de mecanismos eficientes de classificação e identificação de fluxo na rede.

Controle de acesso — Com a atual heterogeneidade de dispositivos sob uma grande diversidade de protocolos de dados e de comunicação, um dos grandes desafios é a questão do controle de acesso e autorização [17], o que geralmente implica em identificação do transmissor.

E-mail spam — O problema de e-mail não solicitado ainda permanece crítico, e é um desafio que cresce anualmente [18]. Nos últimos anos, tem-se observado o aumento tanto de *malwares* quanto de *bots* associados à disseminação de *spam*. Em 2016 chegou-

se ao patamar de 400 bilhões de mensagens não solicitadas por dia, o que representava cerca de 86% dos e-mails transmitidos [19]. A indústria vem tentando estabelecer um padrão de segurança para o serviço de e-mail na Internet, pelo o envio de mensagens com autenticação de servidores com aplicação de protocolos seguros, como o *TLS*, ou ainda com uso de assinaturas digitais.

2.2 Ataques de negação de serviço — DoS

Ataques de negação de serviço — *DoS* (*Denial of Service*) ocorrem quando um adversário (ou um conluio formado pelas partes controladas por ele) direciona maliciosamente tráfego de modo a inviabilizar participantes legítimos de utilizarem serviços ou recursos providos pelo alvo do ataque. Muito embora possa ser um problema já do conhecimento desde a década de 80 [20], tornou-se uma questão de segurança da informação e de redes de computadores nas duas últimas décadas.

Uma das principais fontes de risco associados a ataques *DoS* diz respeito à dinâmica de roteamento da Internet, que tem como objetivo primordial a garantia de entrega do tráfego sendo encaminhado. Deste modo, isso que é uma característica fundamental no *IP* é explorado por adversários maliciosos que podem sintetizar um grande volume de tráfego e direcioná-lo a fim de exaurir os recursos de um provedor de serviço na rede. Com efeito, uma das maiores barreiras à consolidação de um mecanismo de defesa contra esses ataques é a dificuldade de identificação da origem do tráfego, o que é uma informação imprescindível para que o destinatário possa efetivamente filtrá-lo [21]. Entretanto, a inserção de mecanismos de filtragem de tráfego vai de encontro à garantia de entrega, e de encontro à tão defendida liberdade na rede.

O trabalho em [22] discute o local adequado, ou seja, em que camada no modelo de referência mecanismos de defesa contra *DoS* poderiam ser melhor implantados para que funcionassem mais eficazmente. A questão é que ataques *DoS* ocorrem com dinâmicas diversas, e, não obstante a definição fazer referência explícita a *serviços* (camada de aplicação), ataques ocorrem atualmente principalmente pela inundação de tráfego na camada de transporte. O relatório técnico *Verisign Distributed Denial of Services Trends Report* para o quarto trimestre de 2017 [23] aponta que 42% dos ataques *DoS* naquele período foram por inundação de tráfego *UDP* e 30% *TCP*. Com efeito, os recentes ataques têm utilizado muito mais tráfego das camadas de transporte e da camada de rede, em contraposição à exploração de comunicação no nível de aplicação representando apenas 2% dos ataques. Portanto, os efetivos mecanismos contra ataques *DoS* devem funcionar na defesa das camadas de rede e de transporte, impondo um grande desafio de se encontrar uma solução que não impacte a infraestrutura da rede, seja por evitar mudar protocolos consolidados, seja por não atribuir o ônus de funcionalidades aos agentes nesta

infraestrutura.

Ainda levando em consideração a dinâmica em que os ataques *DoS* ocorrem, tem-se observado que o volume de tráfego transmitido em ataques *DoS* tem crescido [24, 25]. O já citado relatório técnico da *Verisign* [23] também aponta que, ao mesmo tempo em que é possível se testemunhar o crescimento da frequência desses ataques, o volume de tráfego despejado em um único ataque tem também aumentado, com os maiores ataques alcançando 20 Gbps, a exemplo dos ocorridos no primeiro trimestre de 2016.

Nos últimos anos, tem-se também testemunhado o aparecimento de novas formas de executar ataques dessa natureza, principalmente os desencadeados a partir de grandes redes de computadores infectados e que frequentemente cooptam dispositivos dispersos em todo o mundo [24]. É por isso que a forma mais nociva desses ataques é denominada *DDoS* (*Distributed Denial of Service*), devido a sua natureza de grande dispersão geográfica. A despeito do esforço despendido na pesquisa e no aprimoramento de mecanismos de defesa contra tais ataques, eles ainda vêm crescendo continuamente, ocorrendo com um aumento tanto em número de participantes quanto em volume de tráfego transmitido [23].

Atualmente, grandes ataques desencadeados pela infecção por softwares maliciosos capazes de cooptar dispositivos e fazê-los responder remotamente a comandos têm sido cada vez mais frequentes. Entre os adversários que podem desencadear esse tipo ataque estão os chamados *bots*, os quais tomam controle de uma ampla e dispersa rede de dispositivos, para compor um conluio de ataque assim denominado *botnet* [18]. Usando tais redes, um adversário em qualquer lugar na Internet pode enviar comandos aos dispositivos infectados que os executarão, podendo assim desencadear um maciço ataque contra alvos [26]. É usando essa arquitetura que recentemente *botnets* têm interrompido servidores importantes na infraestrutura de *DNS*, servidores *WEB* e serviços de diretório *LDAP*, agindo por inundar estes servidores com um pesado tráfego.

A disseminação de tecnologias relacionadas à Internet das Coisas — *IoT* [27], por ter ampliado sem precedentes o universo de dispositivos que podem estar conectados, permitiu o aparecimento de um novo fenômeno no sentido a criar possibilidades de ataques *DoS* em dimensões até então não vistas. A exemplo do ataque à rede da *Dyn.com* em outubro de 2016, que afetou o funcionamento de importantes serviços privados, tais como *Twitter*, *Amazon.com*, *Spotify*, *Comcast*, *Fox News* e *PayPal*, entre outros, o potencial de ataque destas redes é uma demonstração do quanto tráfego os recentes ataques *DoS* podem despejar num alvo por explorar a vulnerabilidade daqueles pequenos dispositivos.

2.3 Uma classificação das estratégias de ataque DoS

Trabalhos de pesquisa têm sugerido classificação dos modelos de ataque [28, 29, 30, 25]. Os aspectos considerados na construção de uma taxonomia dos ataques são, entre

outros, a estratégia, o montante de fluxo transmitido e o protocolo de rede explorado. As seções a seguir descreverão esses aspectos do modelo de adversário que será considerado neste trabalho.

2.3.1 Ataques por inundação

Ataques por inundação de tráfego são o que se denomina o padrão *de facto* dos ataques de negação de serviços, vez que os primeiros ataques *DoS* registrados exploravam basicamente a capacidade de coordenar um grande volume de tráfego endereçado a um alvo. Como já foi colocado, ataques de inundação têm sido os mais testemunhados nesses últimos anos, o que recentemente foi agravado pelo aparecimento de *botnes* e ainda mais recentemente pelo advento de *IoT* [18, 23].

Um dos principais meios de ataque *DoS* explora por inundação de tráfego um mecanismo fundamental da camada de transporte, necessário ao estabelecimento de conexões *TCP*. Por explorar esse mecanismo, um adversário pode exaurir os recursos necessários ao estabelecimento de conexão no alvo atacado, ainda que o equipamento no destino possa gerenciar um número elevado de conexões.

O protocolo *TCP* é orientado à conexão, que é estabelecida por uma troca de mensagens entre cliente e servidor numa rede, e é um protocolo muito presente no funcionamento da Internet, justamente porque é através dele que uma grande parte dos serviços na Internet estabelece comunicação. Como é o caso de importantes protocolos na rede: *HTTP(S)*, *DNS*, *SMTP*, entre outros. A manifestação de vontade de se estabelecer uma conexão acontece quando o cliente envia um segmento *TCP* com o bit *SYN* (*synchronize*) no campo de opções (um octeto no cabeçalho do segmento *TCP*) habilitado. A resposta a esta solicitação de conexão ocorre quando o servidor envia um segmento *TCP* com os bits *SYN* e *ACK* habilitados (mas poderia rejeitar a solicitação ou ainda encerrar a conexão habilitando na resposta os bits *RST* e *FIN* respectivamente). A conexão só é efetivamente estabelecida depois de uma confirmação do solicitante com um segmento em que o bit *ACK* é habilitado.

Porém, se esta confirmação nunca for enviada, o destino pode permanecer reservando recursos de rede, o que pode ser explorado por um adversário, o qual poderá então maliciosamente criar uma inundação de solicitações de conexão, exaurindo os recursos de rede do servidor destino e desse modo artificializando um contexto no qual existe um grande número de clientes que solicitam conexão. Um usuário malicioso pode ainda agravar esse contexto por combinar essa dinâmica com ataques *spoofing*, o que será definido à frente.

Ataques *DoS* por inundação *SYN* são um dos mais combatidos, porque nos últimos anos os tais têm aparecido como um dos mais frequentes entre os ataques de inundação

de tráfego, contudo, outros ataques por inundação também têm se destacado no recente cenário de segurança na Internet. Isso é o que se pode constatar do relatório técnico [23], o qual aponta que os ataques por inundação de tráfego que ocorreram no período considerado foram em sua maioria de tráfego *UDP*. Mas também é grande a ocorrência de ataques de inundação de requisições *DNS* e de tráfego requisições *NTP*, entre outros.

O cenário apresentado acima é suficiente para o convencimento de que o problema de ataque de negação por inundação de tráfego é uma preocupação atual, o qual vem sendo objeto de pesquisa desde quando foi diagnosticado como uma grande vulnerabilidade da rede [31, 32]. Ademais, a cena atual é de um agravamento, haja vista o aparecimento de redes *IoT*, cujas vulnerabilidades permitem que sejam utilizadas como uma ferramenta de ataque de inundação: a exemplo do ataque *DDoS* pelo *malware Mirai*, o qual alcançou patamares de tráfego de 1,5 Gpbs em meados de 2016, e usava como base dispositivos *IoT* [2].

2.3.2 Ataques de manipulação de conteúdo

Outra das frequentes estratégias de ataque *DoS* acontece pela manipulação de conteúdo, principalmente quando se objetiva explorar má implementação de protocolos (programação deficiente da pilha de protocolos *TCP/IP*), falhas em sistemas operacionais ou em *hardware* de equipamentos (projeto defeituoso de *firmware*) e sistemas com falhas de programação e *bugs*. Geralmente, esse tipo de ataque é desencadeado por transmissores maliciosos que sintetizam tráfego intencionalmente malformado, causando a paralisação do sistema ou dispositivo atingido. Manipulação de tráfego é também um mecanismo comum de se atacar por se demandar recursos de rede de um servidor, simplesmente endereçando-o um pesado número de requisições malformadas. O relatório [23] aponta que 12% dos ataques *DoS* no período reportado ocorreram pela transmissão de fragmentos de pacote *IP*, o que demanda tempo de processamento dos receptores na tentativa de reconstrução da integridade do pacote.

Entre os ataques de manipulação, um que se destaca entre os meios de se perpetrar negação de serviço é o denominado ataque *spoofing* [33, 34]. A ameaça consiste na ação de um adversário que manipula tráfego para que pareça ter sido originado em um endereço de uma vítima que tem portanto sua identidade usurpada. O ataque acontece porque a infraestrutura baseada em interconexão de pacotes, como a de redes *IP*, permite que um originador de tráfego possa livremente atribuir o endereço de origem nos campos de identificação do remetente. Este tipo específico de ataque é difícil de ser identificado pelo destinatário do tráfego, o qual o recebe incapaz de perceber a falsificação. Muito menos aquele que teve seu endereço explorado ficará ciente de que sua identidade foi usurpada por um adversário na rede. Ainda na classe de ataques de manipulação de tráfego, um ataque também muito frequente no cenário de segurança de redes é o denominado *ata-*

que por repetição (*replay attacks*). Consiste na estratégia do adversário de primeiramente coletar tráfego capturado, depois maliciosamente tentar reenviá-lo ao destinatário, para isso realizando a manipulação necessária. Este tipo de estratégia pode ser lançado tanto para provocar congestionamento, quanto para se burlar segurança de sistemas, como mecanismos de autenticação.

2.3.3 Ataques por amplificação

Como já discutido, ataques por *spoofing* têm sido um meio frequente de se realizar *DoS*. De fato, ele se inclui numa classe mais abrangente de estratégias, que engloba aquelas que se utilizam de meios de esconder a real fonte do ataque. Usando esta estratégia, usuários maliciosos podem criar uma grande quantidade de solicitações em nome de alguém, uma parte que tem seu endereço usurpado, de tal modo que solicitações são direcionadas a um provedor de serviço na rede, mas com a particularidade de que este provedor, não capaz de distinguir a personificação, envia a resposta ao endereço forjado. Este tipo de ataque é conhecido por *ataques por reflexão* (*reflector attacks*) [35, 34, 28].

Ataques por reflexão são particulares porque diferem dos convencionais ataques *DoS* no sentido a criar um fluxo de rede ainda mais disperso, pois o adversário não precisa tomar controle de uma máquina que faz a reflexão, o refletor (*reflector*): se o adversário endereça tráfego contendo requisições no nome de um alvo a uma grande quantidade de servidores *DNS*, as correspondentes respostas *DNS* serão endereçadas à vítima de maneira natural, ou seja, os servidores *DNS* não precisaram ter sido comprometidos, e a vítima que teve o endereço forjado terá que gerenciar o pesado tráfego que receberá.

Ataques por reflexão também estão entre os mais frequentes meios de se sintetizar inundação de tráfego atualmente. Recentemente, um dos serviços mais alvejados por essa estratégia é o de descoberta de serviços, principalmente quando o protocolo explorado é o *SSDP* [23], embora serviços como o de *LDAP* e o de *DNS* estejam também entre os alvos mais frequentes.

2.3.4 Ataques remotos coordenados

Ataque coordenado [36] é uma designação para toda ofensiva na qual o adversário controla remotamente um grande número de computadores na rede, os quais colaboram para que ele alcance seu objetivo. O adversário pode exercer seu controle através de uma diversidade de protocolos: *TCP/IP*, *HTTP*, *IRC*, entre outros. Definem uma classe dos ataques não perpetrados diretamente pelo adversário, pois a estratégia do adversário reside em se ocultar. Um típico exemplo destes ataques são os chamados *escaneamentos* (de porta de serviços, de vulnerabilidades, de intervalo de endereços *IP*, etc.). Geralmente o adversário quer coletar informação de uma rede a ser atacada escaneando-a, mas é interessante para ele não ser percebido.

Não sem propósitos, quis-se aqui inverter a posição de uma definição mais geral como uma subseção de um conceito mais específico: de acordo com a definição acima, pode-se afirmar que alguns ataques *DoS* se incluem na classe de ataque coordenados. Recentemente, e principalmente nestes últimos anos, grandes ataques foram perpetrados com uso de um novo tipo de ataque coordenado: *crypto ransomware* [37]. Os mais notórios casos, *WannaCry* e *Petya/NotPetya* [18], tiveram como alvo principal versões mais antigas de sistemas operacionais baseados na plataforma *Microsoft Windows*, e consistiam basicamente em um código malicioso que realizava enciframento da totalidade ou quase totalidade dos dados na máquina, espécie de sequestro digital de dados em que o adversário exigia pagamento em moeda digital.

Quando um adversário age fazendo uso de *ransomware* ele pode de uma maneira coordenada dar ordens para que redes inteiras tenham seus serviços negados, paralisados. Como as ameaças desencadeadas de redes *botnets*, a negação de serviço não se dá por uma inundação de tráfego advinda diretamente de quem controla o ataque, mas sim de uma maneira remota, indireta, dispersa geograficamente, o que se mostra como um grande desafio. Por outro lado, ataques coordenados originados em redes *botnets* servem principalmente à negação de serviço por inundação de tráfego.

2.4 Uma classificação dos mecanismos de defesa contra DoS

Nesta seção, será apresentado o estado da arte das estratégias de defesa existentes contra ataques *DoS*. Aqui, também serão discutidas as vantagens e desvantagens de tais esquemas, por se apontar seus pontos fracos e fortes.

Adotar-se-á uma separação desses esquemas baseada na classificação encontrada na literatura [28, 34, 36], a qual guia a definição das subseções a seguir.

2.4.1 Mecanismos de defesa baseados em filtros

Um mecanismo de defesa contra ataques *DoS* consiste primordialmente na interpretação do tráfego e por conseguinte na adoção de alguma ação: seja ela a de alertar sobre uma atividade suspeita, de fazer registro em *log* de sistema, ou até mesmo a de bloquear o tráfego indesejado por se finalizar a conexão [38, 39]. Heurísticas baseadas em filtragem de tráfego são comumente implementadas em muitos equipamentos de segurança de redes, tais como *firewalls* e sistemas de detecção de intrusão *IDS* (*Intrusion Detection System*), e se mostram como o mecanismo mais comum de se fazer a interpretação de tráfego a fim de se construir defesas contra ataques *DoS*.

Convencionalmente, filtragem de pacotes é conduzida por uma de duas abordagens [39]: (a) experiência do que é um *mau uso*, em que a *assinatura* de um ataque é aprendida do que já se presenciou com incidentes reportados, *bugs*, falhas, vulnerabilidades

documentadas. Portanto, o conhecimento *a priori* do que venha a ser considerado tráfego malicioso é usado para determinar regras de filtragem; (b) experiência do que seja um *tráfego anômalo*, ou seja, o conhecimento adquirido ao longo de um período para que se determine o que seja um tráfego não convencional porém não malicioso. Contudo, como já afirmado, a exploração por ataques na rede possui uma grande dinâmica, com surgimento constante de novas formas de fazê-la, frente a que os modelos baseados em filtragem se colocam em desvantagem, seja quando adotam o paradigma do mau uso ou da anomalia [34].

Entre outras abordagens não incluídas nestes modelos básicos, a literatura aponta uma diversidade de outras estratégias de filtragem, algumas baseadas em aprendizado de máquina, outras baseadas na consideração atributos específicos do datagrama trafegado, outras baseadas na rota traçada pelo tráfego, entre outras abordagens que serão descritas a seguir.

Filtragem baseada em atributos do tráfego

Modelos que definem esquemas de filtragem baseada em atributos do tráfego transmitido têm sido propostos na literatura, tais como os que se utilizam de filtragem baseada no endereço *IP* da origem, ou na quantidade de tráfego já enviado por um certo remetente.

O trabalho em [40] define um esquema de filtragem baseada na distância do tráfego recebido, que é estimada levando em consideração atributos do datagrama como o *TTL*. A estimativa da distância é usada para inferir a veracidade da origem do fluxo de rede, numa tentativa de se detectar um tráfego forjado.

Por sua vez, em [41] é proposto um esquema de filtragem que leva em consideração o endereço de origem do tráfego, fazendo-se uma contabilidade dos endereços que têm enviado tráfego não anômalo. Esse esquema leva em consideração que o espectro de endereços de rede dos quais se recebe tráfego aumenta num ataque *DoS*.

Já o trabalho em [42] propõe se construir o conhecimento do que deve ser filtrado baseado no padrão de tráfego, a partir de estatística do volume de dados que está sendo enviado e recebido pela vítima.

Um mecanismo de filtragem baseado no roteamento é proposto em [43], em que os autores sugerem usar a informação de roteamento para determinar se um datagrama chegando a um roteador é válido com respeito à topologia conhecida (por exemplo, as informações constantes em roteadores de borda de sistemas autônomos) e às condições de acesso impostas pelas regras de roteamento.

Filtragem baseada em habilitação (*capabilities*)

Mecanismos de controle na rede baseados em habilitação (*capability*) são aqueles em que as partes possuem algo que possa ser usado como uma demonstração de que possuem autorização, são capazes ou habilitadas a fazer algo. Por exemplo, em uma rede IP pacotes de dados com uma autorização são tratados sob uma determinada política de preferência. Diversos esquemas de defesa em redes de computadores foram propostos com base na gerência de habilitações [44, 45, 46]

Um esquema denominado *StopIt* é proposto em [47], no qual se supõe que transmissores e receptores de tráfego se identificaram previamente para que então troquem informação de autenticação baseadas no endereço de rede do transmissor. O esquema se baseia na suposição de que ataques de *spoofing* não são realizados pelo adversário; o que é uma suposição que põe em dúvida a eficácia do esquema, pois, como já foi colocado anteriormente, *spoofing* é uma das principais ferramentas que proporcionam ataques *DoS*. A solução faz também uso de uma função *hash* com criptografia, ou seja, uma chave secreta é usada para se gerar e verificar o resultado do *hash* das mensagens, o que exige um canal seguro entre transmissor e receptor para que essa chave seja combinada.

Uma proposta de se usar autenticação embutida no tráfego transmitido com uso de assinaturas digitais e algoritmos de *hash* é apresentada em [48]. O esquema funciona por exigir que qualquer participante na rede que deseje enviar tráfego deva pedir autorização ao receptor (de fato ao roteador de borda do sistema autônomo a que este pertence). A autorização é enviada pelo receptor fazendo-se uso de uma assinatura digital. Como no esquema em *StopIt* [47], deve haver uma prévia identificação das partes comunicantes. Porém, de igual modo, o uso de algoritmos criptográficos de *hash* exige canais seguros entre as partes.

Outra solução de defesa baseada em habilitação é proposta em [45]. O esquema funciona por gerenciar solicitações de transmissão através de habilitações, inserindo-as nos pacotes. Essa é outra proposta que usa criptografia através de funções criptográficas de *hash*, as quais são aplicadas para assegurar a privacidade das habilitações e consequentemente evitar que adversários que as visualizem obtenham alguma vantagem. Assim como nas outras propostas, é demandado um canal seguro para a configuração prévia de chaves.

Também baseada em habilitação de tráfego, a proposta em [49] define um mecanismo de controle de prioridade de tráfego em que a filtragem considera o tráfego indesejado como de menor prioridade. O mecanismo funciona por permitir que uma fonte de tráfego possa interagir com o destino para inicialmente se identificar com o receptor, depois ele recebe a credencial necessária a se habilitar para transmissão de tráfego. A interação inicial é realizada por um esquema semelhante ao do *handshake* do protocolo *TCP*, ao final da qual o transmissor tem informação suficiente para marcar o trafego que

ele envia. Para isso, um algoritmo criptográfico de *hash* é utilizado. Portanto, este é outro exemplo em que um canal seguro é previamente demandado entre as partes.

Baseado em criptografia simétrica, o trabalho em [50] propõe marcar pacotes usando uma chave secreta previamente combinada; é proposto se usar o esquema de troca de chaves Diffie-Hellman [51], o que é outro exemplo em que um canal seguro é demandado entre as partes.

2.4.2 Mecanismos de defesa baseados em aprendizado de máquina

Em quase sua totalidade, as defesas que utilizam alguma solução de aprendizado de máquina *ML* (*Machine Learning*) seguem uma abordagem preventiva. Assim, as defesas baseadas em *ML* têm funcionado para compor mecanismos de detecção ou de prevenção de ataques *DoS* [28], mas não proporcionando uma ação de bloqueio para que eles sejam efetivamente cessados.

Agrupamento de dados é utilizado em alguns modelos propostos na literatura [52, 53, 54]. O trabalho em [52] propõe um modelo no qual um método de classificação observa características do tráfego de rede, algumas baseadas em estatística, outras baseadas na entropia[1] de aspectos do fluxo de rede: endereço *IP*, tipo ou tamanho do pacote, porta do serviço destinatário, entre outros aspectos. Já o trabalho em [53] descreve um método no qual uma classificação de tráfego conhecido é usada para treinar um mecanismo que decide sobre se uma transmissão contém anomalia, propondo criar um esquema capaz de reconhecer ataques, mesmo os que não são previamente conhecidos por seu padrão ou assinatura. Uma solução para as ameaças no contexto de redes de sensores sem fio, mas também fazendo uso de técnicas de agrupamento, é apresentada em [54].

Análise multivariada de dados é outra das ferramentas de aprendizado de máquina aplicada a métodos que propõem analisar o tráfego transmitido e detectar ataques conhecidos e desconhecidos, como o proposto nas técnicas em [56, 57].

Uma heurística que faz uso de árvore de decisão na análise da relação das quantidades de tráfego ingressando e deixando uma determinada rede é proposta em [42], a tal considera que há uma correlação entre a taxa de entrada e saída de dados numa situação normal. Sendo assim, quando a rede está sendo atacada, o padrão de tráfego pode ser comparado para se detectar um eventual ataque.

Soluções contra ataques *DoS* no contexto de computação em nuvem têm sido objeto de pesquisa recente [58, 59, 60]. Um modelo baseado em multi agentes é proposto em [59]. O trabalho desenvolve um mecanismo de detecção de ataques *DoS* e propõe coordenar múltiplos agentes para que colaborem para que um agente coordenador detecte

[1] A entropia pode ser pensada como uma descrição mínima da complexidade uma variável aleatória [55]. Aqui pode ser entendida como uma medida de quanto informação uma variável pode carregar.

ataques.

Por sua vez, o estudo em [60] apresenta uma comparação de diferentes métodos de aprendizado de máquina aplicados na defesa de ataques *DoS*, especificamente os que utilizam algoritmos de classificação.

2.4.3 Mecanismos de defesa contra ataques DoS em IoT

A literatura aponta diversas propostas de mecanismos de detecção de ataques *DoS* no ambiente de dispositivos *IoT*. Os trabalhos recentes em [61, 62, 63] apresentam o estado da arte elencando os mecanismos de defesa para *IoT* conhecidos na literatura atual. Até onde foi possível se verificar neste trabalho, na literatura recente poucos são os exemplos de mecanismos proativos contra ataques *DoS*, ou seja, que apresentam um meio de impedir ataques em redes *IoT* uma vez que o ataque foi detectado.

