UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

SÉRGIO DE MEDEIROS CÂMARA

AUTENTICAÇÃO DE EVIDÊNCIAS DIGITAIS PARA SISTEMAS CIBERFÍSICOS DE RECURSOS LIMITADOS

Rio de Janeiro
2017

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

SÉRGIO DE MEDEIROS CÂMARA

AUTENTICAÇÃO DE EVIDÊNCIAS DIGITAIS PARA SISTEMAS CIBERFÍSICOS DE RECURSOS LIMITADOS

Tese de Doutorado submetida ao Programa de Pós-graduação em Informática do Instituto de Matemática e do Instituo Tércio Pacitti da Universidade Federal do Rio de Janeiro - UFRJ, como parte dos requisitos necessários à obtenção do título de Doutor em Informática.

Orientador: Luiz Fernando Rust da Costa Carmo

Rio de Janeiro
2017

CBIB Câmara, Sérgio de Medeiros

Autenticação de Evidências Digitais para Sistemas Ciberfísicos de Recursos Limitados / Sérgio de Medeiros Câmara. – 2017.
147 f.: il.

Tese (Doutorado em Informática) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática, Rio de Janeiro, 2017.

Orientador: Luiz Fernando Rust da Costa Carmo.
.

1. Evidência Digital. 2. Sistemas Ciberfísicos. 3. Recursos limitados. 4. Autenticação. 5. Computação Forense. 6. Redes Elétricas Inteligentes. – Teses. I. Carmo, Luiz Fernando Rust da Costa (Orient.). II. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática. III. Título

CDD

SÉRGIO DE MEDEIROS CÂMARA

Autenticação de Evidências