Em [64], os autores realizam uma análise dos ataques que podem ser desencadeados por se explorar os serviços de rede disponíveis em dispositivos *IoT*. Com base nesse conhecimento, eles propuseram um mecanismo de defesa que faz uso de *honeypots* e *sandbox*, que são usados para atrair adversários que maliciosamente acessam seus serviços para assim se criar uma estratégia de defesa. A abordagem adotada naquele trabalho é portanto proativa, mas não considera a rede *IoT* como uma fonte de ataque, ou seja, como o local onde as ameaças são originadas.

Um outro exemplo de uma abordagem proativa [65] faz aplicação de um *firewall*, usado para prover segurança entre dispositivos em um ambiente *IoT*. Na estratégia adotada pelos autores daquele artigo, regras de *firewall* são aplicadas para filtrar o acesso a serviços e recursos do ambiente e também para que seja filtrado o que os dispositivos podem transmitir para a Internet.

O artigo [66] propõe o uso de criptografia para prover uma comunicação segura entre dispositivos, em uma abordagem de proteção fim a fim. Os autores propõem um modelo de adversário, listando as ameaças que podem ser mitigadas pela solução proposta por eles. Outra solução de segurança que também foca na comunicação inter dispositivos é a apresentada em [67], a qual usa algoritmos de *hash* para prover autenticação mútua das partes. Os autores em [68] também propõem um esquema de autenticação mútua no contexto de redes de sensores sem fio. Entretanto, estas abordagens não foram planejadas para defesa contra ataques de negação, e nem poderiam ser por se basear em uma prévia autenticação entre as partes.

O paradigma colaborativo é seguido por algumas propostas no contexto *IoT*. Em [69, 70] são apresentados mecanismos colaborativos, nos quais as partes trocam informação para alimentar um esquema de detecção de intrusão em redes de sensores sem fio. As soluções nos citados trabalhos limitam-se a alarmar ameaças, não proporcionando uma

ação proativa diante de ataques.

2.5 Mecanismos de defesa baseados em criptografia

Criptografia é mais uma entre outras já citadas ferramentas que compõem soluções contra ataques *DoS*. Dentre as propostas que fazem aplicação de métodos criptográficos, a literatura aponta desde as que usam ciframento do fluxo de rede, fazendo uso de criptografia simétrica para privacidade dos dados, até as que fazem uso de infraestrutura de criptografia baseada em chave pública para autenticação das partes, com verificação por autoridades certificadoras. O uso de algoritmos de *hash*, seja na sua forma convencional ou nas versões criptográficas desses algoritmos já foi discutido nas seções anteriores na exemplificação de soluções que os aplicam.

Os primeiros métodos de defesa contra *DoS* baseados em criptografia já eram propostos por Karn e Simpson [71] em 1999. A solução apresentada por eles era a de que fosse usado *cookies* baseados em funções *hash* para se construir um mecanismo de chave de sessão baseado no *IPSec*, uma versão segura do protocolo Internet. A chave de sessão seria derivada de uma chave previamente combinada entre as partes. Na mesma época, em 1998, Harkins e Carrel [72] propõem um protocolo de troca de chaves também aplicado a versões seguras do protocolo *IP*. Porém, os tais modelos baseados em troca de chaves e autenticação já eram objeto de crítica [73], quando se verificou que o mesmo ferramental criptográfico poderia usado para se explorar ataques de negação, exatamente porque criptografia pode implicar em um aumento da complexidade computacional. Então, o criticismo questionava a viabilidade computacional daquelas soluções.

Meadows [74] foi quem primeiro analisou a aplicação de criptografia no sentido de avaliar o quanto uma autenticação forte permitiria um ataque de negação por demandar processamento. Ou seja, ela discutiu o quanto o maquinário necessário a se estabelecer uma autenticação forte poderia custar computacionalmente e funcionar mais como uma vulnerabilidade a ser explorada do que efetivamente proporcionar proteção. A autora aponta a necessidade de escalar os mecanismos de autenticação em termos de robustez, dosando-os ao longo da comunicação: criptografia de menor custo de processamento no início, menos resistente portanto; e ao longo da comunicação uma criptografia mais resistente, embora de maior custo computacional.

2.5.1 Defesa com uso de prova de trabalho

A ideia de *prova de trabalho — PoW* (*Proof-of-Work*) é devida a Dwork e Naor [75]. O conceito se baseia em uma prova interativa em que uma parte exige da outra um comprometimento de esforço (tempo de processamento) para que acessos aos recursos sejam conquistados. Por exemplo, um servidor provedor de serviço de e-mail demandaria,

quando estivesse sob uma ataque de negação de serviço, que aqueles que desejassem enviar mensagens deveriam resolver um problema computacional que exigisse tempo de processamento.

Com efeito, a ideia criou um paradigma em que uma parte pode desafiar uma outra a mostrar o comprometimento prévio de seus recursos para que este tenha acesso aos recursos oferecidos. Este paradigma vem sendo aplicado a diversos problemas de segurança da informação: além de no problema de e-mail não solicitado (*spam*), que foi o originalmente atacado no artigo [75], o paradigma também tem sido aplicado na defesa de ataques *DoS* [76, 77, 78, 79], em esquemas de cripto moedas [9] e na proteção de servidores que proveem serviços baseados que implementam segurança baseada nos protocolos *TLS/SSL* [80].

Quem primeiro aplicou o conceito de *PoW* na defesa de *DoS* foram Juels e Brainard [77]. Naquela proposta, os autores consideram o problema específico do ataque *DoS* por inundação de requisições de conexões *TCP*. Por sua vez, Back [78] estendeu o mecanismo para uma versão não interativa, apropriada a quando existir uma impossibilidade de se estabelecer uma negociação entre o cliente e o provedor do serviço. Mais tarde, uma versão na qual a emissão de *puzzles* podia ser delegada foi proposta [81], o que permitiu que servidores que demandassem provas de trabalho não necessariamente tivessem que gerenciar o mecanismo, proporcionando uma estrutura hierárquica do protocolo. Também mais tarde, foi proposta em [82] uma outra extensão ao mecanismo na qual o cliente pode dinamicamente ajustar a complexidade da prova a depender de se estar ou não num contexto de ataque.

O conceito de *PoW* tem sido revisto e estendido desde a sua concepção. O artigo em [83] discute o ataque desencadeado por um conluio que pode juntar um grande poder de processamento para resolver com eficiência *puzzles*, mostrando-se assim uma ameaça à solução quando aplicada a defesa de ataques coordenados. A questão posta por eles é de que o processamento a ser realizado é conhecido publicamente: algoritmos de *hash* podem ser previamente computados. Aqueles autores propõem portanto que haja uma dinâmica no processamento dos *puzzles* e sugerem substituir o desafio baseado em *hashes* pelo processamento de um algoritmo a ser dinamicamente escolhido.

Recentemente, os autores em [84] propuseram um conceito relacionado à PoW, mas, de todo modo, original em relação àquele. O novo conceito é denominado prova de espaço — *PoS* (*Proof-of-Space*). Com a mesma preocupação com adversários que podem juntar poder computacional a fim de trapacear nos esquemas de prova de trabalho, os autores propuseram uma prova baseada na capacidade de ocupar um considerável espaço de armazenamento, o que inviabiliza grandes ataques coordenados.

Já se passaram mais de duas décadas desde a concepção da ideia de prova de trabalho, contudo o paradigma não permitiu se construir um mecanismo prático na de-

fesa contra *DoS* [85]. A imposição de uma tarefa que tem que ser resolvida como uma demonstração de que o transmissor comprometeu tempo de processamento ou espaço de armazenamento não se mostra adequado, por exemplo, nos novos desafios criados por ataques desencadeados de redes *IoT*: que geralmente são desprovidos de capacidade seja de processamento ou de armazenamento. Ademais, não há garantia de que quem realmente resolveu o desafio é de fato quem está sendo credenciado.

2.5.2 Defesa com rotulação do tráfego

A literatura aponta o uso de algoritmos de *hash*, seja para marcar o tráfego para eventualmente filtrá-lo, seja para identificar a origem fazendo bloqueio de quem está causando problemas.

Este esquema está de fato incluído em uma abordagem mais genérica, na qual o fluxo de rede é processado para que alguma rotulação seja inserida, seja pelo próprio dispositivo que originou o tráfego, seja pelos ativos na infraestrutura de interconexão da rede. A visão geral de um esquema de rotulação de tráfego é mostrada na Figura 2, na qual é destacado um fluxo de rede enviado de um cliente para um servidor. O tráfego é marcado com algum rótulo anexado aos pacotes e o servidor destinatário desse tráfego gerencia um catálogo dessa rotulação.

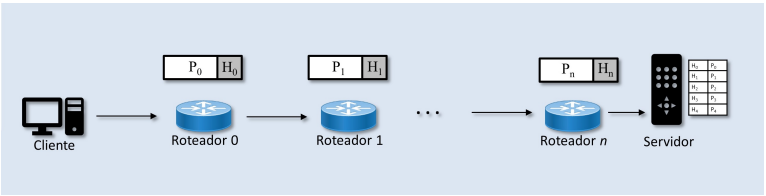


Figura 2 – Visão geral do processo de rotulação de pacotes.

Nas subseções a seguir serão discutidos alguns dos principais métodos de rotulação de tráfego para identificação da fonte na defesa contra ataques *DoS*.

Defesas baseadas em filtro de Bloom

Filtro de Bloom é uma estrutura de dados que representa com eficiência de espaço e tempo de consulta a relação de pertencimento de um elemento a um conjunto. Foi originalmente proposto por Bloom [86], e em sua forma mais clássica pode ser implementado como um mapa de bits cujos valores são inicialmente todos zero. Conforme pode se ver na Figura 3, para se incluir um elemento x em um filtro $\mathcal{F}$, é necessário se aplicar um conjunto $\{hash_i(\cdot)\}$ de funções de *hash*, com cada i-ésima aplicação modificando o mapa de bits por combinar a saída de cada *hash* com o valor atual do mapa por um ou exclusivo lógico: $\mathcal{F} = \mathcal{F} \oplus_i hash_i(x)$. Há uma probabilidade de um falso positivo, ou seja, a chance de o filtro responder que um elemento está incluso, quando de fato não o foi, não é nula

e vai depender dos parâmetros de segurança. Por sua vez, inexiste a chance de um falso negativo, ou seja, de que o filtro sinalize a não inclusão de um elemento quando realmente ele o foi.

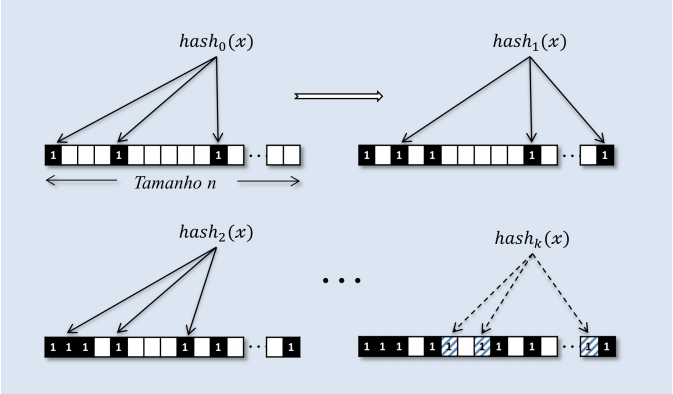


Figura 3 – Visão geral do mecanismo de inclusão em um filtro de bloom.

A estrutura vem sendo aplicada a diversos problemas em redes: roteamento, gerência de *multicast* [87] e de qualidade de serviço. E também na otimização de *web proxy cache* e na mitigação de ataques *DoS* [88].

O trabalho em [89] propõe um meio de detecção de intrusão contra ataques *DoS* na camada de aplicação, ou seja, visando a proteção de serviços na rede. A proposta faz uso de filtros de Bloom para se construir um esquema baseado na detecção de tráfego anômalo, fazendo comparação a um histórico de tráfego previamente analisado.

Os autores em [90] aplicam um filtro de Bloom na construção de um esquema de identificação de quem está originando tráfego na rede. Embora seja alegado que o esquema é resistente a ataques *DoS*, os autores não apontam como um adversário malicioso seria impedido de incluir sua identificação, vez que a inclusão num filtro de Bloom apenas seria segura se fosse utilizado algoritmos criptográficos de funções *hash*.

Recentemente, tem-se discutido a eficácia e segurança desses filtros [88, 91]. Um primeiro ponto é o de inexistir um meio com complexidade computacional linear que proporcione remoção de um elemento do filtro [92]. Um segundo ponto é que o tamanho do filtro deve ser adequado à entropia do tráfego enviado, vez que filtros muito pequenos impõem uma chance maior de falsos positivos. Já filtros grandes podem criar um custo adicional em complexidade de computação ou de armazenamento e transmissão e pode permitir o adversário usar o fato de que um certo registo não se encontra no filtro, o que pode o dar alguma vantagem [91].

Defesas baseadas em rastreamento do tráfego usando criptografia

Na literatura são encontradas propostas de defesa contra ataques *DoS* com a abordagem de se identificar a origem do tráfego (*IP traceback*) [93, 94] com uso de ferramentas

criptográficas [95, 49, 45, 48, 96].

O trabalho em [97] apresenta um estado da arte das propostas que usam criptografia para proporcionar segurança no ambiente de computação em nuvem. Entre as elencadas, o uso de criptografia é principalmente por aplicar algoritmos de *hash* para propor um mecanismo de autenticação de tráfego apropriado para computação em nuvem. Da mesma forma, um estado da arte que elenca soluções que mitigam *DoS* em redes de sensores sem fio, apontando as que utilizam alguma ferramenta de criptografia, é apresentado em [61].

Muitos dos esquemas que permitem rastreamento de tráfego baseados em criptografia seguem o paradigma de habilitação, que, como já foi discutido anteriormente, recebe criticismo sobre se por si sós são suficientes para mitigar ataques de negação de serviço [98]. Os mesmos autores [99] questionam também as soluções fundamentadas apenas em filtragem de tráfego. Também muitos desses esquemas baseados em criptografia de fato estabelecem uma defesa baseada em habilitação para filtragem, ou seja, a filtragem não é apenas baseada nas características intrínsecas do tráfego, mas o é baseada nelas e no que é informado na habilitação trazida.

As recentes propostas baseadas em habilitação, aliadas ou não a filtragem, têm apontado [39, 30, 99] que estes esquemas são apropriados como mitigação de ataques de negação. E as tais têm-se mostrado mais adequadas a alguns tipos em detrimento de não ser tão eficazes em relação a outros, apontando que mais pesquisa deve ser realizada em relação a este paradigma.

3 Uma definição formal de segurança

A primeira definição formal do que é um protocolo seguro surgiu com o conceito de *segurança semântica* (*provable security*) [100]. Intuitivamente, aquela formulação captura a noção da *inviabilidade* de se aprender algo do texto cifrado, ou seja, de não haver possibilidade de se extrair algum conhecimento sobre um texto em claro por se analisar apenas o texto cifrado.

Nas subseções seguintes, serão apresentados os fundamentos teóricos necessários à formulação de uma conjectura da segurança do protocolo proposto neste trabalho, descrevendo-se alguns conhecidos paradigmas de análise de segurança de sistemas.

3.1 O poder computacional do adversário

Na análise de segurança de um protocolo e mais especificamente na classificação de quais os possíveis adversários devem ser levados em conta, um importante aspecto que deve ser observado é o do poder computacional [101] de que eles dispõem. Esta característica é denominada *complexidade computacional do adversário*, e a literatura comumente a dimensiona seja pela capacidade de processamento, seja pela quantidade de memória e armazenamento, pelo tempo de execução disponível e até mesmo pela capacidade de comunicação. É importante destacar que no que se denomina criptografia moderna [102] o conceito de complexidade computacional do adversário é central, pois a definição de segurança demonstrável é fundamentada nesta ideia de complexidade, tendo criando assim a noção de *prova por complexidade* [101]. Considerando esse aspecto, uma classificação dos modelos de adversários será dada como a seguir.

Adversários ilimitados computacionalmente: são aqueles que dispõem de recursos computacionais ilimitadamente, ou seja, um modelo que supõe não haver margem na capacidade de processamento do adversário. Deste modo, tal modelo considera que adversário pode tudo em termos computacionais: processar e armazenar sem limites. Também, são considerados ilimitados aqueles que podem se comunicar sem restrição.

Os tais adversários ilimitados computacionalmente são os considerados na construção de protocolos incondicionalmente seguros [103, 104]. Por se considerar que quem ataca não é limitado em termos de recursos computacionais, para que um esquema seja desenhado para ser incondicionalmente seguro, ele não pode basear sua segurança apenas em hipóteses computacionais, a exemplo de protocolos que levam em conta a dificuldade de se computar a fatoração de grandes números, ou as hipóteses de Diffie-Hellman, entre outros exemplos.

Adversários com complexidade polinomial ou limitados computacionalmente: diferentemente dos adversários computacionalmente ilimitados, são os adversários que executam em tempo polinomial, ou seja, aqueles que computam com recursos de memória, armazenamento e poder de processamento limitados [105]. Os sistemas criptográficos que consideram tal modelo são denominados *computacionalmente seguros*, pois levam em consideração que hipóteses computacionais são impraticáveis de se resolver, e, portanto, se fundamentam na não existência de soluções computacionais práticas para estes problemas.

3.2 Estratégia do adversário

Na demonstração de segurança de protocolos, outro aspecto fundamental é o da caracterização da estratégia do adversário, ou seja, a modelagem do que, como, quando e por quais meios ele executa o ataque. Neste sentido, as definições clássicas de segurança levam em consideração o adversário não apenas como uma entidade isolada, mas propõem ser conveniente considerá-lo como uma entidade que pode corromper outras partes que executam o protocolo.

Adversários estáticos e dinâmicos — Quando é considerado o momento em que o adversário atua, uma classificação a ser adotada é a que os divide entre *estáticos e dinâmicos*. O adversário *estático* escolhe previamente o subconjunto das partes que ele quer corromper. Portanto, toma controle da comunicação realizada por estas partes, fazendo as vezes delas, assumindo a identidade das tais no protocolo. As partes não corrompidas, e, portanto, honestas, assim permanecem até o fim da execução do protocolo.

Os adversários *dinâmicos*, ou *adaptativos*, podem corromper outros participantes a qualquer momento. A escolha do momento e a escolha de quais partes ele quer corromper é arbitrária. É adaptativa no sentido a poder depender da informação que ele recebeu durante a execução do protocolo. Levando em conta este aspecto, a construção de protocolos seguros resistentes a adversários dinâmicos é mais difícil de configurar em comparação com os adversários estáticos.

Adversários ativos e passivos — Outro parâmetro utilizado para classificar a estratégia do adversário é relacionado a como o adversário age em relação a seguir as regras do protocolo. Ao se considerar esta característica, definem-se duas categorias de adversários: *passivos* (ou *semi-honestos*) e *ativos* (ou *maliciosos*). Os adversários *passivos*, ou *semi-honestos*, cumprem fielmente as regras do protocolo, ou seja, este adversário não tenta quebrar a segurança do protocolo por descumprir a sua especificação. Entretanto, este tipo de adversário ainda pode processar toda a informação coletada na execução do protocolo, inclusive a transcrição da troca de mensagens recebidas e transmitidas por e entre as partes, e eventualmente aprender algo que possa lhe dar alguma vantagem ainda

que em um momento futuro.

Por sua vez, o adversário *ativo*, ou *malicioso*, pode agir arbitrariamente e ordenar que as partes corrompidas por ele possam descumprir o especificado no protocolo. Considerar este tipo de adversário é uma hipótese mais forte, no sentido a ser necessário se construir segurança em relação a qualquer ataque que o adversário possa realizar.

Portanto, porque o adversário malicioso foge arbitrariamente do protocolo, ele poderá corromper partes e escolher ou modificar suas entradas, ordenando que as tais partes cooptadas não usem de fato as entradas que elas escolheram. De igual modo, os resultados que as partes corrompidas deveriam receber serão entregues ao adversário. Enfrentar este modelo de adversário é mais difícil, pois qualquer mecanismo que proporcione que as partes honestas do protocolo enfrentem uma estratégia arbitrária de ataque terá um alto custo computacional ou um alto custo de comunicação.

Adversários encobertos (covert) — Vez que a segurança de protocolos resistentes a ataques de adversários passivos limita o contexto no qual eles podem ser aplicados. Contudo, como já argumentado, proporcionar segurança em relação a adversários ativos tem o preço de se exigir uma solução cara. De fato, é preciso se verificar o comportamento de todas as partes, o que geralmente é garantido por provas de zero conhecimento (*zero knowledge proofs*), o que demanda custo computacional ou de comunicação. Uma definição intermediária é apresentada por Aumann e Lindell [106], em que se propõe considerar um adversário que pode agir maliciosamente, mas ele o fará sabendo que as outras partes poderão flagrá-lo com uma certa probabilidade.

O esquema estabelece uma segurança contra adversários maliciosos, mas sem a necessidade de verificação por prova de zero conhecimento, desonerando, portanto, a solução segura contra adversários ativos. O ponto é que o preço de se verificar o comportamento das partes é diminuído, pois esta tarefa, que é realizada por todos num protocolo resistente a adversários maliciosos, no modelo *covert* é transferida para dinâmica do protocolo. O argumento dos autores da proposta baseia-se na suposição de que frente a chance de ser capturando trapaceando, o adversário seja desestimulado a sair das regras.

3.3 O modelo computacional

É necessário que se defina o modelo de computação a ser utilizado na demonstração de segurança neste trabalho; o que de fato tem uma importância mais abrangente, por ser fundamental em paradigmas de argumentação de segurança que consideram a complexidade computacional das partes.

O paradigma de segurança demonstrável captura a ideia de que tudo que possa ser realizado pelo adversário pode também ser simulado por um processo que executa

em tempo polinomial, e que sintetiza de modo indistinguível o que aquele adversário produziria. O modelo computacional adotado deve evidenciar os recursos disponíveis dos envolvidos: do adversário, do simulador, e das reais partes executando o protocolo.

Em computação, o modelo que consegue capturar esses aspectos é a formalização de computação genérica apresentada por Alan Turing: a denominada *máquina de Turing* [107, 108]. Uma máquina de Turing é uma abstração genérica de computação que captura o poder de computar desde uma simples máquina de calcular até o computador genérico nos moldes da arquitetura moderna de computação (modelo de Von Neumann). O modelo é conveniente na análise de segurança em que a complexidade computacional é um fator avaliado, pois evidencia quanto recurso um determinado problema computacional pode demandar, demonstrando a viabilidade computacionalmente de resolvê-lo.

Outro importante aspecto é que máquinas de Turing formalizam bem o paradigma de sistema reativos [109], ou seja, um sistema em que todo o processamento envolvido está bem definido em operações básicas, como encifrar e decifrar, ou assinar e verificar. Nesse contexto, o modelo é bem adequado por capturar as possibilidades de se representar a condição de que as saídas de uma certa função computada dependem das entradas estabelecidas. Esse paradigma de abstração do que pode ser resolvido computacionalmente é muito conveniente quando se quer demonstrar, por exemplo, a complexidade da comunicação envolvida ou o tamanho de armazenamento necessário para se computar algo.

Como já argumentado, a segurança de sistemas criptográficos é pensada para dois contextos: computação com ou sem recursos ilimitados. Quando se quer considerar adversários computacionalmente limitados, um *parâmetro de segurança* há de ser definido. Sendo assim, a ideia de *polinomial* é central, pois é mensurada por se considerar um polinômio avaliado sobre o parâmetro de segurança, uma medida do que é viável computacionalmente [10, 108, 110]. De igual modo, dá-se a interpretação do que se define *adversário em tempo polinomial*, ou também do que se define espaço de memória polinomial no parâmetro de segurança.

3.4 Segurança na criptografia moderna

Os paradigmas pelos quais se argumentam a segurança de protocolos e dados têm ao longo da história sido definidos sob uma dependência em relação ao conceito de criptografia. Anterior à apresentação dos conceitos mais recentes de segurança, a questão da segurança de protocolos era tratada de maneira empírica, basicamente seguindo-se um ciclo em que primeiro se descobria falhas de segurança, então os sistemas eram modificados, depois novamente atacados.

Os fundamentos de uma formalização de segurança da informação foram introduzidos por Claude Shannon [111], pois foi ele quem primeiro definiu segurança de um

sistema criptográfico segundo os conceitos da teoria da informação. O importante trabalho de Shannon iniciou o estudo da questão segurança e definia um sistema criptográfico segundo ele *perfeito* como aquele que garantiria privacidade incondicional, ou seja, levando em conta que os adversários eram computacionalmente ilimitados.

Primeiramente, há de se observar que os protocolos de criptografia de chave pública não atendiam o conceito de segurança incondicional nos moldes pretendidos por Shannon. Os resultados de Diffie e Hellman [112] e de Rivest, Shamir e Adleman [113] proporcionaram sistemas criptográficos assimétricos, o que permitiu um nível de segurança aceitável em um sistema criptográfico prático, sem que se alcançasse a idealização buscada por Shannon, mas também não demandando complexidade computacional ou de comunicação alta para realizá-los. Tais protocolos inovaram por se basear em hipóteses computacionais, como a dificuldade de se fatorar grandes números. Sendo assim, o que se estabeleceu foi um sistema criptográfico resistente a adversários computacionalmente limitados, muito embora os conceitos Shannon não fossem apropriados para representar esse nível de segurança.

Foi Goldwasser e Micali [100] quem introduziu a ideia de *segurança demonstrável* (*provable security*) que veio a se estabelecer como um conceito de segurança mais adequado à criptografia assimétrica, principalmente por trazer a questão complexidade computacional como um fator a ser levado em conta na definição de segurança. O conceito de segurança apontado por eles afirmava que *[...] dado o texto cifrado, tudo que pode ser eficientemente computado acerca do texto não cifrado, pode ser também eficientemente computado sem o texto cifrado [...]* [100]. A intuição introduzida nesse conceito é a de se demonstrar que a chance de o adversário aprender qualquer informação que possa lhe dar alguma vantagem por se analisar o texto cifrado é semelhante (o conceito de *semelhante* é algo que é demonstrado formalmente) a quando a análise é realizada sem o conhecimento deste texto cifrado, ou seja, por se substituir este texto cifrado por um conteúdo aleatório. Se estes dois cenários oferecem uma equiprovável vantagem ao adversário, então *nada* é possível se extrair do texto cifrado, e desta maneira nenhum adversário pode tirar vantagem por analisá-lo. Intuitivamente, constrói-se a ideia de que o texto cifrado não vaza nenhuma informação diante de um adversário *eficiente*, ou seja, em tempo polinomial.

3.5 Sequência de jogos e a argumentação por redução

Os protocolos resistentes a ataques de adversários que executam em tempo polinomial têm sua segurança baseada em alguma hipótese computacional, considerada difícil de solução, ou suposta não existir algoritmo em tempo polinomial que a resolva. É o caso dos protocolos criptográficos que baseiam sua segurança na dificuldade de se ter uma solução viável para o problema de logaritmos discretos de grandes números. É também

o caso dos sistemas criptográficos que fundamentam sua segurança sobre problemas da classe de complexidade *NP* [1].

Desse modo, percebe-se que no projeto de algoritmos arguidos serem computacionalmente seguros, comumente se supõe não existir solução polinomial para os problemas computacionais nos quais eles são baseados. Sendo assim, a estrutura de argumentação da segurança de tais protocolos geralmente segue a lógica de se demonstrar que se a segurança deles for quebrada, ou seja, se existe adversário que viole a sua segurança, então a estratégia desse adversário poderá ser usada para dar solução ao problema para o qual se supunha não existir solução em tempo polinomial. A esta estrutura dá-se o nome de *argumentação por redução*.

De fato, o argumento reducionista baseia toda a noção moderna de segurança, consolidando-se como a principal metodologia de avaliação de segurança de protocolos atualmente. Na prática, o argumento por redução é realizado por duas maneiras: a primeira abordagem é baseada em teoria dos jogos [114]; a segunda é baseada em simulação [115], a qual será discutida na Seção 3.9.

Na abordagem baseada em sequência de jogos [114, 116, 117], a segurança é formalizada por se modelar o protocolo como um jogo interativo de ataque e defesa disputado entre o adversário e um jogador desafiante, existente apenas para se conjecturar a segurança, pois ele não é associado a nenhum participante real do protocolo. Esse último faz o papel de uma parte honesta e que executa em tempo polinomial com a finalidade de desafiar o adversário. De fato, ele gera todas as chaves utilizadas na execução do protocolo e pode responder a questões demandadas pelo adversário. A segurança é então avaliada por se medir a vantagem do adversário face aos desafios enfrentados por ele no jogo.

3.6 Segurança semântica argumentada por sequência de jogos

A fim de exemplificar como é possível argumentar sobre a segurança de protocolos com uso da técnica da sequência de jogos, pode-se tomar como exemplo um sistema criptográfico assimétrico o qual se quer provar possuir segurança semântica, que é tipicamente argumentada pelo modelo de sequência de jogos [114]. Para o exemplo, será utilizada a seguinte definição de sistema criptográfico assimétrico.

Define-se como uma ênupla $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ um sistema criptográfico de chave pública, isto é:

- KGen sendo um algoritmo que, em tempo polinomial, recebe como entrada o parâmetro de segurança 1^n e gera as chaves pública e privada: $(\mathsf{pk}, \mathsf{sk})$;

[1] Os problemas que pertencerem à classe *NP* são aqueles acerca dos quais se supõe não existir algoritmo que os resolva em tempo polinomial, embora uma solução dada a uma determinada instância do problema possa verificada polinomialmente.

- O algoritmo Enc realiza a função de enciframento, executa em tempo polinomial e recebe como entrada a chave pública pk e o texto em claro a ser cifrado m; e
- O algoritmo Dec realiza a função de deciframento, recebe como entrada a chave privada sk e o texto cifrado σ.

A ideia central de segurança semântica descrita no artigo [100] é a de que o adversário não consegue distinguir eficientemente entre dois textos cifrados criados a partir de textos em claro escolhidos por ele mesmo. O enciframento é realizado pelo desafiante e sob as chaves que este escolheu. Isso é formalmente definido pela seguinte sequência de jogos:

- Desafiante: executa $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^n)$;
- Adversário: escolhe duas mensagens m_0, m_1 e as envia para o desafiante;
- Desafiante: escolhe $b \xleftarrow{r} \{0,1\}$; computa $\sigma_b \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$; envia σ_b para o adversário; e
- Adversário: retorna $\hat{b} \in \{0,1\}$ como saída.

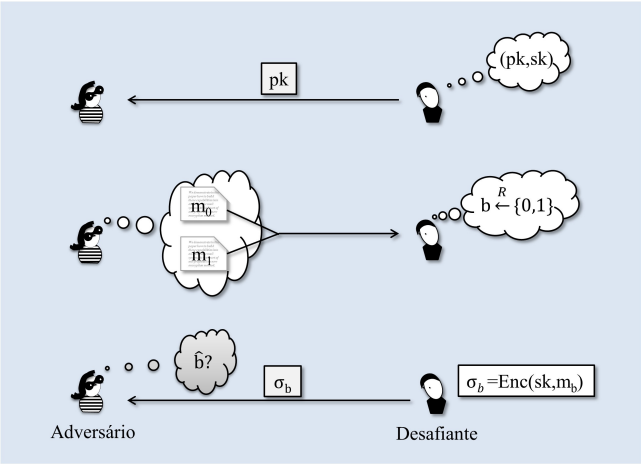


Figura 4 – Jogo sequência na argumentação da segurança semântica de um sistema criptográfico.

O argumento de que o sistema criptográfico é semanticamente seguro reside na conjectura sobre a probabilidade com que o adversário palpitará com sucesso sobre qual dos textos o desafiante cifrou, ou seja, $\Pr\left[\hat{b} = b\right]$. Se nada pode ser aprendido pelo adversário quando ele analisa o texto cifrado (de um dos textos que ele mesmo criou), então ele não pode ter uma performance melhor do que $\Pr\left[\hat{b} = b\right] = \frac{1}{2}$. Ou seja, o adversário não ganha informação que lhe dê mais vantagem em relação ao que ele já sabia antes

de receber o texto cifrado, e sua escolha não seria melhor do que observar dois outros textos cifrados quaisquer, ou de lançar a sorte em um cara ou coroa como uma estratégia igualmente eficiente.

3.7 Argumentação de segurança baseada em oráculo aleatório

O modelo do oráculo aleatório foi introduzido no trabalho de Bellare e Rogaway [118] e é uma das técnicas clássicas de argumentação de segurança na literatura de criptografia. Um sistema criptográfico é dito ter sua segurança conjecturada neste modelo quando todas as partes, incluindo o adversário, interagem mutuamente trocando mensagens de maneira usual, mas adicionalmente eles podem fazer consultas a uma entidade idealizada chamada de *oráculo aleatório*.

O modelo define que as consultas que chegam à entidade oráculo são respondidas por uma única função, denotada por $\mathcal{O}$ e escolhida aleatoriamente no espaço das funções possíveis, determinado por um parâmetro de segurança do sistema [119].

Formalmente, dado o parâmetro de segurança k, $l_{out}(k)$ o tamanho das saídas e uma família de funções $\mathcal{F} : \{0,1\}^* \rightarrow \{0,1\}^{l_{out}(k)}$, um oráculo aleatório $\mathcal{O}$ é uma função escolhida aleatoriamente em $\mathcal{F}$:

$$\mathcal{O} : \{0,1\}^* \rightarrow \{0,1\}^{l_{out}(k)} \tag{3.1}$$

Uma intuição que pode ser tomada para este tipo de construção é imaginar que a cada nova consulta que uma parte pode fazer ao oráculo a respeito de uma entrada x, o oráculo escolhe um $y = \mathcal{O}(x)$ e sempre que esta mesma entrada for novamente consultada, ainda que por uma outra parte diversa daquela que já a consultou, o oráculo retornará o mesmo y como saída. O oráculo $\mathcal{O}$ funciona como uma função puramente aleatória, a menos de receber uma entrada sob a qual dará o mesmo resultado. Na prática, quando se sai do modelo para uma implementação prática do esquema substitui-se o oráculo por uma função apropriada. As funções de *hash* têm sido comumente escolhidas por se considerar que elas são candidatas aceitáveis ao papel de oráculo em muitos contextos de segurança [119].

Embora este método tenha sido utilizado largamente na arguição da segurança de muitos protocolos criptográficos, a literatura discute diversas de suas limitações, principalmente em relação à impossibilidade de se substituir adequadamente o oráculo por algo implementado computacionalmente, ou seja, da limitação de se ter um gerador aleatório baseado em computação. Com efeito, Canetti demonstra [119] que há protocolos para os quais é possível se conjecturar sua segurança pelo método do oráculo aleatório, mas que também deles não se pode ter uma implementação argumentada segura quando o oráculo for substituído por qualquer função computável. Demonstrando assim que a idealização

utilizada pelo modelo não pode ter paralelo no mundo real, pondo em discussão o quão generalizado pode ser o método.

Na argumentação da segurança de protocolos de assinaturas digitais, o modelo de oráculo aleatório tem um papel muito importante, pois muitos dos esquemas práticos têm sua segurança avaliada neste modelo [120]. Estão inclusos neste caso a grande maioria dos esquemas que seguem o paradigma *hash-and-sign*, ou seja, aqueles em que um algoritmo de *hash* é um componente no processo de assinatura.

3.8 Indistinguibilidade computacional

A ideia de *indistinguibilidade* tem um papel central na análise de segurança em criptografia moderna. Ela está intimamente relacionada a noção da impossibilidade do que pode ser aprendido por se analisar um sistema criptográfico, como já discutido em seções anteriores. Está também relacionada à noção da argumentação da segurança de protocolos nos paradigmas em que a complexidade computacional é considerada. É possível se demonstrar, sob os conceitos da teoria da informação, que nada se pode extrair eficazmente de uma informação que computacionalmente indistinguível de algo que foi gerado aleatoriamente. O paradigma de argumentação por simulação a ser apresentado na Seção 3.9 se fundamenta nestas noções e se baseia nas definições apresentadas a seguir.

Seja I um conjunto contável de índices. Um *ensemble* de probabilidades é uma sequência infinita de variáveis aleatórias indexada por um conjunto contável I. A saber, sendo X_i variável aleatória, $X = \{X_i\}_{i \in I}$ é um ensemble indexado por I.

Definição 1. *Uma função $\mu(\cdot) : \mathbb{N} \to [0, 1]$ é chamada de negligível se para toda função polinomial positiva $p(\cdot)$ e para todo $n \in \mathbb{N}$ suficientemente grande, vale a desigualdade $\mu(n) < 1/p(n)$.*

A definição negligibilidade acima apresentada serve para definir o que é *pequeno* em termos de probabilidade no contexto de segurança: a exemplo de quando a chance do adversário distinguir entre duas variáveis for *negligível* em função do parâmetro de segurança utilizado. Intuitivamente, uma função negligível é aquela que decresce mais rápido do que qualquer polinômio pode crescer. Uma noção conveniente para se mostrar que a chance de um adversário poderá ser sempre pequena a despeito do quanto ele pode crescer sua capacidade computacional, desde que execute em tempo polinomial.

Definição 2. *Ensemble indexado por $\mathbb{N}$: Dois ensembles $X \stackrel{def}{=} \{X_n\}_{x \in \mathbb{N}}$ e $Y \stackrel{def}{=} \{Y_n\}_{y \in \mathbb{N}}$ são indistinguíveis em tempo polinomial se para todo algoritmo D que executa em tempo polinomial, para todo polinomial $p(\cdot)$ e para todo n suficientemente grande:*

$$|\Pr[D(X_n, 1^n) = 1] - \Pr[D(Y_n, 1^n) = 1]| < \frac{1}{p(n)} \tag{3.2}$$

Definição 3. *Dois ensembles* $X \stackrel{def}{=} \{X(a,n)\}_{a\in\{0,1\}^*,n\in\mathbb{N}}$ *e* $Y \stackrel{def}{=} \{Y(a,n)\}_{a\in\{0,1\}^*,n\in\mathbb{N}}$, *são ditos computacionalmente indistinguíveis, o que é denotado por* $X \stackrel{c}{\equiv} Y$, *se para todo algoritmo que executa em tempo polinomial,* D, *e para todo* $a \in \{0,1\}^*$ *tal que para todo* $n \in \mathbb{N}$.

$$|\Pr[D(X(a,n)) = 1] - \Pr[D(Y(a,n)) = 1]| < \mu(n) \tag{3.3}$$

Estas definições formalizam o que é indistinguibilidade computacional de duas variáveis aleatórias, a idealização aqui descreve uma entidade que faz o papel de um julgador D (discriminador), alguém que é desafiado a separar os valores gerados pelas duas variáveis aleatórias, X e Y.

Uma exemplificação de aplicação destes conceitos pode ser dada por se considerar a argumentação da segurança semântica apresentada na Seção 3.6. Tomando o adversário $\mathcal{A}$ como aquele que faz as vezes do discriminador D, em que rnd é um texto escolhido aleatoriamente, supondo que σ_b e rnd possuem o mesmo tamanho: $|\sigma_b| = |\mathsf{rnd}|$, a probabilidade de que ele ganhe alguma vantagem em reconhecer o texto cifrado seja pequena, ou seja, que a vantagem a seguir seja negligível:

$$|\Pr[\mathcal{A}(\sigma_b) = b] - \Pr[\mathcal{A}(\mathsf{rnd}) = b]| \tag{3.4}$$

3.9 Demonstração da segurança com o paradigma de simulação

Um mecanismo de argumentação de segurança, frequentemente encontrado na literatura atual, é o considerado no modelo de simulação [115, 121]. De fato, a definição vem do paradigma de indistinguibilidade apresentado anteriormente. Uma interpretação que pode ser dada àquele modelo é de existir dois mundos: o mundo *real* e o *ideal*. A indistinguibilidade estaria então a cargo de ser alegada por um discriminador, caso não fosse capaz de em tempo polinomial apontar qual variável aleatória é a do mundo real e qual adveio do mundo ideal.

Esta nova visão é adequada à argumentação da segurança de protocolos, pois quando se modela um protocolo como um sistema reativo, como o exposto na Seção 3.3, torna-se conveniente a análise da transcrição da sua execução, considerando-a uma variável aleatória.

Deste modo, tomando-se o exemplo de duas distribuições de variáveis aleatórias, P_{ideal} e P_{real}, que correspondem às transcrições do protocolos em dois cenários: um no qual as partes fazem acesso a uma terceira parte confiável, e o outro no qual esta terceira parte não existe e o acesso a funcionalidade é substituído por um subprotocolo executado entre as partes. Denomina-se então esses dois cenários *ideal* e *real* respectivamente.

Então, sendo k o parâmetro de segurança do protocolo, o que se almeja é que seja negligível a seguinte diferença:

$$|\Pr[D(P_{ideal}) = 1] - \Pr[D(P_{real}) = 1]| < \mu(k) \tag{3.5}$$

O paradigma de simulação não está explicitamente formalizado nesta construção por indistinguibilidade. Mas a ênfase está em se comparar o que um adversário pode usar para atacar uma execução real de um protocolo e o que poderia ocorrer num cenário ideal, onde a execução da função é, por hipótese, segura. A construção baseia-se na análise das mensagens trocadas entre as partes, pois a transcrição destas tais mensagens compõe a visão que o adversário tem da execução do protocolo. O que é possível se argumentar é que se estas duas visões são indistinguíveis computacionalmente, então o adversário não terá nenhuma vantagem.

Uma maneira de construir tal comparação é a de definir um simulador que emule a comunicação (transcrição das mensagens entre as partes) de modo não seja viável distinguir a que contexto pertence a comunicação: se ao real ou ao ideal. Diz-se que para um determinado protocolo se é possível construir um simulador que realize tal emulação, então a visão que o adversário possui do protocolo é independente das entradas, e ele ganha nenhum aprendizado a respeito delas.

Uma ilustração destes conceitos pode ser dada pelo seguinte cenário hipotético ilustrado na Figura 5. Imagine-se um protocolo em que múltiplas partes cooperam para que uma certa funcionalidade F seja executada entre eles (F pode ser uma função distribuída, tal como uma eleição, um leilão, etc.). No mundo ideal de execução desta funcionalidade, as partes a realizam por acessarem uma terceira parte confiável, que faz as vezes desta funcionalidade ideal, confiável e incorruptivelmente. Assim, as partes simplesmente entregam suas entradas a esta confiável, que avalia a função e retorna o resultado às partes correspondentes em segredo. Este contexto é chamado por *Mundo ideal* na Figura 5.

Por sua vez, o *Mundo real* é aquele em as partes não contam com a terceira parte confiável, a qual e é de fato realizada por um protocolo distribuído G que as partes executam.

A estrutura de argumentação acima é apropriada para se demonstrar que a segurança da execução do protocolo no mundo real é a mesma da especificada no mundo ideal. Importante dizer que quando há uma terceira parte confiável, é trivial se realizar qualquer funcionalidade distribuída com segurança. Exatamente porque esta terceira parte é considerada incorruptível. Porém, esta suposição é meramente uma idealização, e na prática a execução de protocolos seguros querem fugir da dependência de uma terceira parte. De qualquer modo, como idealização, o contexto de execução do mundo ideal é uma referência quando se quer medir a segurança de um protocolo executando no mundo real. Então, um argumento pode ser construído por se alegar que a visão da interação,

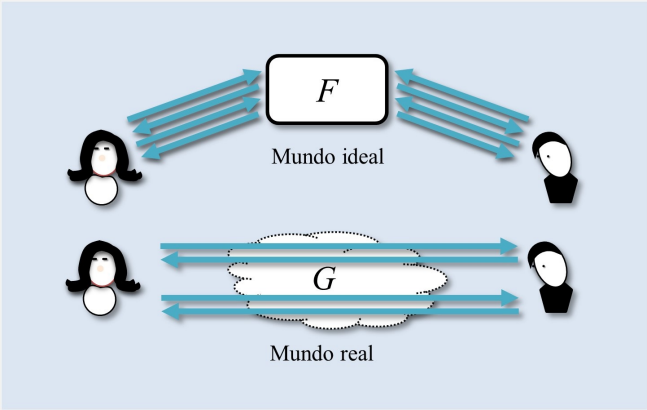


Figura 5 – Comparação da idealização do protocolo no mundo *ideal* e o protocolo realizado no mundo *real*.

denotada aqui View, entre as partes e a uma terceira parte confiável F é indistinguível da interação entre as partes que executam de modo distribuído o protocolo G, realização da funcionalidade ideal F.

Deste modo, denominando-se x e y as entradas respectivas das partes, e $\mathsf{View}^F(x, y)$ a visão do protocolo executado no mundo ideal, e $\mathsf{View}^G(x, y)$ a visão do mundo real, o que se deseja demonstrar é que elas são indistinguíveis computacionalmente, a menos de uma chance negligível:

$$\mathsf{View}^F(x, y) \stackrel{c}{\equiv} \mathsf{View}^G(x, y) \tag{3.6}$$

Outro exemplo será apresentado aqui a fim de se argumentar sobre privacidade de um protocolo. Então, considere-se uma função $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$. As respectivas entradas das partes são $x, y \in \{0,1\}^*$ e a saída as variáveis aleatórias $(f_1(x, y), f_2(x, y))$, sendo que $f_1(x, y)$ é o valor retornado para a primeira parte, assim como $f_2(x, y)$ para o segunda. Então a privacidade é demonstrada por se provar que esta visão pode ser simulada a partir das entradas e das saídas que uma parte ingressa e recebe no protocolo.

Uma argumentação pode ser alcançada por se construir um simulador que imite a visão de uma parte a partir de entradas e saídas coerentes com a função sendo analisada. A existência de um simulador como este garante que esta visão que uma parte tem pode ser emulada, significando que tudo que ela pode aprender depende apenas das entrada e saída que ela já conhece. Entretanto, essa limitação vale também para o adversário, o que é interessante para construção da privacidade do protocolo que realiza a função.

Quando se leva em consideração que ao se demonstrar que tudo que um adversário vê, ou seja, a distribuição das mensagens que ele pode coletar — o que usado para que ele ataque o protocolo — pode ser construída por um simulador, quando de fato qualquer pessoa pode estar realizando esta simulação, é intuitivo pensar que este adversário

não ganha nenhuma vantagem além do que se ele estivesse analisando qualquer outra informação, ainda que gerada aleatoriamente.

Por outro lado, se não existe um simulador que imite o adversário, então neste caso existe um discriminador D que em tempo polinomial pode separar a execução no mundo real da no mundo ideal, e então poderia se concluir que o adversário tem realmente algum poder para extrair alguma vantagem em analisar as mensagens trocadas na execução do protocolo.

A intuição apresentada neste exemplo é formalizada em [10, 121], o que se tornou um paradigma de como se argumentar da privacidade de um protocolo, conhecido por *modelo de simulação*, e é apresentado pela seguinte definição de lá transcrita:

Definição 4. *[10] Seja $f(x, y)$ uma funcionalidade. Diz-se que Π computa com privacidade f, se existem algoritmos que executam em tempo polinomial denotados $\mathsf{Simul}_1, \mathsf{Simul}_2$, tais que são mantidas computacionalmente as seguintes indistinguibilidades:*

$$\{\mathsf{Simul}_1(1^n, x, f_1(x, y))\}_{x,y\in\{0,1\}^*} \stackrel{c}{\equiv} \{\mathsf{View}_1^{\Pi}(x, y, n))\}_{x,y\in\{0,1\}^*} \tag{3.7}$$

$$\{\mathsf{Simul}_2(1^n, y, f_2(x, y))\}_{x,y\in\{0,1\}^*} \stackrel{c}{\equiv} \{\mathsf{View}_2^{\Pi}(x, y, n))\}_{x,y\in\{0,1\}^*} \tag{3.8}$$

em que $|x| = |y|$, e $n \in \mathbb{N}$.

Na Figura 6, é ilustrado um cenário do mundo real em que o adversário já controlou uma das partes, deste modo tomando conta da comunicação que essa parte realizaria: substituindo entradas e recebendo saídas, enfim fazendo as vezes destas partes por ele controlada.

Esta visão do protocolo no mundo real pode ser simulada por se emular a execução do protocolo no mundo ideal (com acesso à F) em que a parte corrompida é substituída pela estratégia do adversário. Nesta emulação, o simulador irá alimentar a parte honesta (emulada) com entradas e saídas que ela espera.

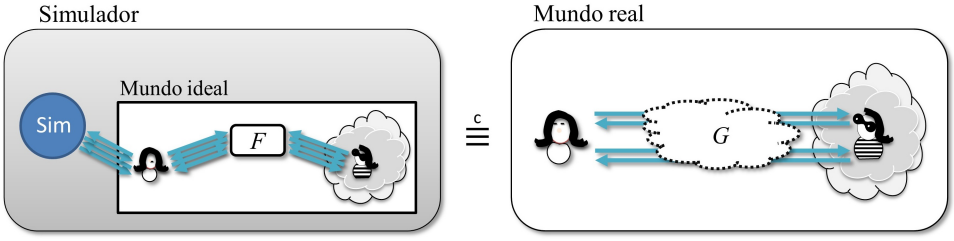


Figura 6 – Visão geral do simulador emulando o protocolo no mundo ideal a fim de criar uma visão indistinguível do mundo real.

Então os valores escolhidos pelo simulador serão submetidos a funcionalidade ideal, que responderá naturalmente. Porém, a reação do adversário, ou seja, a estratégia que ele fará uso para atacar o protocolo poderá ser copiada pelo simulador para ser utilizada na criação de uma visão indistinguível do protocolo real.

Este é o paradigma de argumentação da segurança em [10, 121, 107, 108, 115, 122], para citar alguns dos trabalhos que fundamentam esses conceitos.

4 Ferramentas criptográficas

Neste capítulo, serão apresentadas as ferramentas criptográficas a serem utilizadas neste trabalho. Aqui serão discutidos os conceitos de assinatura digital e como eles estão relacionados à ideia de autenticação de mensagens.

4.1 Assinatura digital e autenticação de mensagens

Uma assinatura digital é um mecanismo de *prova interativa*[1] que permite uma parte, denominada *assinador* (ou *assinante*), convencer um conjunto de outras partes, os *verificadores*, acerca da veracidade de uma declaração. Funciona como um análogo a uma assinatura convencional, escrita a punho e apensada a um documento. De fato, pode ser implementada como uma função criptográfica que pode prover um nível de segurança muito mais forte do que aquela versão escrita.

Um conceito similar é o de *autenticação de mensagem*, que também é um protocolo interativo que proporciona se verificar a validade de um conteúdo digital, ou seja, que um determinado conteúdo foi homologado ou aprovado por uma determinada entidade, pessoa [121].

Uma assinatura digital deve ser universal no sentido a permitir que qualquer pessoa seja capaz de verificar a validade do que está sendo provado. Sendo assim, não considera a segurança do canal, ou seja, não se baseia na privacidade do que é transmitido. Um dos seus requisitos fundamentais é que elas devam ser resistentes a ataques em que o adversário tenta forjá-la. Na demonstração de sua segurança, é requerido que:

- Um assinante possa criar eficientemente uma assinatura digital de qualquer documento, de tal modo que qualquer um possa eficientemente verificar a validade desta assinatura; e
- Que não seja viável computacionalmente um adversário forjar uma assinatura em nome de outrem, ou seja, que não seja viável se criar uma assinatura de um documento a qual ninguém tenha ainda criado.

Por sua vez, métodos de autenticação levam em conta a insegurança do canal e consideram se há privacidade do que é transmitido. Estes são verificáveis apenas pelas partes que se comunicam, sendo preciso a interação com aquele que criou a autenticação para que ela seja verificada. Em suma, a informação de autenticação de uma mensagem

[1] Uma *prova interativa* não é uma prova no sentido clássico da matemática. *Grosso modo*, é um esquema em que partes interagem para que umas convençam outras sobre a verdade de alguma alegação.

só é verificada se quem a criou permitir. Esquemas de autenticação, portanto, devem satisfazer os seguintes requisitos [121]:

- Cada uma das partes comunicantes pode eficientemente criar autenticação de qualquer mensagem de sua escolha;
- Cada uma das partes comunicantes pode verificar eficientemente a informação de autenticação de qualquer mensagem; e
- É inviável computacionalmente que qualquer parte externa (não pertencente ao conjunto das partes comunicantes) crie uma mensagem de autenticação.

Mais formalmente, uma assinatura digital é definida como a seguir:

Definição 5. *[123] Uma assinatura digital consiste em uma ênupla de algoritmos probabilísticos* $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *que executam em tempo polinomial e têm como domínio um espaço de mensagens* $\mathcal{M} = \{M_k\}$*, tal que*

- *O algoritmo de geração de chaves* KGen *recebe como entrada o parâmetro de segurança* k *e retorna como saída um par de chaves* $(\mathsf{pk}, \mathsf{sk})$*, tal que* pk *é denominada chave pública ou chave de verificação,* sk *é a chave privada ou chave de assinatura;*
- *Para um parâmetro de segurança* k*, o algoritmo de assinatura* Sign *(possivelmente probabilístico) recebe como entrada a chave privada* sk *e uma mensagem* $m \in M_k$ *e retorna como saída a assinatura* σ*. É convencionada a notação:* $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(m)$ *e é assumido que se* $m \notin M_k$ *o algoritmo retorna o símbolo* $\perp$*; e*
- *Para um parâmetro de segurança* k*, o algoritmo determinístico de verificação* Vrfy *toma como entrada a chave pública* pk*, uma mensagem* $m \in M_k$ *e uma assinatura* σ*. O resultado é um único bit* b*, com* $b = 1$ *significando aceitar e caso contrário* $b = 0$ *significando rejeitar. É convencionada a notação* $b = \mathsf{Vrfy}_{\mathsf{pk}}(m, \sigma)$*.*

A definição acima formaliza completamente uma assinatura digital e fundamentará a definição de outros esquemas de assinatura que estendem este conceito clássico. Na prática, o que um assinador de uma mensagem deve fazer é invocar a geração de chaves KGen e receber um par de chaves $(\mathsf{pk}, \mathsf{sk})$ e uma vez de posse delas assinar qualquer mensagem a sua vontade: $\sigma \leftarrow \mathsf{Sign}_{\mathsf{pk}}(m)$, então ele envia (publica) o par (m, σ) que pode ser verificado abertamente, vez que para isso é necessário apenas que se verifique $\mathsf{Vrfy}(m, \sigma) \stackrel{?}{=} 1$, conforme ilustrado na Figura 4.1 (com a omissão da invocação da geração de chaves KGen).

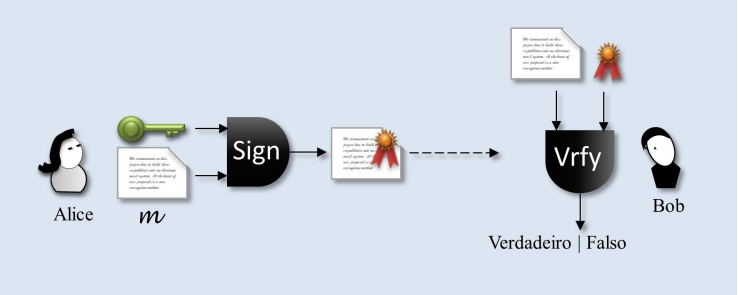


Figura 7 – Visão geral do funcionamento de um esquema de assinatura digital convencional.

4.2 Definição de segurança de assinaturas digitais

Uma outra importante diferença entre esquemas de assinatura digital e algoritmos de autenticação é que estes podem ser incondicionalmente seguros, ou seja, há autenticação resistente a adversários computacionalmente ilimitados, ainda que seja preciso limitar o número de mensagens autenticadas [123]. Por outro lado, algoritmos de assinatura digital são computacionalmente seguros inerentemente, ou seja, não há versões incondicionalmente seguras desses esquemas. Uma intuição de que é impossível tê-los em versões incondicionalmente seguras pode ser dada pelo seguinte exemplo: a um adversário com poder computacional ilimitado são dados um texto m e a chave pública pk de um assinante. É certo que o assinante poderia gerar a assinatura da mensagem m, então mesmo que ele não o tenha feito, ela poderá eventualmente ser encontrada pelo adversário: ele pode então escolher assinaturas σ e verificar $\mathsf{Vrfy}_{\mathsf{pk}}(m, \sigma) = 1$. Deste modo, apesar de não saber a chave secreta do assinante o adversário pode verificar mensagens, e como possui poder ilimitado poderá encontra com chances não negligíveis uma assinatura válida.

Outro fato é que nenhuma assinatura digital seria incondicionalmente segura, mesmo quando se considera adversários limitados computacionalmente. Isso porque quando um adversário escolhe aleatoriamente uma assinatura digital qualquer σ, a chance de que seja uma assinatura válida para m não é nula, ou seja, não é negligível. De fato, qualquer um pode fazê-lo, mesmo sem o conhecimento da chave pública do assinante e considerando, portanto, um adversário muito menos municiado.

A definição formal de segurança de assinaturas digitais é fundamentada principalmente num requisito: incapacidade do adversário forjar uma assinatura, como já discutido. O que se requer é que o adversário não seja capaz de forjar em tempo polinomial qualquer assinatura que possa ser verificada como válida relativa a uma mensagem a qual o detentor da chave privada correspondente não tenha assinado. Outro aspecto é que se requer que o adversário seja incapaz de criar eficientemente uma nova assinatura válida para um documento, ainda que não tenha como alvo um documento em específico, ou seja, criar uma

assinatura válida para um documento aleatório. O primeiro ataque está no denominado contexto de *falsificação universal*, o segundo está entre os ataques *falsificação existencial*, sendo que a resistência contra este último é uma suposição mais forte. Formalmente, uma definição de segurança de assinaturas digitais pode ser formulada como a seguir:

Definição 6. *[124] Um esquema de assinatura digital é seguro se uma falsificação existencial é impossível computacionalmente, até mesmo sob ataques adaptativos de texto cifrado escolhido (*IND-CCA2*).*

A argumentação da segurança de esquemas de assinaturas digitais resistentes a ataques IND-CCA2, pode ser construída com o seguinte jogo entre o adversário $\mathcal{A}$ e um desafiante [125]:

- *Configuração*: o desafiante invoca o algoritmo de geração de chaves KGen do qual recebe sk e pk respectivamente chaves pública e privada; o adversário $\mathcal{A}$ recebe a chave pública pk;
- *Consultas*: o adversário $\mathcal{A}$ escolhe n mensagens, a sua vontade, $m_1, ..., m_n$, e pede ao desafiante as assinaturas correspondentes $\{\sigma_1, ..., \sigma_n\}$, em que $\sigma_i = \mathsf{Sign}(\mathsf{sk}, m_i)$; e
- *Saída*: o adversário $\mathcal{A}$ vence o jogo quando eventualmente ele retorna como saída o par (M, σ), em que M não foi uma consulta no passo anterior.

4.3 Assinaturas digitais não repudiáveis

Como uma extensão à ideia original de assinatura digital, Chaum e Antwerpen [125] propuseram a ideia de assinaturas *não repudiáveis* (*undeniable signatures*), com as quais a tarefa de verificação depende da cooperação do assinante para que a assinatura seja verificada. Ou seja, sem interação com o assinante, não é possível se verificar a assinatura. Deste modo, as assinaturas não repudiáveis divergem dos esquemas clássicos em que assinaturas podem ser verificados por qualquer um que receba o documento assinado, ainda que contra a vontade de quem assinou. Tais esquemas têm um nicho de aplicação, pois se apresentam mais adequados a aplicações das quais é requisito gerenciar quem pode verificar documentos assinados, por exemplo, quando há uma questão de privacidade e o assinante não deseja que publicamente se conheça que ele assinou um documento.

A Figura 8 apresenta uma visão geral da dinâmica envolvida em assinaturas não repudiáveis. Na ilustração, em (a) o assinante deseja enviar um documento para um destinatário específico, mas ainda não houve nenhuma interação entre eles. Quando então a parte verificadora o recebe, ainda não poderá realmente verificar sua assinatura. Neste ponto, assinatura digital é indistinguível de qualquer outra sequência aleatória. Apenas

depois que o assinante concorda em interagir com o verificador, em (b), enviando-lhe informação necessária ao reconhecimento da assinatura, é que ele poderá de fato verificá-la.

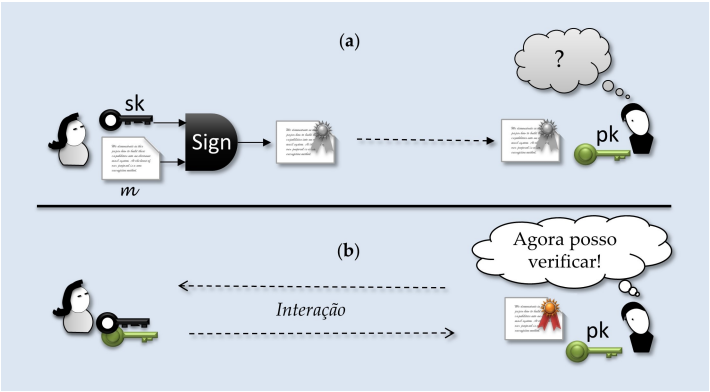


Figura 8 – Dinâmica existente na verificação de uma assinatura não repudiável.

Originalmente, a proposta de assinaturas não repudiáveis era a de permitir que um assinante tivesse controle sobre quem poderia verificar a assinatura de um documento. Entretanto, o assinante não ganha total controle de quem realmente pode verificar, nem muito menos de quantas vezes esta verificação pode ser feita: um receptor malicioso pode simplesmente repassar um documento assinado. Com o protocolo, o assinante irá controlar o tempo em que a assinatura poderá ser feita, mas não há domínio total sobre quem pode verificar, como foi originalmente proposto pelos autores. Uma desvantagem desse esquema é que ele requer interação, o que pode demandar uma complexidade de comunicação não apropriada a algumas aplicações.

4.4 Assinaturas digitais designáveis

Jackobsson, Sako e Impagliazzo [126] introduziram a ideia de *assinatura digital designável — DVS* (*Designated Verifier Signature*). Este novo conceito estende o de assinaturas não repudiáveis no sentido a permitir que um assinador de mensagem possa escolher quem efetivamente pode verificá-la. Ademais, o esquema proposto não exige interação com o assinante, o que pode ser interessante para muitos problemas de segurança.

Diferente dos esquemas tradicionais de assinatura, as designáveis impõem restrições a um Bob malicioso receptor de uma assinatura criada por Alice. Caso ele tente repassá-la a uma terceira parte, Cindy, esta não poderia se convencer da veracidade da assinatura, pois a visão que ela teria da assinatura é de esta ser indistinguível de qualquer outra sintetizada por Bob. Ou seja, a assinatura designada a Bob pode apenas ser por ele verificada e apenas ele pode acreditar na demonstração que Alice o enviou. De fato, a assinatura designável é criada em dependência a uma chave secreta que Bob possui.

O importante é que ainda que Bob revele sua chave secreta, Cindy não teria evidências de que ela não foi manufaturada maliciosamente por ele. De qualquer modo, o que resta a Cindy é acreditar num Bob malicioso, ou seja, se ele realmente não repassou uma assinatura manipulada. Então, se Cindy decidir acreditar em Bob, ela estará fazendo por convicções próprias.

Dois cenários que devem ser analisados são os como a seguir. Primeiro, imagine-se que uma Cindy maliciosa invada o computador de Bob e encontre documentos assinados digitalmente por Alice, que usou um esquema de *DVS*. Uma questão crucial é a de haver evidências para que Cindy acredite nessas assinaturas, ou seja, para creditá-las como autenticamente geradas por Alice. Não há uma resposta absoluta. O ponto é que Cindy pode crer perfeitamente que Bob não manipula as assinaturas nem suas chaves privadas, o que faria Cindy apostar na veracidade das assinaturas encontradas e assim formar suas convicções. De qualquer modo, não haveria evidências para se alegar que elas são legítimas, ou seja, levando-se em conta a informação digital encontrada, ela não poderia demonstrar para outros que aquilo foi assinado legitimamente.

O segundo exemplo leva em consideração uma Cindy que estivesse escutando promiscuamente o canal de rede de Alice, e a visse enviar documentos assinados digitalmente usando um esquema *DVS*. Até se poderia pensar que haveria mais elementos para se usar como evidências de que aquelas assinaturas eram legítimas, e frente a uma Alice honesta interessada na integridade das suas assinaturas isso seria realmente uma preocupação. Um exemplo deste contexto seria o de um esquema de autenticação: Alice querendo se autenticar para com Bob. Cindy não poderia verificar a assinatura, mas poderia capturá-la e retransmiti-la para se passar por Alice e eventualmente se autenticar perante Bob. Formalmente, esquemas *DVS* são definidos como a seguir:

Definição 7. *Assinaturas digitais designáveis [126] — Seja* (P_A, P_B) *um protocolo em que Alice prova para Bob a veracidade de uma declaração* Θ*. Diz-se que Bob é um verificador designado e a seguinte relação se mantém: para todo protocolo* (P_A, P'_B, P_C) *entre Alice, Bob e Cindy, em que Bob prova* ϑ *para Cindy, existe um outro protocolo* (P''_B, P_C) *tal que Bob pode executar* P''_B *tal que Cindy não seja capaz de distinguir a transcrição de* (P_A, P'_B, P_C) *da em* (P''_B, P_C)*.*

A definição acima pode ser interpretada pela seguinte sequência de jogos, ilustrada na Figura 9: qualquer que seja a informação que Cindy pode capturar da visão que ela tem de qualquer protocolo executado entre Alice e Bob não irá ajudá-la a distinguir entre uma evidência legítima e uma manipulada ou sintetizada por Bob. Ou seja, ainda que Bob envie para Cindy a transcrição de toda informação trocada com Alice na execução do protocolo (P_A, P_B), isto não a ajudará a se convencer sobre a prova de ϑ.

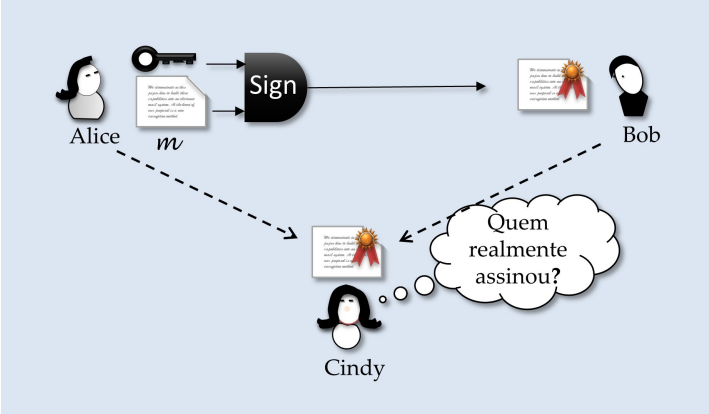


Figura 9 – Indistinguibilidade proporcionada por assinaturas designáveis.

4.5 Assinaturas digitais fortemente designáveis

O conceito de *assinaturas fortemente designáveis — SDVS* (*Strong Designated Verifer Signature*) veio para estender a ideia em *DVS* e proporcionou um esquema que considera um contexto em que não só Bob pode tentar trapacear, mas leva também em conta que Alice pode vazar a assinatura de um documento. Neste sentido, com o novo esquema não só Bob (o receptor designado), mas também qualquer um pode criar uma assinatura indistinguível da legitimamente endereçada a Bob, como ilustrado na Figura 10. Essa habilidade amplia as dúvidas de uma Cindy curiosa em relação a quem de fato assinou o documento. Com efeito, não só Bob poderá equivocar Cindy, mas também Alice e, de fato, qualquer outro poderá fazê-lo.

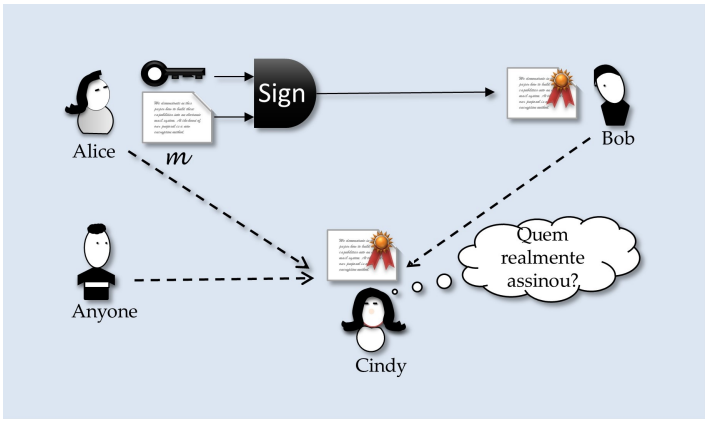


Figura 10 – Visão geral da propriedade de equivocação existente em um esquema *SDVS*.

O que de fato muda com este cenário é que ainda que uma Cindy maliciosamente invada o computador da Alice ela não teria evidências (que pudessem ser usadas para convencer terceiros) de que Alice realmente criou as assinaturas de documentos encontradas no computador invadido. Ou Cindy teria que crer, simplesmente porque a encontrou

no computador de Alice. De qualquer forma, Cindy teria a mesma dúvida se ela estivesse espiando maliciosamente o canal de rede entre Alice e Bob. Um cenário interessante seria o de quando Alice e outros estivessem do mesmo lado da rede e todos assinando documentos para Bob, usando o esquema *SDVS* e usando um canal comum entre eles e Bob. Neste caso, Cindy estaria sempre incapaz de dizer quem assinou.

Esta propriedade será utilizada no esquema proposto neste trabalho, pois essa dinâmica será utilizada para proporcionar que uma Alice, que deseja se identificar perante um Bob receptor, possa usar um esquema *SDVS* para rotular uma fração de mensagens com assinaturas válidas e reconhecíveis por Bob e criar também rótulos com assinaturas não válidas (que Bob verá que não validam para as mensagens correspondentes) e indistinguíveis das assinaturas legítimas. A visão que qualquer adversário tem dessa mistura de assinaturas não o possibilitará fazer qualquer distinção, a menos que ele consiga quebrar a segurança do esquema de assinatura digital subjacente. Então, o rótulo com assinatura válida será usado apropriadamente para identificar um fluxo na rede.

5 Descrição do esquema proposto

Neste capítulo, serão integrados os conceitos apresentados até aqui, com o objetivo de descrever a proposta deste trabalho. Será demonstrado como o ferramental criptográfico apresentado será reunido para se construir um método de identificação de tráfego apropriado para uma rede baseada em interconexão de pacotes, como a infraestrutura de roteamento *IP*.

5.1 Marcando pacotes com rótulos

A ideia de se marcar pacotes com o objetivo de proporcionar um controle baseado em habilitação (método já apresentado na Seção 2.4.1) não é recente. Quando todos os aspectos de segurança não são levados em consideração, uma aplicação ingênua de ferramentas de criptografia seria a de se anexar uma assinatura digital a cada pacote transmitido, criando assim um esquema baseado em assinaturas digitais que identificasse um fluxo na rede.

É importante destacar que os algoritmos convencionais de assinatura não seriam o suficiente para se permitir um esquema de identificação seguro, isso porque o adversário poderia meramente repetir o tráfego em algum outro momento capturado. Outro aspecto é o da complexidade de se computar assinaturas de cada pacote, que é alta quando se considera assinar todo o fluxo, e por sua vez o receptor ter que verificá-lo completamente, o que do mesmo modo demandaria tempo computacional.

A proposta deste trabalho apresenta um mecanismo que faz uso de assinaturas digitais de modo mais eficiente. Ademais, é considerado aqui que um método de defesa contra *DoS* que se baseia em criptografia deve levar em conta os ataques formalizados no modelo de adversário proposto na Seção 2.4, inclusive no caso particular em que os conhecidos métodos falham em não resistir a um ataque de reenvio de fluxo de dados.

A abordagem adotada neste trabalho é a de explorar o poder de equivocar um adversário, por proporcionar que um transmissor possa assinar alguns pacotes de uma maneira válida e reconhecível pelo receptor, destinatário das assinaturas designáveis, mas também inserir assinaturas não válidas e indistinguíveis de qualquer outra gerada pelo transmissor. Esta dinâmica tem o objetivo de frustrar os planos de um adversário que intenta criar tráfego malicioso indiscriminadamente, inclusive na situação em que ele possa capturar tráfego e reenviá-lo. Mais especificamente, o transmissor de tráfego fará uso de um esquema *SDVS* no intuito de executar as tais assinaturas. Ao mesmo tempo em que ele faça uso de chaves válidas, fará também o de chaves falsas (sintaticamente válidas,

escolhidas no domínio correspondente, mas escolhidas semanticamente falsas, aleatoriamente), em que as assinaturas geradas por ela têm efetivamente a intenção de amplificar a dúvida no adversário.

O modelo proposto neste trabalho foi inspirado em duas importantes ideias do cenário da criptografia moderna: o esquema de *privacidade diferencial* de Dwork [127] e no modelo de *k-anonimidade* [128] introduzido por Samarati e Sweeney. O conceito trazido por privacidade diferencial é o de se inserir um erro controlado em um conjunto de dados, de maneira que se permita reportar informação com privacidade: o controle do erro é feito em privacidade e quem o conhece sabe como desconsiderá-lo. A intuição no modelo de k-anonimidade é o de desvincular o relacionamento entre dados armazenados em um banco de dados estruturado. O modelo permite recuperação de dados de modo que não haverá certeza das relações. A intuição é a de que quando uma informação é repassada, haverá incerteza sobre quem ela pertence, ou a que outros dados ela está relacionada, pois ela pode corresponder a no mínimo k outros indivíduos.

Uma visão geral do esquema é apresentada na Figura 11 em que é mostrado um fluxo de rede baseado em pacotes. A estratégia é a de se escolher aleatoriamente alguns deles, nos quais assinaturas digitais válidas geradas com uso de chaves válidas serão afixadas. Por outro lado, há pacotes que não receberão assinaturas e outros que receberão assinaturas semanticamente falsas com o objetivo de equivocar o adversário.

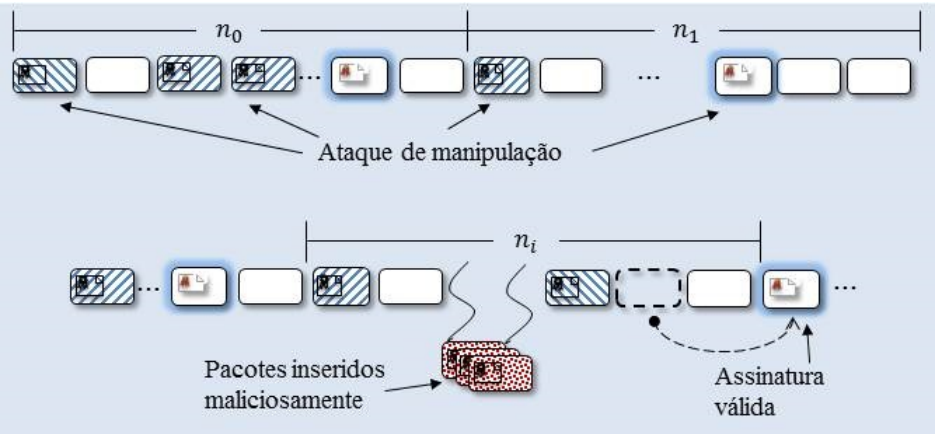


Figura 11 – Visão geral do esquema proposto: inserção de assinaturas válidas intercaladas com a de assinaturas falsas.

Ainda considerando a Figura 11, a visão que o adversário tem desses pacotes assinados não lhes dá nenhuma vantagem para distinguir válidos entre não válidos. Entretanto, o receptor designado das assinaturas poderá fazê-lo com eficiência, e interpretar as mensagens coerentemente assinadas como uma *habilitação* enviada pelo transmissor, vez que apenas ele poderia tê-las criado. O receptor registrará aquela fonte e poderá criar uma política baseada em reputação de quem gerou tráfego legítimo. Isso será possível porque

endereço do remetente e chave pública do assinante têm que funcionar coerentemente para criar assinaturas válidas, o que o receptor poderá usar para identificar o transmissor.

5.2 Detalhes e parâmetros de segurança no modelo proposto

Nesta seção, será apresentado o protocolo proposto em detalhes, descrevendo-se os parâmetros de segurança e como eles compõem a solução completa.

O esquema proposto usa os seguintes parâmetros de segurança:

- n: o tamanho da sequência de pacotes a ser considerada em cada iteração de assinaturas inseridas;
- Σ: o número de pacotes nos quais alguma assinatura será inserida em cada iteração de assinaturas;
- σ: o número de assinaturas válidas entre o conjunto total Σ de assinaturas inseridas em cada sequência de n pacotes; e
- τ: o limiar de segmentos de tamanho n sem assinaturas válidas, ou seja, o receptor irá esperar no máximo $\tau \cdot n$ pacotes sem que possa identificar ao menos uma assinatura válida; de modo contrário, o status de autenticação para com este transmissor será perdido.

O protocolo proposto funciona como a seguir. O transmissor, a fim de convencer que ele é uma fonte identificável de um fluxo de pacotes na rede irá trocar informação necessária para receber a chave pública do destinatário. Nenhuma suposição sobre como será publicada esta chave pública, e não será considerado que as partes utilizarão uma infraestrutura de chaves públicas para isso. Portanto será apenas considerado que esta chave é publicamente acessível.

Depois, para cada sequência de n pacotes a ser enviado, o transmissor realizará uma iteração de assinaturas, isto é: aleatoriamente escolher Σ pacotes nos quais serão inseridas assinaturas, dentre estas σ serão válidas. Por sua vez, o receptor poderá verificar e contabilizar as válidas, e portanto vai poder administrar se o transmissor tem enviado assinaturas válidas ou se o limiar foi ultrapassado o tornado não mais habilitado.

Mais formalmente, o protocolo proposto consistirá nos seguintes passos:

- **Inicialização**: Esta fase começa por todas as partes abertamente concordarem sobre um parâmetro de segurança 1^N, usando-o no algoritmo de geração de chaves KGen para definir suas chaves criptográficas.

Algoritmo 1: Inicialização

begin

Alice recebe $(\mathsf{pk}^A, \mathsf{sk}^A) \xleftarrow{R} \mathsf{KGen}\left(1^N\right)$

Bob recebe $(\mathsf{pk}^B, \mathsf{sk}^B) \xleftarrow{R} \mathsf{KGen}\left(1^N\right)$

end

- **Assinatura**: Depois da fase inicial, a assinante Alice, para cada iteração em que n pacotes $p_1, ..., p_n$ serão transmitidos, terá que escolher aleatoriamente um subconjunto Σ dentre eles, ou seja, $\Sigma \subset \{p_1, ..., p_n\}$ em que alguma assinatura será inserida, seja ela válida ou falsa. Ou seja, uma vez que escolheu Σ, ela irá subdividi-lo aleatoriamente em um subconjunto σ cujos pacotes receberão assinaturas válidas. Naturalmente, os elementos no subconjunto restante $\Sigma \backslash \sigma$ irão receber assinaturas falsas. Essa dinâmica é descrita no Algoritmo 2 a seguir:

Algoritmo 2: Alice (Assinante).

begin

for $\{p_1, ..., p_n\}$ **do**

Gera chaves falsas: $(\mathsf{sk}_f^A, \mathsf{pk}_f^A) \xleftarrow{R} \mathsf{KGen}\left(1^N\right)$

Escolhe aleatoriamente: $\Sigma = \{p_1, \ldots, p_{|\Sigma|}\}$

Escolhe aleatoriamente: $\sigma \subset \Sigma$

for $p_i \subset \sigma$ **do**

$\sigma_i \leftarrow \mathsf{Sign}(p_i, \mathsf{sk}^A, \mathsf{pk}^B)$

Anexa$(p_i|\sigma_i)$

end

for $p_i \subset \Sigma \backslash \sigma$ **do**

$\sigma_j \leftarrow \mathsf{Sign}(p_j, \mathsf{sk}_f^A, \mathsf{pk}_f^B)$

Anexa$(p_i|\sigma_i)$

end

end

end

- **Verificação das assinaturas**: Para cada pacote recebido com assinatura, Bob deverá verificar se assinatura anexa é válida a fim de manter atualizada a contabilidade do tráfego enviado e de se o transmissor ultrapassou o limite de $\tau \cdot n$ pacotes transmitidos sem uma assinatura válida. Portanto, se Bob, como uma fonte de tráfego de rede, não conseguir inserir pacotes dentro deste segmento de tolerância, Alice considerará que o tráfego eventualmente recebido não é mais legítimo, ou alguém está tentando trapacear o esquema. Todo este processo é formalizado no Algoritmo 3 a seguir:

Algoritmo 3: Bob (Verificador).

begin
  for $\forall\ p = (p_i|\sigma_i)$ **do**
    if $\mathsf{Vrfy}(p_i, \sigma) \stackrel{?}{=} 1$ **then**
      Atualizar o contador (τ)
    end
    else
      Atualizar o contador(τ)
    end
  end
end

- **Simulador**: Similar à argumentação de segurança do esquema *SDVS* em [126, 129], aqui o simulador exerce uma importante função na suposição de segurança do protocolo. Na argumentação da segurança do esquema *SDVS*, a existência de um simulador que permite imitar toda transcrição da comunicação trocada numa autêntica execução do protocolo é usada como argumento da inviabilidade, para todos exceto o verificador designado, de se distinguir entre uma assinatura válida e um bloco binário, do mesmo tamanho, aleatoriamente escolhido. O simulador irá internamente escolher alguma aleatoriedade e tomar como entrada as chaves $(\mathsf{sk}^A, \mathsf{pk}^B)$ e os pacotes a serem assinados. Então, o simulador Simul da conjectura de segurança do protocolo *SDVS* (porque o esquema é suposto ser seguro, e portanto existirá um simulador utilizado na sua argumentação de segurança) será usado quando a simulação se referir a assinaturas escolhidas dos pacotes no subconjunto σ, já as outras assinaturas, $\Sigma\backslash\sigma$ serão assinadas com as chaves falsas pelo algoritmo Sign.

Algoritmo 4: Simulador.

Input: n pacotes: $\Sigma = \{p_1, ..., p_n\}$; $\mathsf{sk}_f^A, \mathsf{pk}_f^B \leftarrow \mathcal{G}(1^N)$
begin
  Escolhe $\sigma \subset \Sigma$
  for $p_i \in\ \Sigma\backslash\sigma$ **do**
    $\sigma_i\ \leftarrow \mathsf{Sign}(p_i, \mathsf{sk}_f^A, pk_f^B)$
  end
  for $p_i \in\ \sigma$ **do**
    $\sigma_i\ \leftarrow \mathsf{Simul}(p_i^{'}, \mathsf{sk}_F^A, \mathsf{pk}_F^B)$
  end
end

O argumento aqui será o de que se o adversário possui uma estratégia para eficientemente distinguir entre o conjunto de pacotes assinados pelo simulador descrito acima

e um conjunto autenticamente assinado por Alice, tal estratégia poderia ser utilizada para se construir um esquema que a use para quebrar a mesma propriedade no algoritmo *SDVS*, contradizendo a suposição de que esta propriedade é verdadeira naquele algoritmo, isto é, o de ser inviável para qualquer pessoa, exceto o verificador designado, distinguir entre uma assinatura válida e uma assinatura gerada aleatoriamente. Em outras palavras, a habilidade do adversário de fazer esta distinção na saída do simulador é reduzida a capacidade de quebrar o esquema *SDVS*.

Consequentemente, se existe um simulador como o tal descrito acima, ele pode ser usado para se criar assinaturas indistinguíveis: sintetizando um conjunto aleatório de pacotes que têm assinaturas indistinguíveis das assinadas por Alice legitimamente. Por sua vez, se uma estratégia do adversário existe que lhe dê vantagem em distinguir entre esses dois mundos, consequentemente esta hipotética estratégia poderá ser usada para quebrar a assinatura designável.

Aqui será usada uma argumentação nos moldes da demonstração por argumento híbrido no Capítulo 3 de [10]. O que deve ser considerado é que existe dois tipos de assinaturas σ^a geradas por Alice: as válidas e as criadas aleatoriamente, as quais aqui serão designadas por σ_v^a e σ_r^a, respectivamente. Ambas são indistinguíveis das sintetizadas pelo simulador σ^{Sim}, denominadas aqui σ_v^{Sim} e σ_r^{Sim}. No entanto, o que é necessário para se fazer uma distinção entre estes dois mundos é uma estratégia que permita se reconhecer ao menos um par de elementos um em cada lado, ou seja, que dê alguma vantagem em reconhecer assinaturas vindas de Alice em oposição às advindas do simulador. Entretanto, pelas propriedades do esquema *SDVS*, os pares σ_v^a, σ^{Sim} e σ_r^a, σ^{Sim} são indistinguíveis, e está é a dificuldade que o adversário deve transpor.

Deste modo, se houver uma estratégia que dê vantagem para que se faça a distinção entre as duas sequências apontadas acima, essa estratégia poderia ser usada como a seguir. Alguém que quisesse quebrar as propriedades do esquema *SDVS* por distinguir entre duas assinaturas uma gerada legitimamente e outra gerada aleatoriamente, sintetizaria um dos conjuntos de assinaturas e incluiria as assinaturas nesses conjuntos e submeteria a estratégia, que deveria as distinguir, o que contrariaria a hipótese do esquema *SDVS* ser seguro.

5.3 Aplicação em um cenário de rede

Aqui será apresentado um cenário no qual o esquema proposto poderá ser utilizado como uma ferramenta de defesa contra ataques *DoS*. Neste cenário, é sugerido que o esquema proposto funcione como um recurso de identificação da fonte geradora de tráfego, o que será útil para um destinatário gerenciar a permissão ou bloqueio de adversários.

O cenário aqui apresentado considera ataques *DoS* gerados em redes de roteamento

de pacotes como a do *IP*. A estratégia será de aplicar o esquema na eventualidade de ataque de DoS, ou seja, uma vez que a rede receptora esteja consciente de um ataque *DoS* (porque algum mecanismo de detecção *DoS* a alarmou), então será ativado o esquema proposto a fim de se estabelecer um filtro, permitindo a rede selecionar quem tem boa reputação e quem está agindo como adversário. Portanto, os usuários legítimos poderão continuar transmitindo, porém os de baixa reputação serão efetivamente bloqueados. Para isso, diante de uma ameaça, o receptor irá exigir que todo tráfego que ele receba deva ser assinado conforme o esquema proposto.

No cenário ilustrado na Figura 12, é levado em consideração um contexto de rede *IP* em que há um provedor de serviço alvo de ataques *DoS*, na figura representado por *Servidor alvo*. As *Redes A*, *B*, e *C* hospedam clientes deste serviço, elas estão conectadas por sistemas autônomos (*AS*) dispersos na Internet, e naquelas redes é onde estarão funcionando usuários maliciosos e legítimos.

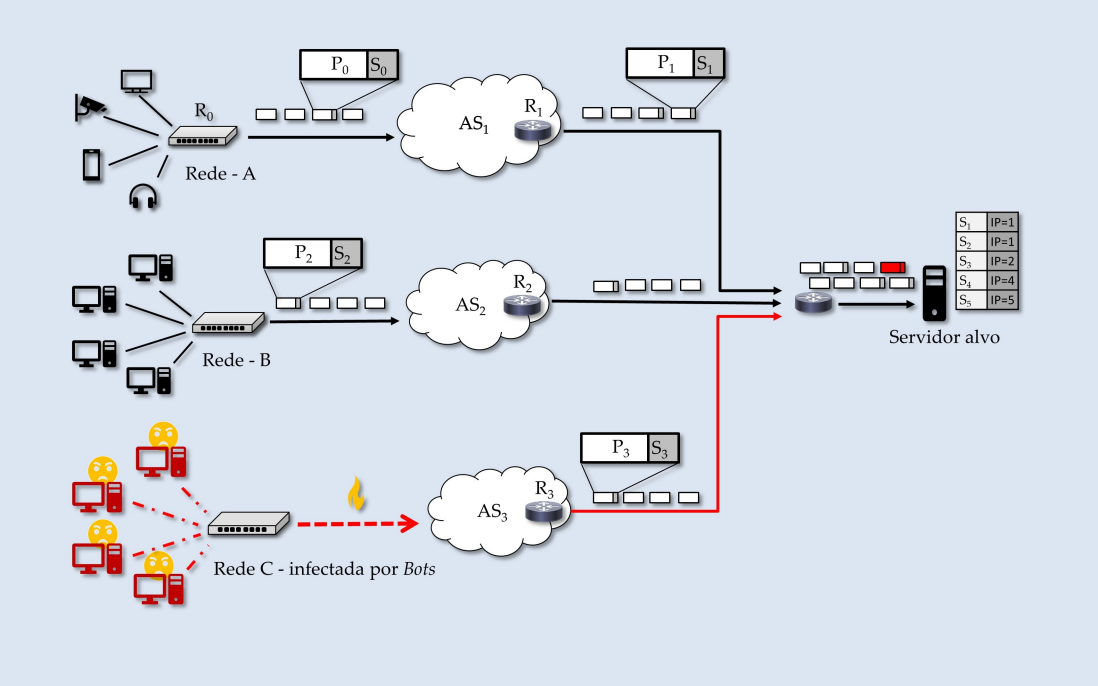


Figura 12 – Cenário de aplicação do esquema em uma rede enfrentando ataques de *DoS*.

No cenário descrito na Figura 12, a ideia é que o mecanismo quando ativado deva bloquear o tráfego malicioso vindo da *Rede C*, infectada. Neste caso, o destinatário poderá identificar aquele tráfego, porque neste específico exemplo ele estará sendo assinado por um roteador de borda do sistema autônomo AS_3.

Uma grande vantagem do esquema proposto é a de que ele pode ser implementado de modo hierárquico, ou seja, o processo de assinatura não necessariamente tem que ser realizado pela real fonte de tráfego num estilo fim a fim. O esquema pode ser realizado de mode descentralizado, quando roteadores, pontos de acesso sem fio, e outros ativos de

interconexão poderão fazê-lo, inclusive de modo simultâneo. No cenário exemplificado, poderia existir a realização concomitante do esquema pelo roteador de borda R_3 no sistema autônomo AS_3 e também dos computadores ou ainda do equipamento de interconexão (*switch*) ao qual eles estão diretamente conectados. Na Figura 13, é dada uma visão de

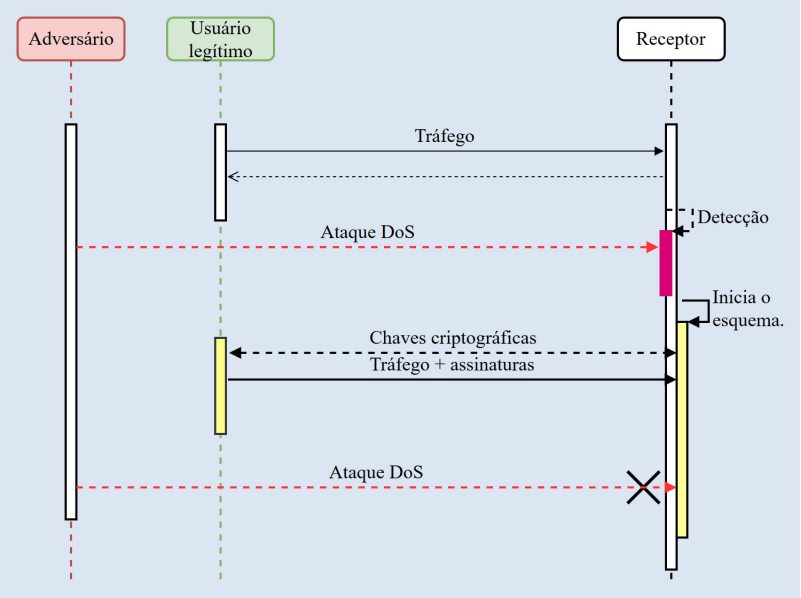


Figura 13 – Diagrama de sequência do protocolo proposto.

engenharia de software (diagrama de sequência) do protocolo. Em princípio, o receptor não está recebendo tráfego de ataque e aceita fluxo da rede sem a necessidade de que o tráfego seja assinado pelo esquema. Na eventualidade de um ataque, o receptor ao perceber uma ameaça detectada, bloqueia todo o tráfego não identificado, exigindo assim que quem deseja transmitir tenha que cooperar via protocolo. Quando então a origem legítima de tráfego começa a usar o protocolo, enviando pacotes convenientemente assinados, o receptor inicia o processo de verificação das assinaturas, o que perdurará enquanto o protocolo estiver ativo porque o receptor ainda se sente ameaçado pelos ataques. Neste ponto, o receptor reconhecerá aquele adversário que não conseguirá apensar assinaturas convenientemnete.

Na Figura 14, é ilustrado um cenário questionado pelos revisores de um dos artigos publicados pelo autor desta tese e relacionado a este trabalho [130]. No contexto discutido, o receptor administrará o caso em que uma origem que já é conhecida como de boa reputação começa a enviar tráfego malicioso: por exemplo, se ela eventualmente foi tomada por um adversário. Este é um cenário previsto no modelo de adversário, ou seja, que o adversário é dinâmico no sentido a poder escolher a qualquer momento que parte ele vai tomar total controle. De qualquer forma, é uma ameaça muito concreta pois nenhum esquema pode se fundamentar na suposição de que o sistema de um participante é infalível.

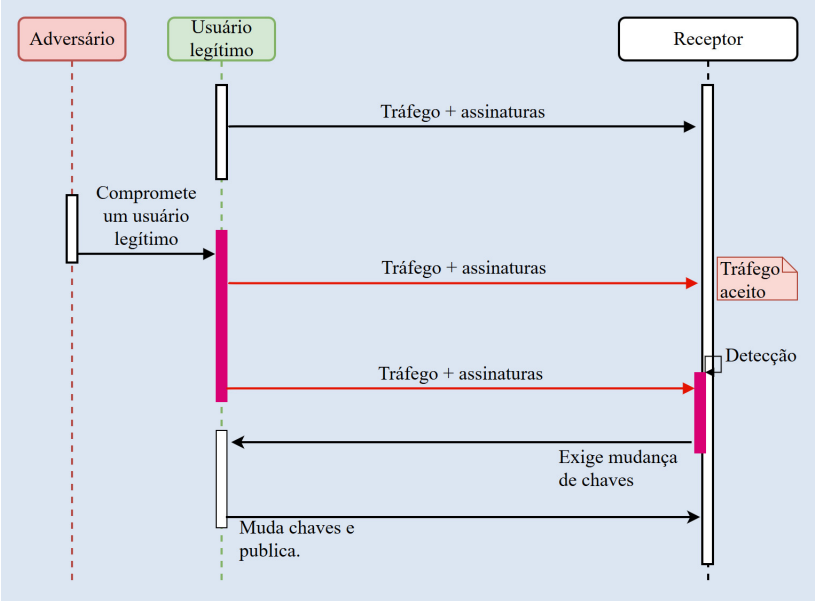


Figura 14 – Diagrama de sequência: redefinição de chaves.

Por exemplo, a quebra de senhas foi a principal causa de recentes grandes ataques com *IoT*, como foi o caso dos ataques por códigos maliciosos relacionados ao vírus *Mirai* [131].

Observe-se aqui que o objetivo do esquema proposto é o de se estabelecer um mecanismo em que falhas de autenticação ativarão um filtro que, durante um ataque, permitirá que apenas tráfego identificado como bem comportado seja recebido. Por outro lado, se um já bem avaliado transmissor poderá ser invadido e ser usado por adversários que planejam um ataque, o esquema possibilitará uma boa estratégia porque o receptor poderá exigir que um transmissor mude as chaves dele. Esta é uma boa defesa, pois inviabilizará ataques em que o adversário tomou um grande número de dispositivos: porque seria inviável estabelecer uma mudança em massa das chaves de milhões de dispositivos. A questão é que pode não ser computacionalmente difícil gerar chaves criptográficas (para muitos sistemas criptográficos não o é), mas a configuração coordenada de um grande número de chaves que devem ser diferentes e que devem ser mudadas ao comando do receptor é uma tarefa inviável para o adversário.

Outro ponto é que quando o mecanismo é exigido pelo receptor, qualquer que seja o transmissor na rede que queira ter seu tráfego aceito deve ser identificado. Também, qualquer adversário que tente trapacear com chaves criptográficas ao seu gosto, escolhidas estrategicamente, ele terá seu tráfego reconhecido, porém entendido como malicioso. Ainda que um adversário tente mudar em intervalos escolhidos por ele estas chaves de assinatura, o receptor poderá optar por receber apenas tráfego de boa reputação e descartar qualquer outro. Portanto o esquema poderá ser usado num estilo lista branca (*white-list*) tão comumente aplicadas em segurança de redes.

6 Análise da resiliência do protocolo

Neste capitulo será apresentada uma argumentação da segurança e da resiliência do esquema proposto, demonstrando-as por especificação formal de sistemas, realizada pela modelagem por diagramas de estados, com base na qual serão verificadas aquelas propriedades.

É necessário se formalizar uma medida de quão resiliente o esquema proposto se mostra frente a ataques de negação de serviço. Serão demonstradas quais as chances que um adversário pode ganhar por atacar uma rede em que transmissores e receptores de tráfego concordaram em usar o esquema proposto. Com este objetivo em vista, serão analisados os cenários de ataques já apontados anteriormente no texto.

6.1 Considerando ataques por *spoofing*

Qualquer tentativa de capturar tráfego, seja por copiá-lo e usá-lo para ganhar vantagem numa tentativa de retransmiti-lo, seja por forjá-lo para que pareça ter sido originado por uma outra parte (considerada pelo receptor como de boa reputação), terá sucesso se o adversário puder sintetizar assinaturas que possam ser validadas no nome da origem vitimada. Entretanto, será demonstrado aqui que qualquer estratégia que o adversário faça uso não permitirá que ele possa manter um envio contínuo de tráfego malicioso que seja reconhecido pelo destino como se legítimo fosse. Será demonstrado que as chances que ele terá para realizar esses ataques com sucesso são pequenas (uma formalização do que isso realmente significa será apresentada), a menos que ele saiba como violar o esquema de assinaturas *SDVS* subjacente.

Como já apontado anteriormente, ninguém exceto o verificador designado irá reconhecer uma assinatura válida gerada por *SDVS*. Deste modo, restará ao adversário lançar mão à estratégia de copiar todos os pacotes assinados e inseri-los em uma nova sequência forjada para um ataque por *spoofing*. Porém, esta estratégia não é gratuita, pois a inserção dos pacotes assinados tem que corresponder às assinaturas e, portanto, não podem ser manipulados livremente. De fato, a manipulação não é possível para um adversário em tempo polinomial (como discutido na Seção 4.2). Então, seria necessário um grande poder computacional para adaptar aqueles pacotes assinados a um novo tráfego, de modo a ainda permanecerem coerentes às respectivas assinaturas. Essa estratégia pode ser pensada como uma prova de trabalho às avessas: o usuário legítimo não precisa de esforço computacional para ter seu tráfego aceito, mas o adversário tem que despender tempo computacional para forjar tráfego e tê-lo aceito como autêntico pelo receptor.

Outro ponto a considerar é o do parâmetro τ, que é o limiar de quantos segmentos de n pacotes serão aceitos sem alguma sequer assinatura válida. Com efeito, se o atacante obteve uma boa dica de qual dos pacotes é o com assinatura válida, ou seja, se ele possui uma boa estratégia de localizar os σ dentre o total de Σ assinados, a vantagem que ele adquire é relativa. Ainda que ele tenha uma estratégia como essa, a simples cópia daqueles pacotes para um eventual ataque por *spoofing* irá deixá-lo transmitir não mais que $2\tau n$ pacotes, até que o receptor perceba que não mais sinalizações de identificação foram recebidas o suficiente para decidir que o tráfego daquela origem não é mais legítimo. Contudo, por se considerar que este ataque hipotético não é factível de se realizar, a estratégia que restará para o adversário será a de tentar inserir maliciosamente uma sequência de τn pacotes. Ou seja, para se tomar um exemplo em que é definido um pequeno $\tau = 1$, as possibilidades do adversário estarão reduzidas a inserção de n pacotes.

Outro cenário, discutido pelos revisores de um artigo do autor desta tese publicado em [130] e correlato a este trabalho, é o em que se considera um usuário malicioso que usurpa a identidade de um transmissor e assina pacotes em seu nome com uma chave secreta não válida e diferente da legítima, ou seja, o receptor considerará a chave pública (legítima) do transmissor que teve sua identidade usurpada. Em primeiro plano, a preocupação é o de esquema estar proporcionando um meio de sobrecarregar o receptor, despachando pacotes com assinaturas nunca válidas, desencadeando um ataque no estilo poluição (*pollution attacks*) [132] contra o mecanismo. Entretanto, é preciso lembrar que o receptor espera uma taxa conhecida de pacotes assinados e, portanto, o ataque por poluição não poderá exceder inconsequentemente o número de pacotes assinados a cada intervalo de n totais enviados. De qualquer modo, o receptor irá contar assinaturas válidas nos pacotes que receber, e esta contabilidade poderá o alarmar caso um transmissor malicioso esteja tentando indefinidamente o repassar pacotes apenas para onerá-lo em processamento.

6.2 Considerando ataques por inundação

O adversário poderá adicionar maliciosamente pacotes no fluxo regular da rede, a fim de provocar ataques por inundação (*flooding*). Por se realizar esse tipo de investida, o adversário estará modificando a distribuição dos pacotes assinados e eventualmente poderá deixar um segmento não autenticado. Então, para realizar esse tipo de inserção ele teria que conhecer a posição das assinaturas válidas, pois a estratégia mais eficiente neste caso seria inserir tráfego entre assinaturas válidas, e assim explorar ao máximo a autenticação gerada pelo usuário legítimo. Considerando que com uma taxa de σ/Σ para cada segmento de n pacotes, a distância esperada entre duas assinaturas válidas será n/σ, ou seja, a melhor estratégia para o adversário seria portando inserir tráfego à meia distância entre assinaturas válidas, como foi argumentado. Se o adversário então não

pode distinguir e, portanto, não pode localizar assinaturas válidas, então ele não poderá ter estratégia melhor do que uma inserção em posições aleatórias.

6.3 Ataques de negação de habilitação

Como já discutido na Seção 2.4.1, habilitação de tráfego é um dos meios de um transmissor se identificar para com um receptor na rede. Um ataque de negação de habilitação (*Denial of Capability*) [79] é o que ocorre quando um adversário tenta frustrar um esquema baseado em habilitação, seja ela um *token*, um valor de *hash*, uma resposta a um desafio, ou algo assinado digitalmente. O esquema proposto neste trabalho é baseado em habilitação e, portanto, os ataques possíveis contra estes mecanismos serão analisados e confrontados com a solução aqui proposta.

O descarte indiscriminado de pacotes por um adversário que interfere no canal da rede é conhecido em redes sem fio e em redes *IP*, e essas ameaças têm suas próprias contra medidas [35], incluindo os mecanismos de controle de congestionamento e retransmissão de pacotes. Não obstante, um adversário poderá sempre manipular conteúdo nos pacotes, o que de qualquer modo é uma efetiva estratégia de ataque que frustra qualquer mecanismo baseado em habilitação.

Contudo, considerando o mecanismo de autenticação proposto, para que um adversário lance algum ataque para tentar negar um mecanismo de habilitação, ele terá novamente o seu sucesso na dependência de distinguir assinaturas válidas no esquema *SDVS*. Caso contrário, ele terá que descartar uma boa percentagem, uma análise que será dada no Capítulo 7.

Ademais, se o adversário tentar frustrar a autenticação, seja por descartar pacote ou manipular o seu conteúdo, a chance que ele terá de com sucesso interromper uma autenticação vai depender dos parâmetros usados no esquema. De qualquer forma, um mecanismo de retransmissão (como o de redes *IP*) exigirá do remetente que ele repita pacotes descartados. Alternativamente, se ele tentar modificar δ dos Σ pacotes, a chance alcançada por ele dependerá de quão hábil ele será em interferir nos pacotes válidos, vez que o remetente assinará pacotes e continuamente estará desafiando o adversário. Obviamente, se δ é grande o suficiente para frustrar todos os pacotes assinados, o adversário estará se expondo em adotar tal estratégia.

Para apresentar aqui um simples exemplo com parâmetros concretos, quando o esquema for configurado para assinar um total de $n = 100$ pacotes, dentre esses $\Sigma = 10$ com alguma assinatura, dos quais apenas $\sigma = 1$ contendo uma assinatura válida. O destinatário tolerará até $\tau = 5$ n-sequências sem uma assinatura válida, ou seja, até 500 pacotes. Um adversário que tenta manipular 50% do total do tráfego terá aproximadamente a chance $(5/10)^5 \approx 0,0312$ em probabilidade de frustrar a identificação da origem.

6.4 Aplicação de autômatos finitos na demonstração de segurança

No projeto de protocolo em que uma demonstração da segurança é necessária, a demanda por um modelo de argumentação de segurança pode ser atendida por diversos paradigmas. Um modelo conhecido paradigma de verificação de propriedades de protocolos de rede é o da modelagem por *máquinas de estados*, também conhecidos como *autômatos*. A especificação de sistemas por autômatos é aplicada em diversas propostas [133, 134, 135, 136].

6.4.1 Autômatos finitos

Uma definição intuitiva, informal do conceito de autômatos finitos *FSA* (*Finite States Automata*) pode ser tomada por considerá-los como uma abstração de um processo em que o todo pode ser modelado como um passo a passo. É uma poderosa ferramenta de modelagem de sistemas, tanto que o modelo é *Turing* completo, ou seja, tem o poder capturar tudo o que é computável e que possa ser programado e executado como um processo em um computador.

Mais formalmente, o modelo proporciona uma abstração das relações entre conjuntos, de maneira que elementos de um conjunto são pensados para representar estados de um sistema, tal que o relacionamento entre eles seja a transição entre esses estados. A transição condicionada a um evento que faz o estado local do sistema mover do estado S para o estado Q é denotada por $S \rightarrow Q$. Mais formalmente:

Definição 8. ***Autômato de estados finitos*** *[137] — um autômato de estados finitos é uma ênupla* $(Q, \Sigma, q_0, F, \delta)$ *em que* Q *é um conjunto não vazio de estados, entre os quais* q_0 *é chamado de estado inicial.* Σ *é o alfabeto, conjunto finito e não vazio,* $F \subseteq Q$ *é um subconjunto de estados denominados finais ou de aceitação. A função de transição é* $\delta : Q \times \Sigma \rightarrow Q$*, isto é, quando aplicada a um elemento do alfabeto retorna o próximo estado.*

Uma exemplificação simples de um autômato finito que modela um processo que realiza contagem de caracteres em uma palavra pode ser pensado como a seguir. O processo é programado para contar a ocorrência de caracteres 'c' numa palavra $\{a_1, a_2, \ldots, a_n\}$, contando o número de ocorrências até o limite de 5. O esquema está ilustrado na Figura

15; a Tabela 1 elenca as regras de transição correspondentes a este exemplo.

$$\begin{aligned} Estados &:= \{q_0, q_1, q_2, q_3, q_4\}, \\ Estado\ inicial &:= q_0, \\ Alfabeto &:= \{a, b, \ldots z, A, B, \ldots, Z\}, \\ Transições &:= \delta\left(q_i, `c'\right) \ \rightarrow \ q_{i+1|i<5}, \\ Estados\ finais &:= \{q_1, q_2, q_3, q_4, q_5\}. \end{aligned}$$

No exemplo acima, o autômato finito representa um processo em que cada um dos

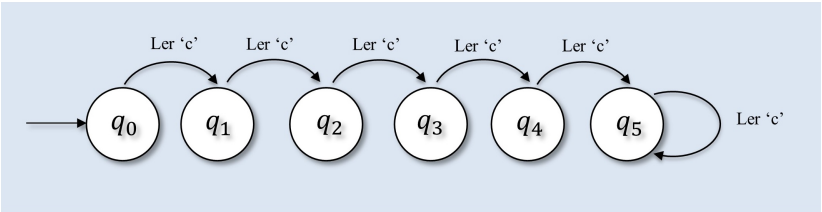


Figura 15 – Diagrama de estados que representa o processo de contagem de caracteres.

Tabela 1 – Relação entre a representação do estado anterior, o símbolo sendo lido e o próximo estado.

Estado anterior	**Símbolo**	**Novo estado**
q_0	'c'	q_1
q_1	'c'	q_2
q_2	'c'	q_3
q_3	'c'	q_4
q_4	'c'	q_5
q_5	'c'	q_5

caracteres da palavra é lido, o que terminará por se contar n caracteres, em que q_n é o estado em que o processo terminou, ou ao menos cinco caracteres 'c' foram contados.

Como uma ferramenta de argumentação da segurança de protocolos, autômatos finitos dão uma visão macro do comportamento das partes no sistema. Então, as propriedades desejadas do protocolo podem ser associadas a estados, e o processo de modelar inclui o de se elencar todos os estados relevantes para a argumentação de segurança, com os correspondentes eventos de transição. Um aspecto crítico para demonstração de segurança é apontar nesse processo quais são os estados inseguros e apresentar uma demonstração de quais eventos podem levar a esses estados, formalizando-se as chances de ocorrer esses eventos e a chance de se alcançar esses estados inseguros.

Uma abordagem típica adotada na análise de segurança com a modelagem por autômatos é a de primeiramente se demonstrar que o protocolo é correto e seguro quando não sob a ação dos adversários. Depois, ao se considerar quais partes o adversário tomou controle, de acordo com o modelo adversarial adotado, a segurança do protocolo é analisada e as probabilidades de se alcançar um estado inseguro é demonstrada sob as tais ameaças.

Mais especificamente, com o protocolo proposto, quer se demonstrar que partes transmissoras legítimas poderão sempre provocar eventos que as levem a ser identificadas para com a parte receptora do tráfego, dentro dos estados de segurança do protocolo. Porém, é preciso demonstrar que as partes controladas pelo adversário não possam levar o sistema a um estado inseguro: levar o receptor a um estado identificação de tráfego que não foi legitimamente originado por quem é informado tê-lo feito.

6.4.2 O invariante como princípio

A tarefa de modelagem da segurança de protocolos como autômatos finitos pode se apresentar árdua porque, a depender da natureza do protocolo, a formalização da relação entre todos os estados, regras de transição e eventos pode implicar em um modelo incomensurável e difícil de se verificar. Em um caso em que o modelo terminou em um grande número de estados, ferramentas automáticas de verificação se mostram convenientes. Porém, um artifício tomado da prova por indução pode encurtar trabalho, é o denominado *princípio do invariante* [134, 135, 136]. A ideia na argumentação da validade um *invariante* em relação a um autômato é semelhante à estrutura de prova por indução: mostra-se primeiro o invariante ser verdadeiro para um caso base, e depois mostra-se ele ser verdadeiro para o caso $n + 1$, partindo da hipótese que o caso n já vale. Em particular, com autômatos se argumenta o invariante ser verdadeiro para todos os estados iniciais, e depois conjectura-se ele se manter verdadeiro para todos os estados alcançáveis, supondo-se que se parte de estados considerados seguros por hipótese.

Definição 9. ***Invariante*** — *um invariante é um predicado P a respeito dos estados, tal que sempre que ele for verdadeiro em algum estado q_i, isto é, $P(q_i)$ é verdadeiro, existindo uma transição $q_i \rightarrow q_j$, o predicado também será verdadeiro no estado q_j: $P(q_j)$ é verdade.*

A abordagem seguida na argumentação da segurança de protocolos pelo uso de autômatos é a de se mostrar que todos os estados que o adversário pode alcançar não violam o invariante [136]. Se é suposto que o invariante do modelo é o predicado: *o adversário desconhece a chave privada da parte legítima*, então será possível demonstrar que o adversário nunca alcançará o estado de autenticado, ou seja, nunca levará o autômato a um estado que representa a identificação de um transmissor que nunca de fato tentou se fazê-lo, a menos que aquele predicado seja violado. Além disso, considerando que para

um dado estado o invariante se mantém verdadeiro, ainda que o adversário acumule informação e processe todo conhecimento que ele capturou, seja qual for a estratégia da qual ele dispõe, ele não tornará falso o invariante considerado. Mais formalmente, esta argumentação será definida como a seguir.

6.5 Análise do protocolo proposto pela modelagem por autômatos finitos

Nesta seção, será formalizado o modelo proposto como um autômato finito, seguindo-se a abordagem em [136] e a notação adota em [134]. Toda a dinâmica do protocolo apresentada no Capítulo 5 será modelada por autômatos, por se descrever a interação entre as partes, os estados alcançados por elas, os eventos e os comandos executados, ou seja, todo o contexto observado pelas partes.

6.5.1 O protocolo proposto como um autômato finito

Os estados relacionados à modelagem da parte transmissora Alice e do receptor Bob serão denotados por $q_{\mathcal{T}}, q_{\mathcal{R}}$ respectivamente. Todos os elementos de rede que podem ser relevantes para definição do *status* da rede serão resumidos em estados no autômato. Assim sendo, serão incluídos nos estados associados às partes as mensagens m_i nas interfaces de rede, sejam a ser enviadas ou a ser lidas pela i-ésima parte. Os valores internos mantidos por cada uma dessas partes serão denotados por v_i, representando tudo que a parte recebeu ou enviou ou que por qualquer razão ela considera informação útil. Desta forma, a representação do atual contexto que cada uma das i-ésimas partes mantém será denotada como na Equação 6.1 a seguir

$$\begin{cases} q_{\mathcal{T}} =< v^i_{\mathcal{T}}, m^i_{\mathcal{T}} >, \\ q_{\mathcal{R}} =< v^i_{\mathcal{R}}, m^i_{\mathcal{R}} > . \end{cases} \tag{6.1}$$

Ademais, ainda seguindo [134], o estado associado ao adversário será reduzido ao que ele visualiza como contexto corrente $q_{\mathcal{A}} = m^1_{\mathcal{A}}, m^2_{\mathcal{A}}, \ldots, m^M_{\mathcal{A}}$, ou seja, o conjunto de mensagens que ele capturou até o momento corrente.

Portanto, o estado global do protocolo executado entre o transmissor e receptor sob a influência de um adversário é denotado como a seguir

$$[q_{\mathcal{T}},\ q_{\mathcal{R}},\ q_{\mathcal{A}}]\,. \tag{6.2}$$

A notação na Equação 6.3 será a adotada para representar as regras de transição. A condição $C_k\,(s_i|q_i)$ depende do estado corrente $q_i =< v^i_k, m^i_k >$, de modo que a regra de transição considera o atual valor v_i e qual mensagem está na rede prestes a ser processada m^i_k. A transição representada na a Equação 6.3 é o caso mais genérico, ou seja, o do

adversário, pois este está na condição de estar acumulando o que aprendeu até o momento, ou seja, a visão que ele tem da execução do protocolo, as mensagens trocadas entre as partes, inclusive as das execuções anteriores, o que é representado por se condicionar a transição ao conhecimento $q_{\mathcal{A}}$, como

$$r_k^{(i|q_{\mathcal{A}})} = if\ C_k\left(q_i|q_{\mathcal{A}}\right)\ then\ s \rightarrow s'. \tag{6.3}$$

Entretanto, no caso específico das partes honestas, estas condições de transição dependem apenas do estado atual q_i e do evento que a conduz do estado s ao s', novo estado, denotando-se o caso das partes honestas como

$$r_k^i = if\ C_k(q_i)\ then\ s \rightarrow s'. \tag{6.4}$$

De acordo com a nomenclatura apresentada acima, será exemplificado o caso como a seguir. Um dado transmissor está no momento atual no estado $q_T^i =< v_T^i, m_T^i >$, isto é, mantém a variável interna v_T^i e está por processar a mensagem m_T^i. Assim que este transmissor processa esta mensagem, ele reagirá por mandar a mensagem m_R^i ao receptor, e moverá seu estado (o do transmissor) para o $q_T^{i+} =< v_T^{i+}, \varepsilon >$, isto é, ele agora aumentou o conhecimento armazenado em sua memória e está agora esperando pela próxima mensagem, o que é representado pelo símbolo de mensagem vazia ε na fila de processamento da rede, em regras de transição como em

$$\begin{aligned} r_T^i = \textbf{If}\ C_T(q_S^i =< v_T^i, m_T^i >)\ \textbf{Then}\ & \left[q_T^i =< v_T^i, m_T^i >, q_R^i =< v_R^i, \varepsilon >, \ldots\right] \\ & \longrightarrow \left[q_T^{i+} =< v_T^{i+}, \varepsilon >, q_R^{i+} =< v_R^i, m_R^i >, \ldots\right]. \end{aligned} \tag{6.5}$$

Entretanto, esta regra quando executada muda o estado global do protocolo. As regras de transição mostram que o receptor, como o destinatário das mensagens enviadas, terá seu estado local modificado de q_R^i para q_R^{i+}, mantendo a mensagem transmitida na fila de espera da interface de rede pronta para ser processada. Este novo estado para o receptor será então considerado na execução da próxima regra de transição do seu estado local. O diagrama do autômato finito correspondente a este exemplo é mostrado na Figura 16.

6.6 Diagrama de estados do protocolo

Uma vez que a sintaxe foi introduzida, a descrição do protocolo proposto modelado como um autômato finito é como o ilustrado na Figura 17, o qual corresponde as regras de transição elencadas a seguir.

6.6.1 As regras de transição

As regras de transição aqui descritas seguirão a sintaxe já introduzida e considerará as partes no protocolo: transmissor, receptor. Não se cogita estas regras para o adversário

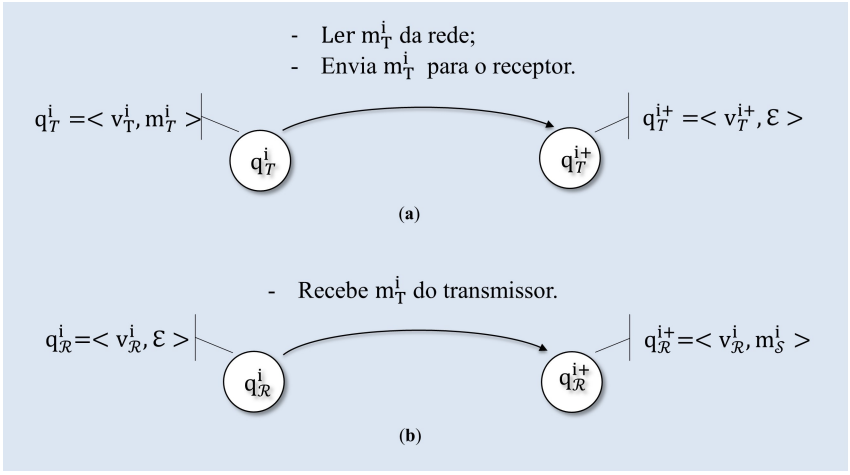


Figura 16 – Em (**a**), o transmissor envia uma mensagem; Em (**b**), o receptor responde para o transmissor depois de receber a mensagem enviada.

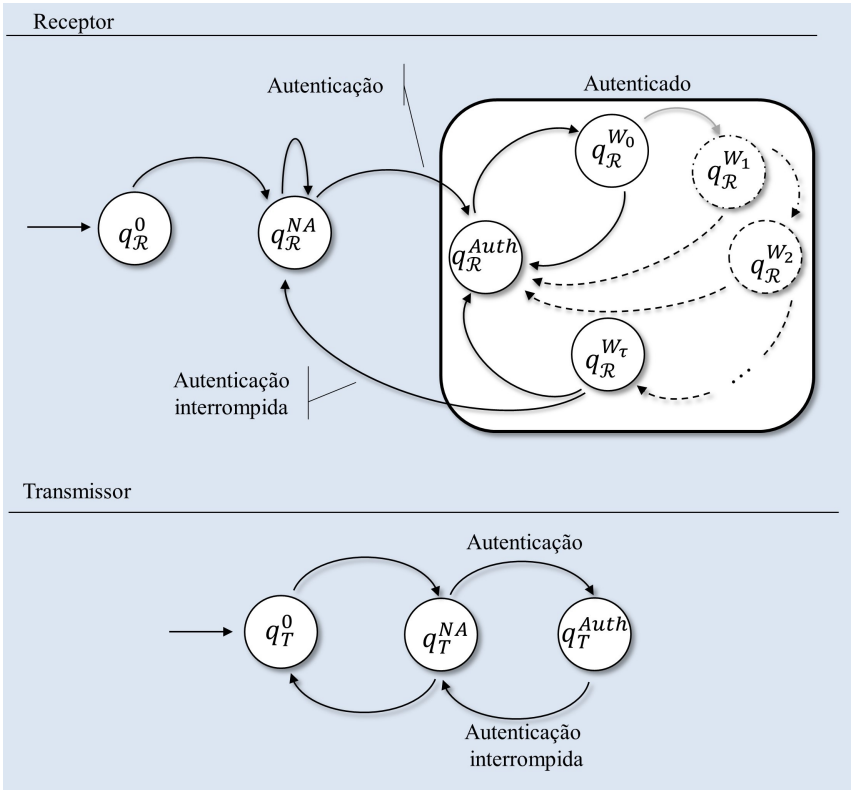


Figura 17 – O diagrama de estados do protocolo proposto. Em (**a**), os estados para o receptor. Em (**b**), para o transmissor.

porque ele funciona por corromper alguma das partes, e, portanto, quando atacando está fazendo as vezes ou do transmissor ou do receptor.

Estados do receptor	
$q_{\mathcal{R}}^{0}$	O estado inicial do protocolo, nenhuma transmissão realizada ainda;
$q_{\mathcal{R}}^{NA}$	Representa o estado de não autenticação;
$q_{\mathcal{R}}^{Auth}$	O receptor reconhece o transmissor e consegue identificá-lo;
$q_{\mathcal{R}}^{W_i}, i \geq 0$	O receptor está esperando por um novo sinal de autenticação, devido a ter-se alcançado i sequências de n-pacotes sem autenticação;
$q_{\mathcal{R}}^{W_\tau}$	O limiar de tolerância foi alcançado sem que uma assinatura válida repactuasse a autenticação.

Estados do transmissor	
$q_{\mathcal{T}}^{0}$	O estado inicial do protocolo, nenhuma transmissão foi realizada ainda;
q_{T}^{NA}	Representa o estado de não autenticação;
$q_{\mathcal{T}}^{Auth}$	Representa o transmissor autenticado.

Estados do adversário	
$q_{\mathcal{A}}^{0}$	Estado inicial do protocolo;
$q_{\mathcal{A}}^{NA}$	Representa o estado de não autenticação;
$q_{\mathcal{A}}^{Auth}$	Representa os estados nos qual o adversário é autenticado.

6.6.2 Os invariantes do protocolo

A demonstração da segurança do protocolo proposto será concretizada aqui por se definir seus invariantes e apresentar como eles se relacionam com as propriedades as quais se deseja argumentar serem válidas.

***Invariante 1**: o adversário não conhece a chave secreta do assinante honesto* — Este invariante fornece garantias de que qualquer parte que inicia o protocolo sendo incapaz de criar uma assinatura válida de uma mensagem designada a um receptor honesto, permanecerá incapaz de fazê-lo, a despeito do que ela aprende do contexto que observou. Isso também demonstra que se uma parte desonesta deseja se fazer autenticada, ela não terá sucesso, vez que se ela o fizer ela terá violado a invariante, ou seja, terá aprendido a chave secreta do assinante. O adversário, a despeito de toda a informação que captou, seria incapaz de sintetizar assinaturas válidas $\mathsf{Synth}(\cdot)$, caso contrário, ele teria quebrado a chave secreta.

Para se demonstrar que o invariante se mantém durante todos os estados alcançáveis pelo adversário, é preciso considerar os estados inseguros como sendo todos os estados de autenticação, ou seja, $Q = \{q_{\mathcal{R}}^{Auth}, q_{\mathcal{R}}^{W_0} \ldots, q_{\mathcal{R}}^{W_\tau}\}$. É suficiente se demonstrar que para alcançar qualquer desses estados um transmissor malicioso deve fazer um receptor sair do estado $q_{\mathcal{R}}^{NA}$ para o $q_{\mathcal{R}}^{Auth}$. No entanto, ao se observar as regras de transição do receptor,

Regras do receptor

$$
\begin{array}{ll}
r^{0}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{0}_{\mathcal{R}} =< v^{i}_{s}, m^{0}_{S} >) \textbf{ Then } [\ldots, q^{0}_{\mathcal{R}}, \ldots] \longrightarrow \left[\ldots, q^{NA}_{\mathcal{R}}, \ldots\right] \\
r^{NA}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{NA}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, m^{i}_{\mathcal{S}} >; \mathsf{Vrfy}(m^{i}_{\mathcal{R}})) \textbf{ Then } \left[\ldots, q^{NA}_{\mathcal{R}}, \ldots\right] \longrightarrow \\
 & \left[\ldots, q^{Auth}_{\mathcal{R}}, \ldots\right] \\
r^{Auth}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{Auth}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, m^{i}_{\mathcal{R}} >; \neg\mathsf{Vrfy}(m^{i}_{\mathcal{R}})) \textbf{ Then } \\
 & \left[\ldots, q^{Auth}_{\mathcal{R}}, \ldots\right] \longrightarrow \left[\ldots, q^{W_0}_{\mathcal{R}}, \ldots\right] \\
r^{W_j}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{W_j}_{\mathcal{R}} =< v^{j}_{\mathcal{R}}, m^{j}_{\mathcal{R}} >; \mathsf{Vrfy}(m^{j}_{\mathcal{R}})) \textbf{ Then } \left[\ldots, q^{W_j}_{\mathcal{R}}, \ldots\right] \longrightarrow \\
 & \left[\ldots, q^{Auth}_{\mathcal{R}}, \ldots\right] \\
r^{W_j}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{W_j}_{\mathcal{R}} =< v^{j}_{\mathcal{R}}, m^{j}_{\mathcal{R}} >; \neg\mathsf{Vrfy}(m^{j}_{\mathcal{R}}); j < \tau); \textbf{ Then } \\
 & \left[\ldots, q^{W_j}_{\mathcal{R}}, \ldots\right] \longrightarrow \left[\ldots, q^{W_{j+1}}_{\mathcal{R}}, \ldots\right] \\
r^{W_\tau}_{\mathcal{R}} & = \textbf{If } C_{\mathcal{R}}(q^{W_\tau}_{\mathcal{R}} =< v^{\tau}_{\mathcal{R}}, m^{\tau}_{\mathcal{R}} >; \neg\mathsf{Vrfy}(m^{\tau}_{\mathcal{R}})) \textbf{ Then } \left[\ldots, q^{W_\tau}_{\mathcal{R}}, \ldots\right] \longrightarrow \\
 & \left[\ldots, q^{NA}_{\mathcal{R}}, \ldots\right]
\end{array}
$$

Regras do transmissor

$$
\begin{array}{ll}
r^{0}_{\mathcal{T}} & = \textbf{If } C_{\mathcal{R}}(q^{0}_{\mathcal{T}}) \textbf{ Then } [q^{0}_{\mathcal{T}}, \ldots] \longrightarrow \left[q^{NA}_{\mathcal{T}}, \ldots\right] \\
r^{NA}_{\mathcal{T}} & = \textbf{If } C_{\mathcal{R}}(q^{NA}_{\mathcal{R}}) \textbf{ Then } \left[q^{NA}_{\mathcal{T}}, q^{NA}_{\mathcal{R}} =< v^{NA}_{\mathcal{R}}, \varepsilon >, \ldots\right] \longrightarrow \\
 & \left[q^{Auth}_{\mathcal{T}}, q^{NA+}_{\mathcal{R}} =< v^{NA}_{\mathcal{R}}, m^{NA}_{\mathcal{T}} : Sign >, \ldots\right] \\
r^{W_i}_{\mathcal{T}} & = \textbf{If } C_{\mathcal{R}}(q^{W_i}_{\mathcal{R}}) \textbf{ Then } \left[q^{W_i}_{\mathcal{T}}, q^{W_i}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, \varepsilon >, \ldots\right] \longrightarrow \\
 & \left[q^{Auth}_{\mathcal{T}}, q^{W_i+}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, m^{W_i}_{\mathcal{T}} : Sign >, \ldots\right]
\end{array}
$$

um receptor honesto irá alcançar estados apenas se a seguinte regra for desencadeada

$$
\begin{aligned}
r^{NA}_{\mathcal{T}} = \textbf{If } C_{\mathcal{T}}(q^{NA}_{\mathcal{T}} | q_{\mathcal{A}}) \textbf{ Then } & \left[q^{NA}_{\mathcal{T}}, q^{NA}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, \varepsilon >, \ldots\right] \\
& \longrightarrow \left[q^{Auth}_{\mathcal{T}}, q^{NA+}_{\mathcal{R}} =< v^{i}_{\mathcal{R}}, m^{NA}_{\mathcal{T}} : \mathsf{Synth}(q_{\mathcal{A}}) >, \ldots\right].
\end{aligned}
\tag{6.6}
$$

Isto é, a única opção para o adversário seria sintetizar (Synth) uma assinatura válida (no nome de uma parte honesta), porque extraiu vantagem do que observou, então ele assim violaria uma propriedade fundamental do esquema de assinatura *SDVS* considerado: não viabilidade de se forjar uma assinatura.

Ademais, deixe-se supor que o adversário pôde convencer o receptor acerca da sua identidade forjada, fazendo-o crer que o tráfego gerado por ele é legítimo, e por conseguinte levado o receptor ao um dos estados de autenticação sob a condição de que ele não tenha aprendido a forjar assinaturas. Neste caso, este adversário só irá manter este estado se puder forjá-las, caso contrário ele perderá o *status* de autenticado eventualmente.

***Invariante 2**: Um transmissor que já está identificado permanecerá identificado enquanto o desejar* — Este invariante se sustenta se o protocolo apresentar resiliência frente aos ataques de um adversário malicioso que tomou controle do canal de comunicação e está a interferir nas mensagens trocadas entre transmissor e receptor, tentando frustrar o mecanismo de autenticação.

Vale observar que os mecanismos de controle de congestionamento poderão pedir retransmissão de pacotes descartados pelo adversário, um ataque comum aos mecanismos de autenticação baseados em habilitação. Entretanto, o esquema proposto não vai se resguardar em qualquer desses recursos de resiliência da rede. Então, será considerado os ataques já apresentados no modelo adversarial e levar em conta que o adversário poderá não só descartar pacotes a sua vontade, mas também manipulá-los para eventualmente usá-los em outros ataques.

A argumentação deste segundo invariante poderá ser dada por se considerar a cadeia de Markov do modelo de estados apresentado do protocolo. Sendo assim, as regras de transição serão modeladas como na Matriz 6.7. A probabilidade de o adversário interferir em um pacote com assinatura válida e designado ao receptor é denotada por δ. À frente, será apresentado formalmente quanto δ depende dos parâmetros de segurança do protocolo.

$$P = \begin{bmatrix} & Auth & W_0 & W_1 & \dots & W_\tau & NA \\ Auth & 1-\delta & \delta & 0 & \dots & 0 & 0 \\ W_0 & 1-\delta & 0 & \delta & \dots & 0 & 0 \\ W_1 & 1-\delta & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ W_\tau & 1-\delta & 0 & 0 & \dots & 0 & \delta \\ NA & 1-\delta & 0 & 0 & \dots & 0 & \delta \end{bmatrix} \tag{6.7}$$

Na Matriz 6.7, os elementos $W_0, ..., W_\tau, NA$ referem-se aos estados descritos no autômato na Figura 17 e cada uma das células $P_{m,n}$ nesta matriz representa a probabilidade de transição entre os correspondentes estados. Baseada nesta matriz, uma análise da distribuição estacionária da cadeia de Markov será apresentada, vez que a distribuição estacionária tem relação com a função distribuição que pode modelar a chance em probabilidade do quanto o processo se mantém em cada um dos estados. Portanto, é possível inferir desta distribuição quais as chances de um transmissor legítimo ter sua autenticação frustrada, ou de mantê-la, ainda que sob interferência do adversário.

Seja o processo de Markov sobre os estados $\mathcal{S} = \{q_{\mathcal{R}}^{Auth}, q_{\mathcal{R}}^{W_0} \dots, q_{\mathcal{R}}^{W_\tau}, q_{\mathcal{R}}^{NA}\}$ e a matriz probabilidade das transições P. Considera-se que $\pi_j,\ j \in \mathcal{S}$ é uma distribuição sobre S, isto é, π é uma linha com $|S|$ elementos, tais que, $\sum_j \pi_j = 1$, $\pi_j \geq 0$, para cada $j \in S$. Então, se uma distribuição inicial π_0 faz a cadeia de Markov estacionária com distribuição π, tem-se

$$\pi_j = \pi \cdot P. \tag{6.8}$$

Desta forma, para todo $j \in S$, sendo π_j o produto interno entre π e o j-ésima coluna de

P. Tem-se

$$\pi_j = \sum_{i \in \mathcal{S}} \pi_i \cdot P_{i,j}. \tag{6.9}$$

Por sua vez, será possível encontrar uma distribuição do estado estacionário da cadeia de Markov para o modelo do protocolo proposto. Da matriz P, tem-se

$$1 = \pi_{Auth} + \pi_{W_0} + \pi_{W_1} + \ldots + \pi_{W_\tau} + \pi_{W_{NA}}, \tag{6.10}$$

$$\pi_{Auth} = (1-\delta)\cdot\pi_{Auth} + (1-\delta)\cdot\pi_0 + \ldots + (1-\delta)\cdot\pi_\tau + (1-\delta)\cdot\pi_{NA}, \tag{6.11}$$

$$\pi_{W_0} = \delta.\pi_{Auth}, \tag{6.12}$$

$$\pi_{W_1} = \delta\cdot\pi_{W_0} = \delta^2\cdot\pi_{Auth}, \tag{6.13}$$

$$\pi_{W_2} = \delta\cdot\pi_{W_1} = \delta^3\cdot\pi_{Auth}, \tag{6.14}$$

$$\vdots$$

$$\pi_{W_\tau} = \delta\cdot\pi_{W_{\tau-1}} = \delta^{\tau+1}\cdot\pi_{Auth}, \tag{6.15}$$

$$\pi_{NA} = \delta \cdot \pi_{W_\tau} + \delta \cdot \pi_{NA} = \delta^{\tau+2} \cdot \pi_{Auth} + \delta \cdot \pi_{NA}. \tag{6.16}$$

Por se fazer as substituições em 6.10, tem-se

$$\begin{cases} 1 = \pi_{Auth} + \pi_{W_0} + \pi_{W_1} + \ldots + \pi_{W_\tau} + \pi_{W_{NA}} \\ \quad = \pi_{Auth} + \delta \cdot \pi_{Auth} + \delta^2 \cdot \pi_{Auth} + \ldots + \delta^{\tau+2} \cdot \pi_{Auth} + \delta \cdot \pi_{NA} \\ \quad = \pi_{Auth} \cdot (1 + \delta + \delta^2 + \ldots + \delta^{\tau+2}) + \delta \cdot \pi_{NA} \end{cases} \tag{6.17}$$

Aplicando-se a forma geral de uma soma geométrica e fazendo-se as devidas substituições, tem-se

$$\begin{cases} \pi_{NA} = \dfrac{\delta^{(\tau+2)}}{1-\delta} \cdot \pi_{Auth}, \\ \quad 1 = \pi_{Auth} \cdot \dfrac{1-\delta^{(\tau+3)}}{1-\delta} \cdot \delta \cdot \pi_{NA}, \\ \pi_{Auth} = \dfrac{1-\delta}{1+\delta^{\tau+2}-\delta^{\tau+3}}. \end{cases} \tag{6.18}$$

Finalmente, é possível agora usar π_{Auth} para determinar qual a chance que o adversário possui de frustrar o mecanismo de autenticação

$$\pi_{NA} = \frac{\delta^{\tau+2}}{1 + \delta^{\tau+2} - \delta^{\tau+3}}. \tag{6.19}$$

Por se observar este resultado, é possível afirmar que quando o adversário não interfere nos pacotes, seja porque ele não logrou sucesso em identificar as assinaturas válidas, a probabilidade do transmissor permanecer no estado π_{Auth} dependerá da vontade dele transmissor. Caso contrário, quando uma parte maliciosa tentar interferir nas mensagens assinadas, descartando todas as assinaturas válidas com probabilidade δ, quanto mais alto for δ mais baixa será a chance de um processo permanecer em π_{auth}. Portanto, $\delta = 1$ implica em $\pi_{Auth} = 0$.

6.6.3 A dependência de δ em relação ao mecanismo de assinatura

Se o adversário realiza uma baixa taxa de interferência $\delta \ll 1$, então o processo irá estacionar no estado autenticado (o receptor terá em conta que o originador do tráfego se mantém identificado) com maior probabilidade. O argumento aqui é de que nenhuma estratégia eficiente dê ao adversário uma boa chance de frustrar o mecanismo de autenticação, exceto se o adversário descarte todos os pacotes, ou busque uma estratégia que lhe dê chances de quebrar as chaves dos esquemas de assinaturas.

A título de se oferecer aqui um exemplo numérico, supõe-se que um cenário em que os parâmetros utilizados são como a seguir: a cada sequência de $n = 100$ pacotes serão assinados $\Sigma = 10$ no total, com apenas $\sigma = 1$ dentre essas assinaturas sendo válidas e designadas ao receptor que tolerará o limiar de até $\tau = 3$ destas n sequências sem assinaturas válidas, ou seja, 300 pacotes no máximo.

Então se o adversário, por exemplo, tentar descartar apenas um pacote, ele terá $\delta = 1/10$ de probabilidade de atingir a informação de autenticação, o que implicará em $\pi_{NA} \approx 9.9e^{-6}$, isto é, uma chance de 99,999% de que o receptor permaneça em um dos estados de autenticação.

Quando se considera os mesmos parâmetros de segurança com os valores $\Sigma = 10, \Sigma = 30$ e $\Sigma = 50$, o resultado é como na Figura 18. As curvas nos gráficos demonstram a probabilidade de um receptor interromper a autenticação de um transmissor que estava legitimamente identificado.

Da análise dos gráficos, é fácil inferir que a chance do atacante evolui proporcionalmente à medida da interferência causada. Naturalmente, ele terá mais chances se interferir mais. Porém, com a escolha de parâmetros adequados, este jogo exigirá um risco maior do adversário. Nas linhas à frente, será demonstrado como o protocolo pode ser ajustado

para impor a um atacante uma estratégia em que ele tenha que arriscar tudo, ou ter suas chances minimizadas.

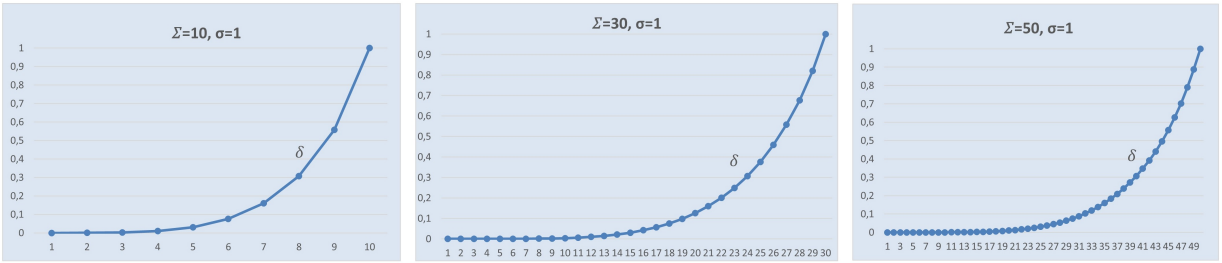


Figura 18 – Relação entre o número de pacotes que o adversário tenta manipular (abcissa), a probabilidade do receptor descartar a autenticação (δ).

Ademais, o caso em que o transmissor cria mais de uma assinatura válida deve ser analisado. Quando o transmissor definir um valor maior para $\sigma \geq 1$, número de assinaturas válidas, permitindo $\binom{\Sigma}{\sigma}$ maneiras de se dispor estas assinaturas, o desafio para o adversário vai se tornando maior, pois se ele interfere em até $\sigma - 1$ assinaturas válidas, o esquema não é abalado. O fato de que o protocolo mistura assinaturas válidas e não válidas amplifica, portanto, a resistência do protocolo. Por exemplo, quando se combina $\Sigma = 10$ e $\sigma = 2$, é possível ter $\binom{10}{2} = 42$ maneiras de como dispor todas assinaturas no meio do tráfego. Portanto, enquanto $\sigma = 1$ requer que um atacante tenha que manipular apenas cinco pacotes para ganhar 50% de chances de acertar um assinado com validade, quando $\sigma = 2$ o adversário irá precisar interferir em pelo menos sete pacotes para que adquira a mesma probabilidade de sucesso.

Outro exemplo será analisado por se definir $\Sigma = 20$, $\sigma = 5$ e $n = 1000$. Neste caso, o transmissor terá $\binom{20}{5} = 15.504$ maneiras diferentes de dispor assinaturas válidas e não válidas entre os pacotes. A Figura 19 ilustra como a chance do adversário evolui com o aumento da sua interferência no tráfego, por se incrementar a investida de 1 a 20 pacotes a serem alvejados. Uma análise direta naquele gráfico leva a conclusão de que a interferência em menos de cinco pacotes terá uma chance praticamente nula de frustrar o esquema de autenticação.

Estes exemplos fornecem uma demonstração visual que harmoniza com os resultados demonstrados em (6.18) e (6.19), reafirmando que o processo se mantém um estado estacionário coerente com a resiliência que se quer demonstrar. Então, sempre que o transmissor desejar estabelecer uma autenticação para com um receptor, ele irá mantê-la com sucesso até que ele mesmo decida encerrá-la.

A resiliência do protocolo é ainda mais reforçada quando se define uma taxa maior de assinaturas válidas, o que pode ser amplificado quando se é maximizado o número de maneiras que assinaturas podem ser dispostas no tráfego, isto é, quando $\sigma \approx \Sigma/2$ essa

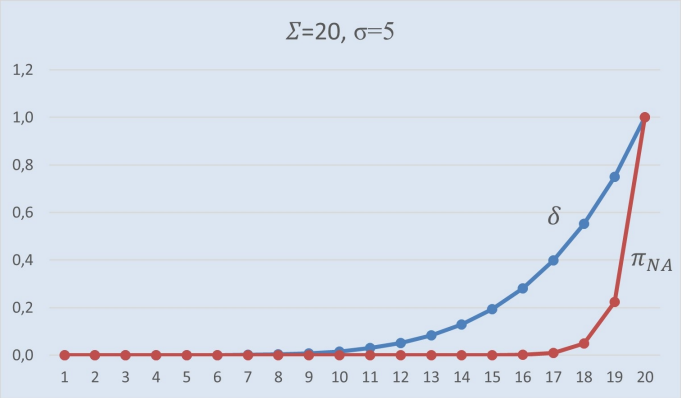


Figura 19 – Relação entre o número de pacotes manipulados (abcissa), a probabilidade de atingir os validamente assinados (δ) e a probabilidade de invalidar uma autenticação (π_{NA}).

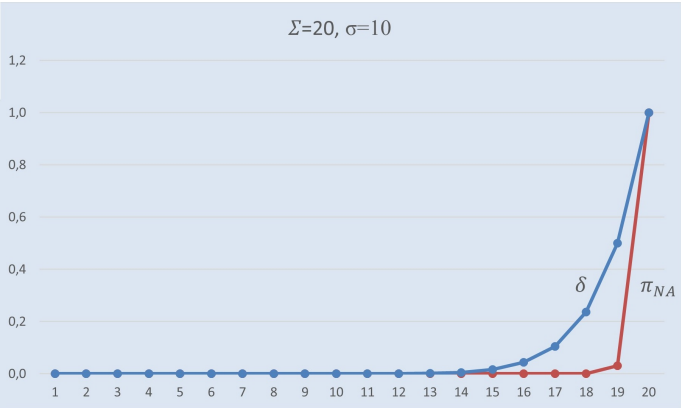


Figura 20 – Ganho quando o número de combinação possível da mistura de assinaturas válidas e não válidas é maximizado.

quantidade de combinações é maximizada. Deste modo, se o adversário tentar interferir em um número menor do que o total $d < \Sigma$ de assinaturas, ainda que até $d = \Sigma - 1$, as chances de sucesso estarão reduzidas à probabilidade

$$\frac{\binom{\Sigma}{\sigma}}{\binom{\Sigma-1}{\sigma}} \cong 2, \text{quando } \sigma \cong \Sigma/2. \tag{6.20}$$

Portanto, quando se permitem parâmetros que proporcionam uma combinação máxima de maneiras de se mesclar as assinaturas no tráfego, o que é considerado no cálculo na Equação 6.20, fica demonstrado que se o adversário não tentar interferir em todos os pacotes assinados, as chances que ele terá de realmente atingir as assinaturas válidas será pequena. Até mesmo se ele fizer a menos de um, $\Sigma - 1$ pacotes, a chance será de 1/2. Esta habilidade de esconder assinaturas válidas entre aleatórias tem um efeito de exigir do adversário um tudo ou nada, o que pode denunciá-lo. conforme o ilustrado na Figura 20.

7 Discussão e conclusão

7.1 Discussão

A solução apresentada supõe um modelo adversarial o qual considera que qualquer parte legítima com poder de gerar tráfego na rede pode ser completamente tomada pelo adversário. Uma discussão necessária é a de se considerar as partes que já foram legitimadas para com um destinatário, e que portanto já são cotadas por ele como de boa reputação, e então se avaliar o caso em que o adversário toma o controle dessas partes, examinando-se de como o esquema resistiria a um ataque como este. Este cenário é uma real ameaça, como já abordado no Capítulo 2, tanto que os ataques recentes às redes de *IoT* têm mostrado como são vulneráveis os dispositivos que nelas funcionam.

Em primeiro plano, é importante frisar que nenhuma solução de segurança pode se basear na hipótese de que sistemas são invulneráveis e podem executar livre de ameaças. É importante destacar aqui que nenhuma solução de *DoS* pode resolver os problemas de segurança dos dispositivos. Este problema é uma questão separada da proteção de tráfego. A solução apresentada aqui supõe justamente o pior caso em que o dispositivo é tomado, e constrói uma defesa baseada na assinatura de tráfego, o que é usado para se filtrar quem tem uma boa reputação. Considera-se aqui que a proteção dos dispositivos é uma questão de quem os administra, e supõe-se até que mesmo diante da eventualidade de invasão ou tomada de controle deles pelo adversário o esquema proposto aqui ainda funcione. Como argumentado no texto, o mecanismo apresentado prescinde de um esquema de detecção de ataques *DoS*. Portanto, diante de uma ameaça percebida pela detecção, o esquema proposto poderia atuar, permitindo apenas quem permanecesse com a reputação de um originador de tráfego bem-comportado.

Ainda mais, o argumento é que o esquema sempre identifique um originador de tráfego. E sempre ele conseguirá fazê-lo: ainda que um adversário, estrategicamente, escolha chaves privadas ao seu gosto, o tráfego gerado por ele será identificado. E ainda que este adversário tenha furtado chaves de partes honestas e as explore para criar um pesado volume de rede, todo esse tráfego poderá ser bloqueado pelo esquema, pois seria de qualquer forma identificado e elencado como um fluxo não desejado.

Diante de um grande conluio na rede enviando tráfego malicioso, o receptor do tráfego poderia ativar o esquema e exigir que aquelas partes tomadas pelo adversário, até então bem avaliados transmissores, mudassem suas chaves internas. É certo que para o adversário fazer esta mudança pontualmente em alguns dispositivos tomados pode não ser difícil, mas coordenar esta mudança em grande escala seria um grande desafio. Mas

ainda que o adversário não responda a esta exigência, ou ainda que ele decida mudar periodicamente as chaves de assinatura, o receptor poderia decidir por aceitar apenas o tráfego de boa reputação.

Outrossim, é importante frisar aqui que o esquema proposto não é rigorosamente um mecanismo de autenticação, na definição formal de autenticação apresentada no Capítulo 4, a despeito de terem sido indiscriminadamente empregadas no texto as expressões *autenticado* e *autenticação*. O esquema proposto identifica uma origem, mas sem a preocupação de dar garantias da sua real identidade, seja empresa ou indivíduo. O que de fato se alcança com o esquema é um meio de se proporcionar que uma origem tenha uma *marca*, o que funciona exatamente porque ela não precisa ser verificada por uma cadeia de servidores de certificação digital. O esquema é apropriado porque ele cria um meio de uma fonte se mostrar distinta: de se permitir que o tráfego gerado por ela seja distinto dos demais. É conveniente que essa identidade seja volátil, fácil de mudar, pois essa propriedade permite que o esquema não dependa de uma chave secreta que tenha que ser memorizada pelo usuário, nem dependa de um certificado digital emitido por autoridades certificadoras.

Ainda sobre esse aspecto de chave privada no assinador de tráfego, o esquema se mostra interessante porque essa chave não precisa ser forte, pois se ela for tomada ou invertida pelo adversário, o esquema sempre definirá um filtro que o receptor administrará. De fato, ainda que a chave seja escolhida fraca e o adversário ao observar assinaturas pudesse invertê-la, o receptor poderia exigir que a origem mudasse a chave. Olhando por esse lado, o esquema parece construir uma prova de trabalho ao inverso: o adversário é quem teria que gastar tempo computacional se quisesse ter tráfego aceito; os usuários legitimados continuariam simplesmente assinando o tráfego.

7.2 Conclusão

A proposta de um mecanismo de defesa contra ataques DoS descrita nesta tese introduz um modelo de identificação de tráfego entre um transmissor e um receptor, o qual poderá ser aplicado em um contexto de redes de computadores baseadas em troca de pacotes. A solução apresentada é inovativa na maneira de usar assinaturas digitais, e portanto a contribuição trazida neste trabalho advém da maneira original que criptografia é aplicada como uma ferramenta para se construir um mecanismo de identificação de tráfego, o que pode ter uma aplicação mais abrangente. Este mecanismo proporciona uma identificação da fonte do fluxo de rede tal que um eficiente esquema de defesa *DoS* seja possível, pois não considera uma infraestrutura de chave pública e demanda a aplicação de esquemas criptográficos de baixa complexidade computacional.

O modelo é apresentado com uma argumentação de segurança, vez que é formali-

zado uma medida da sua resistência face às conhecidas estratégias de ataque *DoS*. Este argumento é concretizado pela especificação formal do protocolo proposto, a qual é baseada em um modelo de autômato finitos. Uma análise probabilística é apresentada pela confrontação do esquema de defesa com o modelo adversarial apresentado, mostrando-se assim as chances de um atacante burlar o esquema e uma medida das chances de realizá-lo com sucesso.

Referências

1 KOLIAS, C. et al. DDoS in the IoT: Mirai and other botnets. *Computer*, v. 50, n. 7, p. 80–84, 2017. ISSN 0018-9162. Citado na página 27.

2 BERTINO, E.; ISLAM, N. Botnets and Internet-of- Things security. *Computer*, IEEE, v. 50, n. 2, p. 76–79, 2017. Citado 2 vezes nas páginas 27 e 34.

3 LYU, M. et al. Quantifying the Reflective DDoS Attack Capability of Household IoT Devices. In: *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. New York, NY, USA: ACM, 2017. (WiSec '17), p. 46–51. ISBN 978-1-4503-5084-6. Citado na página 27.

4 AHN, J. H. et al. Computing on Authenticated Data. *Journal of Cryptology*, v. 28, n. 2, p. 351–395, abr. 2015. Citado na página 27.

5 ATENIESE, G. et al. Sanitizable signatures. In: VIMERCATI, S. d. C. di; SYVERSON, P.; GOLLMANN, D. (Ed.). *Computer Security – ESORICS 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 159–177. Citado na página 27.

6 HANSER, C.; SLAMANIG, D. Blank digital signatures. In: *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*. New York, NY, USA: ACM, 2013. (ASIA CCS '13), p. 95–106. Citado na página 27.

7 SYTA, E. et al. Keeping authorities "honest or bust" with decentralized witness cosigning. In: *2016 IEEE Symposium on Security and Privacy (SP)*. San Jose, CA: IEEE, 2016. p. 526–545. Citado na página 27.

8 DRIJVERS, M. et al. *On the Provable Security of Two-Round Multi-Signatures*. 2018. Disponível em: <https://eprint.iacr.org/2018/417>. Citado na página 27.

9 NAKAMOTO, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Citado 2 vezes nas páginas 27 e 42.

10 GOLDREICH, O. *Foundations of Cryptography: Volume 1*. New York, NY, USA: Cambridge University Press, 2006. ISBN 0521035368. Citado 5 vezes nas páginas 28, 50, 59, 60 e 74.

11 IACOVAZZI, A.; ELOVICI, Y. Network flow watermarking: A survey. *IEEE Communications Surveys Tutorials*, v. 19, n. 1, p. 512–530, 2017. Citado na página 29.

12 AHMED, M.; MAHMOOD, A. N.; HU, J. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, v. 60, p. 19–31, 2016. Citado na página 29.

13 YU, K. et al. Internet traffic identification based on community detection by label propagation. In: *IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*. Hangzhou, China: [s.n.], 2012. v. 2, p. 786–791. Citado na página 29.

14 DAINOTTI, A.; PESCAPE, A.; CLAFFY, K. C. Issues and future directions in traffic classification. *IEEE Network*, v. 26, n. 1, p. 35–40, 2012. Citado 2 vezes nas páginas 29 e 30.

15 ZHOU, D. et al. A survey on network data collection. *Journal of Network and Computer Applications*, v. 116, p. 9–23, 2018. Citado na página 30.

16 ZHAO, P. et al. ILLIA: Enabling k-anonymity-based privacy preserving against location injection attacks in continuous lbs queries. *IEEE Internet of Things Journal*, v. 5, n. 2, p. 1033–1042, 2018. Citado na página 30.

17 STOJANOV, R. et al. Linked data authorization platform. *IEEE Access*, v. 6, p. 1189–1213, 2018. Citado na página 30.

18 SYMANTEC. *Internet Security Threat Report (ISTR) 2018 | Symantec*. [S.l.], 2018. Disponível em: <https://www.symantec.com/security-center/threat-report>. Citado 4 vezes nas páginas 30, 32, 33 e 36.

19 HATTON, L.; JOHN, A. Delivering genuine emails in an ocean of spam. *IEEE Software*, v. 34, n. 4, p. 11–15, 2017. Citado na página 31.

20 GLIGOR, V. D. A note on denial-of-service in operating systems. *IEEE Transactions on Software Engineering*, SE-10, n. 3, p. 320–324, 1984. Citado na página 31.

21 ZARGAR, S. T.; JOSHI, J.; TIPPER, D. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. *IEEE Communications Surveys Tutorials*, v. 15, n. 4, p. 2046–2069, 2013. Citado na página 31.

22 FENG, W.-c. The case for TCP/IP puzzles. In: *ACM SIGCOMM Workshop on Future Directions in Network Architecture*. [S.l.]: ACM, 2003. (FDNA '03), p. 322–327. Citado na página 31.

23 VERISIGN. *DDoS Report On DDoS Attack Trends And Insights - Verisign*. Disponível em: <https://www.verisign.com/en_US/security-services/ddos-protection/ddos-report/index.xhtml>. Citado 5 vezes nas páginas 31, 32, 33, 34 e 35.

24 KASPERSKY. *DDoS attacks in Q1 2018*. Disponível em: <https://securelist.com/ddos-report-in-q1-2018/85373/>. Citado na página 32.

25 WANG, B. et al. DDoS attack protection in the era of cloud computing and software-defined networking. *Computer Networks*, v. 81, p. 308–319, 2015. Citado na página 32.

26 HERON, S. Botnet command and control techniques. *Network Security*, v. 2007, n. 4, p. 13–16, 2007. Citado na página 32.

27 ATZORI, L.; IERA, A.; MORABITO, G. The Internet-of-Things: A survey. *Computer Networks*, v. 54, n. 15, p. 2787–2805, 2010. Citado na página 32.

28 DOULIGERIS, C.; MITROKOTSA, A. DDoS attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks*, v. 44, n. 5, p. 643–666, 2004. Citado 4 vezes nas páginas 32, 35, 36 e 39.

29 BONGUET, A.; BELLAICHE, M. A survey of denial-of-service and distributed denial of service attacks and defenses in cloud computing. *Future Internet*, v. 9, n. 3, p. 43. Citado na página 32.

30 HOQUE, N.; BHATTACHARYYA, D. K.; KALITA, J. K. Botnet in DDoS attacks: Trends and challenges. *IEEE Communications Surveys Tutorials*, v. 17, n. 4, p. 2242–2270, 2015. Citado 2 vezes nas páginas 32 e 45.

31 WANG, H.; ZHANG, D.; SHIN, K. G. Detecting SYN flooding attacks. In: *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*. [S.l.: s.n.], 2002. p. 1530–1539. Citado na página 34.

32 LEMON, J. Resisting SYN flood DoS attacks with a syn cache. In: *BSD Conference 2002 on BSD Conference*. Berkeley, CA, USA: USENIX Association, 2002. (BSDC'02), p. 10–10. Citado na página 34.

33 CHEN, W.; YEUNG, D.-Y. Defending against TCP SYN flooding attacks under different types of IP spoofing. In: *International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies*. [S.l.: s.n.], 2006. ((ICNICONSMCL'06)), p. 38–38. Citado na página 34.

34 PENG, T.; LECKIE, C.; RAMAMOHANARAO, K. Survey of network-based defense mechanisms countering the DoS and DDoS problems. *ACM Comput. Surv.*, v. 39, n. 1, 2007. Citado 4 vezes nas páginas 34, 35, 36 e 37.

35 PAXSON, V. An analysis of using reflectors for distributed denial-of-service attacks. *SIGCOMM Comput. Commun. Rev.*, v. 31, n. 3, p. 38–47, 2001. Citado 2 vezes nas páginas 35 e 81.

36 ZHOU, C. V.; LECKIE, C.; KARUNASEKERA, S. A survey of coordinated attacks and collaborative intrusion detection. *Computers & Security*, v. 29, n. 1, p. 124–140, 2010. Citado 2 vezes nas páginas 35 e 36.

37 SCAIFE, N. et al. Cryptolock (and drop it): Stopping ransomware attacks on user data. In: *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*. [S.l.: s.n.], 2016. p. 303–312. Citado na página 36.

38 FERGUSON, P.; SENIE, D. *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing Internet Requests for Comments, RFC Editor, RFC 2827.* 2000. Disponível em: <https://www.ietf.org/rfc/rfc2827.txt>. Citado na página 36.

39 KALKAN, K.; GÜRKAN, G.; ALAGÖZ, F. Filtering-based defense mechanisms against DDoS attacks: A survey. *IEEE Systems Journal*, v. 11, n. 4, p. 2761–2773, 2017. Citado 2 vezes nas páginas 36 e 45.

40 YOU, Y.; ZULKERNINE, M.; HAQUE, A. Detecting flooding-based DDoS attacks. In: *2007 IEEE International Conference on Communications*. [S.l.: s.n.], 2007. p. 1229–1234. Citado na página 37.

41 PENG, T.; LECKIE, C.; RAMAMOHANARAO, K. Proactively detecting distributed denial of service attacks using source ip address monitoring. In: *Networking 2004*. [S.l.]: Springer, Berlin, Heidelberg, 2004, (Lecture Notes in Computer Science). p. 771–782. Citado na página 37.

42 GIL, T. M.; POLETTO, M. MULTOPS: A data-structure for bandwidth attack detection. In: *10th Conference on USENIX Security Symposium - Volume 10*. Berkeley, CA, USA: USENIX Association, 2001. (SSYM'01). Citado 2 vezes nas páginas 37 e 39.

43 PARK, K.; LEE, H. On the effectiveness of route-based packet filtering for distributed dos attack prevention in power-law internets. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 31, n. 4, p. 15–26, 2001. Citado na página 37.

44 ANDERSON, T.; ROSCOE, T.; WETHERALL, D. Preventing Internet Denial-of-service with capabilities. *SIGCOMM Comput. Commun. Rev.*, v. 34, n. 1, p. 39–44, 2004. Citado na página 38.

45 YANG, X.; WETHERALL, D.; ANDERSON, T. A DoS-limiting network architecture. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 35, n. 4, p. 241–252, 2005. Citado 2 vezes nas páginas 38 e 45.

46 KAMBHAMPATI, V.; PAPADOPOULOS, C.; MASSEY, D. A taxonomy of capabilities based DDoS defense architectures. In: *2011 9th IEEE/ACS International Conference on Computer Systems and Applications (AICCSA)*. [S.l.: s.n.], 2011. p. 157–164. Citado na página 38.

47 LIU, X.; YANG, X.; LU, Y. To filter or to authorize: Network-layer DoS defense against multimillion-node botnets. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 38, n. 4, p. 195–206, 2008. Citado na página 38.

48 WANG, L.; WU, Q.; LUONG, D. D. Engaging edge networks in preventing and mitigating undesirable network traffic. In: *2007 3rd IEEE Workshop on Secure Network Protocols*. [S.l.: s.n.], 2007. p. 1–6. Citado 2 vezes nas páginas 38 e 45.

49 YAAR, A.; PERRIG, A.; SONG, D. SIFF: a stateless internet flow filter to mitigate DDoS flooding attacks. In: *IEEE Symposium on Security and Privacy*. [S.l.: s.n.], 2004. p. 130–143. Citado 2 vezes nas páginas 38 e 45.

50 NATU, M.; MIRKOVIC, J. Fine-grained capabilities for flooding DDoS defense using client reputations. In: *2007 Workshop on Large Scale Attack Defense*. New York, NY, USA: ACM, 2007. (LSAD '07), p. 105–112. Citado na página 39.

51 DIFFIE, W.; HELLMAN, M. New directions in cryptography. *IEEE Trans. Inf. Theor.*, IEEE Press, Piscataway, NJ, USA, v. 22, n. 6, p. 644–654, 2006. Citado na página 39.

52 LEE, K. et al. DDoS attack detection method using cluster analysis. *Expert Systems with Applications*, v. 34, n. 3, p. 1659–1665, 2008. Citado na página 39.

53 BORAH, S. et al. Advanced clustering based intrusion detection (ACID) algorithm. In: *Advances in Computing and Communications*. [S.l.]: Springer, Berlin, Heidelberg, 2011. (Communications in Computer and Information Science), p. 35–43. Citado na página 39.

54 BUTUN, I.; RA, I.-H.; SANKAR, R. An intrusion detection system based on multi-level clustering for hierarchical wireless sensor networks. *Sensors*, v. 15, n. 11, p. 28960–28978, 2015. Citado na página 39.

55 COVER, T. M.; THOMAS, J. A. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. New York, NY, USA: Wiley-Interscience, 2006. ISBN 0471241954. Citado na página 39.

56 TAN, Z. et al. A system for Denial-of-Service attack detection based on multivariate correlation analysis. *IEEE Transactions on Parallel and Distributed Systems*, v. 25, n. 2, p. 447–456, 2014. Citado na página 39.

57 HOQUE, N.; BHATTACHARYYA, D. K.; KALITA, J. K. A novel measure for low-rate and high-rate DDoS attack detection using multivariate data analysis. In: *2016 8th International Conference on Communication Systems and Networks*. [S.l.: s.n.], 2016. (COMSNETS), p. 1–2. Citado na página 39.

58 YAN, Q. et al. A multi-level DDoS mitigation framework for the industrial Internet of Things. *IEEE Communications Magazine*, v. 56, n. 2, p. 30–36, 2018. Citado na página 39.

59 KESAVAMOORTHY, R.; SOUNDAR, K. R. Swarm intelligence based autonomous DDoS attack detection and defense using multi agent system. *Cluster Computing*, p. 1–8, 2018. Citado na página 39.

60 ALQAHTANI, S. M.; JOHN, R. A comparative analysis of different classification techniques for cloud intrusion detection systems'alerts and fuzzy classifiers. In: *2017 Computing Conference*. [S.l.: s.n.], 2017. p. 406–415. Citado 2 vezes nas páginas 39 e 40.

61 OSANAIYE, O. A.; ALFA, A. S.; HANCKE, G. P. Denial of service defence for resource availability in wireless sensor networks. *IEEE Access*, v. 6, p. 6975–7004, 2018. Citado 2 vezes nas páginas 40 e 45.

62 GENDREAU, A. A.; MOORMAN, M. Survey of intrusion detection systems towards an end to end secure internet of things. In: *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*. [S.l.: s.n.], 2016. p. 84–90. Citado na página 40.

63 ZARPELãO, B. B. et al. A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications*, v. 84, p. 25–37, 2017. Citado na página 40.

64 PA, Y. M. P. et al. IoTPOT: A novel honeypot for revealing current IoT threats. *Journal of Information Processing*, v. 24, n. 3, p. 522–533, 2016. Citado na página 40.

65 GUPTA, N.; NAIK, V.; SENGUPTA, S. A firewall for Internet of Things. In: *2017 9th International Conference on Communication Systems and Networks*. [S.l.: s.n.], 2017. ((COMSNETS)), p. 411–412. Citado na página 40.

66 BRACHMANN, M. et al. End-to-end transport security in the IP-based internet of things. In: *2012 21st International Conference on Computer Communications and Networks*. [S.l.: s.n.], 2012. ((ICCCN)), p. 1–5. Citado na página 40.

67 JANG, S. et al. An efficient device authentication protocol without certification authority for internet of things. *Wireless Personal Communications*, v. 91, n. 4, p. 1681–1695, 2016. Citado na página 40.

68 PORAMBAGE, P. et al. Two-phase authentication protocol for wireless sensor networks in distributed IoT applications. In: *2014 IEEE Wireless Communications and Networking Conference (WCNC)*. [S.l.: s.n.], 2014. p. 2728–2733. Citado na página 40.

69 FAROOQI, A. H. et al. A novel intrusion detection framework for wireless sensor networks. *Personal and Ubiquitous Computing*, v. 17, n. 5, p. 907–919, 2013. Citado na página 40.

70 SALMON, H. M. et al. Intrusion detection system for wireless sensor networks using danger theory immune-inspired techniques. *International Journal of Wireless Information Networks*, v. 20, n. 1, p. 39–66, 2013. Citado na página 40.

71 KARN, P.; SIMPSON, W. *Photuris: Session-Key Management Protocol*. Disponível em: <https://tools.ietf.org/html/rfc2522>. Citado na página 41.

72 CARREL, D.; HARKINS, D. *The Internet Key Exchange (IKE)*. Disponível em: <https://tools.ietf.org/html/rfc2409>. Citado na página 41.

73 SIMPSON, W. *IKE/ISAKMP Considered Dangerous*. Disponível em: <https://tools.ietf.org/html/draft-simpson-danger-isakmp-01>. Citado na página 41.

74 MEADOWS, C. A formal framework and evaluation method for network denial of service. In: *12th IEEE Computer Security Foundations Workshop*. [S.l.: s.n.], 1999. p. 4–13. Citado na página 41.

75 DWORK, C.; NAOR, M. Pricing via processing or combatting junk mail. In: *Advances in Cryptology — CRYPTO 92*. [S.l.]: Springer, Berlin, Heidelberg, 1992. (Lecture Notes in Computer Science), p. 139–147. Citado 2 vezes nas páginas 41 e 42.

76 GROZA, B.; WARINSCHI, B. Cryptographic puzzles and DoS resilience, revisited. *Designs, Codes and Cryptography*, v. 73, n. 1, p. 177–207, 2014. Citado na página 42.

77 JUELS, A.; BRAINARD, J. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In: *Network and Distributed Systems Security Symposium*. San Diego, CA: [s.n.], 1999. ((NDSS)), p. 151–165. Citado na página 42.

78 BACK, A. *Hashcash - A Denial-of-Service Counter-Measure*. 2002. Disponível em: <https://blockchainpapers.org/items/show/9>. Citado na página 42.

79 PARNO, B. et al. Portcullis: Protecting connection setup from Denial-of-capability attacks. In: *2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. New York, NY, USA: ACM, 2007. (SIGCOMM '07), p. 289–300. Citado 2 vezes nas páginas 42 e 81.

80 DEAN, D.; STUBBLEFIELD, A. Using client puzzles to protect TLS. In: *10th Conference on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2001. (SSYM'01). Citado na página 42.

81 WATERS, B. et al. New client puzzle outsourcing techniques for DoS resistance. In: *11th ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2004. (CCS '04), p. 246–256. Citado na página 42.

82 WANG, X.; REITER, M. K. Defending against denial-of-service attacks with puzzle auctions. In: *2003 Symposium on Security and Privacy, 2003.* [S.l.: s.n.], 2003. p. 78–92. Citado na página 42.

83 WU, Y. et al. Software Puzzle: A Countermeasure to resource-inflated Denial-of-Service attacks. *IEEE Transactions on Information Forensics and Security*, v. 10, n. 1, p. 168–177, 2015. Citado na página 42.

84 DZIEMBOWSKI, S. et al. Proofs of Space. In: *Advances in Cryptology — CRYPTO 2015*. [S.l.]: Springer, Berlin, Heidelberg, 2015. (Lecture Notes in Computer Science), p. 585–605. Citado na página 42.

85 GUPTA, D.; SAIA, J.; YOUNG, M. Proof of Work without all the work. In: *19th International Conference on Distributed Computing and Networking*. [S.l.]: ACM Press, 2018. (ICDCN '18), p. 1–10. Citado na página 43.

86 BLOOM, B. H. Space/Time trade-offs in hash coding with allowable errors. *Commun. ACM*, v. 13, n. 7, p. 422–426, 1970. Citado na página 43.

87 JOKELA, P. et al. LIPSIN: Line speed publish/subscribe inter-networking. In: *ACM SIGCOMM 2009 Conference on Data Communication*. New York, NY, USA: ACM, 2009. (SIGCOMM '09), p. 195–206. Citado na página 44.

88 ANTIKAINEN, M.; AURA, T.; SäRELä, M. Denial-of-Service attacks in Bloom-filter-based forwarding. *IEEE/ACM Transactions on Networking*, v. 22, n. 5, p. 1463–1476, 2014. Citado na página 44.

89 WANG, C. et al. SkyShield: A sketch-based defense system against application layer DDoS attacks. *IEEE Transactions on Information Forensics and Security*, v. 13, n. 3, p. 559–573, 2018. Citado na página 44.

90 ROTHENBERG, C. E. et al. Self-routing Denial-of-Service resistant capabilities using In-packet Bloom filters. In: *2009 European Conference on Computer Network Defense*. Washington, DC, USA: IEEE Computer Society, 2009. (EC2ND '09), p. 46–51. Citado na página 44.

91 ROTTENSTREICH, O.; KESLASSY, I. The Bloom paradox: When not to use a Bloom filter. *IEEE/ACM Transactions on Networking*, v. 23, n. 3, p. 703–716, 2015. Citado na página 44.

92 QIAN, J.; ZHU, Q.; WANG, Y. Bloom filter based associative deletion. *IEEE Transactions on Parallel and Distributed Systems*, v. 25, n. 8, p. 1986–1998, 2014. Citado na página 44.

93 GAO, Z.; ANSARI, N. Tracing cyber attacks from the practical perspective. *IEEE Communications Magazine*, v. 43, n. 5, p. 123–131, 2005. Citado na página 44.

94 SINGH, K.; SINGH, P.; KUMAR, K. A systematic review of IP traceback schemes for Denial of Service attacks. *Computers & Security*, v. 56, p. 111–139, 2016. Citado na página 44.

95 SNOEREN, A. C. et al. Hash-based IP traceback. In: *2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. New York, NY, USA: ACM, 2001. (SIGCOMM '01), p. 3–14. Citado na página 45.

96 JIN, G. et al. A hash-based path identification scheme for DDoS attacks defense. In: *2009 Ninth IEEE International Conference on Computer and Information Technology*. [S.l.: s.n.], 2009. v. 2, p. 219–224. Citado na página 45.

97 CHENG, L. et al. FACT: A framework for authentication in cloud-based IP traceback. *IEEE Transactions on Information Forensics and Security*, v. 12, n. 3, p. 604–616, 2017. Citado na página 45.

98 ARGYRAKI, K.; CHERITON, D. R. Network capabilities: The good, the bad and the ugly. In: *Fourth Workshop on Hot Topics in Networks*. Maryland, US: ACM Press, 2005. ((HotNets-IV)). Citado na página 45.

99 ARGYRAKI, K.; CHERITON, D. Scalable network-layer defense against Internet bandwidth-flooding attacks. *IEEE/ACM Transactions on Networking*, v. 17, n. 4, p. 1284–1297, 2009. Citado na página 45.

100 GOLDWASSER, S.; MICALI, S. Probabilistic encryption. *Journal of Computer and System Sciences*, v. 28, n. 2, p. 270–299, 1984. Citado 3 vezes nas páginas 47, 51 e 53.

101 HAZAY, C.; LINDELL, Y. *Efficient Secure Two-Party Protocols: Techniques and Constructions*. 1st. ed. New York, NY, USA: Springer-Verlag New York, Inc., 2010. ISBN 3642143024, 9783642143021. Citado na página 47.

102 KATZ, J.; LINDELL, Y. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. [S.l.]: Chapman & Hall/CRC, 2007. ISBN 1584885513. Citado na página 47.

103 BEN-OR, M.; GOLDWASSER, S.; WIGDERSON, A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: *Twentieth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 1988. (STOC '88), p. 1–10. Citado na página 47.

104 CHAUM D. CRÉPEAU, C.; DAMGÅRD, I. Multiparty unconditionally secure protocols. In: *Twentieth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 1988. (STOC '88), p. 11–19. Citado na página 47.

105 GOLDREICH, O. On expected probabilistic polynomial-time adversaries: A suggestion for restricted definitions and their benefits. In: *Theory of Cryptography: 4th Theory of Cryptography Conference*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. (TCC 2007), p. 174–193. Citado na página 48.

106 AUMANN, Y.; LINDELL, Y. Security against covert adversaries: Efficient protocols for realistic adversaries. *Journal of Cryptology*, v. 23, n. 2, p. 281–343, 2010. Citado na página 49.

107 CANETTI, R. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 13, n. 1, p. 143–202, 2000. Citado 2 vezes nas páginas 50 e 60.

108 CANETTI, R. Universally composable security: A new paradigm for cryptographic protocols. In: *42nd IEEE Symposium on Foundations of Computer Science*. Washington, DC, USA: IEEE Computer Society, 2001. (FOCS '01), p. 136–. Citado 2 vezes nas páginas 50 e 60.

109 BACKES, M.; PFITZMANN, B.; WAIDNER, M. A general composition theorem for secure reactive systems. In: *Theory of Cryptography*. [S.l.]: Springer, Berlin, Heidelberg, 2004. (Lecture Notes in Computer Science), p. 336–354. Citado na página 50.

110 BARAK, B. How to go beyond the black-box simulation barrier. In: *42Nd IEEE Symposium on Foundations of Computer Science*. Washington, DC, USA: IEEE Computer Society, 2001. (FOCS '01), p. 106–. Citado na página 50.

111 SHANNON, C. Communication theory of secrecy systems. *Bell System Technical Journal*, p. 656–715, 1949. Citado na página 50.

112 WHITFIELD, D.; MARTIN, E. H. New directions in cryptography. *IEEE Transactions on Information Theory*, v. 22, n. 6, p. 644–654, 1976. Citado na página 51.

113 RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, ACM, New York, NY, USA, v. 21, n. 2, p. 120–126, fev. 1978. Citado na página 51.

114 SHOUP, V. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology EPrint Archive*, v. 2004, p. 332, 2004. Citado na página 52.

115 LINDELL, Y. How to simulate it? A tutorial on the simulation proof technique. *IACR Cryptology ePrint Archive*, 2016. Citado 3 vezes nas páginas 52, 56 e 60.

116 BELLARE, M.; ROGAWAY, P. The game-playing technique. *IACR Cryptology ePrint Archive*, 2004. Citado na página 52.

117 KILIAN, J.; ROGAWAY, P. How to protect DES against exhaustive key search. In: *Advances in Cryptology — CRYPTO '96: 16th Annual International Cryptology Conference*. Berlin, Heidelberg: Springer, 1996. p. 252–267. Citado na página 52.

118 BELLARE, M.; ROGAWAY, P. Random oracles are practical: A paradigm for designing efficient protocols. In: *1st ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 1993. (CCS '93), p. 62–73. Citado na página 54.

119 CANETTI, R.; GOLDREICH, O.; HALEVI, S. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, ACM, New York, NY, USA, v. 51, n. 4, p. 557–594, 2004. Citado na página 54.

120 BONEH, D.; BOYEN, X. Short signatures without random oracles. In: CACHIN, C.; CAMENISCH, J. L. (Ed.). *Advances in Cryptology - EUROCRYPT 2004*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 56–73. Citado na página 55.

121 GOLDREICH, O. *Foundations of Cryptography: Volume 2, Basic Applications*. New York, NY, USA: Cambridge University Press, 2004. ISBN 0521830842. Citado 5 vezes nas páginas 56, 59, 60, 61 e 62.

122 PASS, R. Bounded-concurrent secure multi-party computation with a dishonest majority. In: *Thirty-sixth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 2004. (STOC '04), p. 232–241. Citado na página 60.

123 KATZ, J. *Digital signatures*. New York: Springer, 2010. ISBN 978-0-387-27711-0. Citado 2 vezes nas páginas 62 e 63.

124 POINTCHEVAL, D.; STERN, J. Security arguments for digital signatures and blind signatures. *Journal Cryptology*, v. 13, n. 3, p. 361–396, 2000. Citado na página 64.

125 CHAUM, D.; ANTWERPEN, H. van. Undeniable signatures. In: *Advances in Cryptology*. New York, NY, USA: Springer-Verlag New York, Inc., 1989. (CRYPTO '89), p. 212–216. Citado na página 64.

126 JAKOBSSON, M.; SAKO, K.; IMPAGLIAZZO, R. Designated verifier proofs and their applications. In: *15th Annual International Conference on Theory and Application of Cryptographic Techniques*. Berlin, Heidelberg: Springer-Verlag, 1996. (EUROCRYPT'96), p. 143–154. Citado 3 vezes nas páginas 65, 66 e 73.

127 DWORK, C. Differential privacy. In: *Proceedings of Automata, Languages and Programming: 33rd International Colloquium*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. (ICALP 2006), p. 1–12. Citado na página 70.

128 SWEENEY, L. Achieving K-anonymity privacy protection using generalization and suppression. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, World Scientific Publishing Co., Inc., River Edge, NJ, USA, v. 10, n. 5, p. 571–588, out. 2002. Citado na página 70.

129 SAEEDNIA, S.; KREMER, S.; MARKOWITCH, O. An efficient strong designated verifier signature scheme. In: *Information Security and Cryptology - ICISC 2003*. [S.l.]: Springer, Berlin, Heidelberg, 2003. (Lecture Notes in Computer Science), p. 40–54. Citado na página 73.

130 ALMEIDA, M. P. de et al. New DoS defense method based on strong designated verifier signatures. *Sensors*, v. 18, n. 9, 2018. ISSN 1424-8220. Citado 2 vezes nas páginas 76 e 80.

131 WANG, D. et al. Targeted online password guessing: An underestimated threat. In: *2016 ACM SIGSAC Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2016. (CCS '16), p. 1242–1254. Citado na página 77.

132 ZHANG, P.; LIN, C. Security threats in network coding. In: _____. *Security in Network Coding*. Cham: Springer International Publishing, 2016. p. 9–19. ISBN 978-3-319-31083-1. Citado na página 80.

133 MITCHELL, J. C.; MITCHELL, M.; STERN, U. Automated analysis of cryptographic protocols using Murphi;. In: *1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097)*. [S.l.: s.n.], 1997. p. 141–151. Citado na página 82.

134 SHMATIKOV, V.; STERN, U. Efficient finite-state analysis for large security protocols. In: *11th IEEE Computer Security Foundations Workshop*. [S.l.: s.n.], 1998. p. 106–115. Citado 3 vezes nas páginas 82, 84 e 85.

135 MITCHELL, J. C.; SHMATIKOV, V.; STERN, U. Finite-state analysis of SSL 3.0. In: *7th Conference on USENIX Security Symposium - Volume 7*. Berkeley, CA, USA: USENIX Association, 1998. (SSYM'98), p. 16–16. Citado 2 vezes nas páginas 82 e 84.

136 SHMATIKOV, V.; MITCHELL, J. C. Finite-state analysis of two contract signing protocols. *Theor. Comput. Sci.*, v. 283, n. 2, p. 419–450, jun. 2002. Citado 3 vezes nas páginas 82, 84 e 85.

137 SIPSER, M. *Introduction to the Theory of Computation 2nd (second) edition*. [S.l.]: 2nd Edition, 2005. Citado na página 82.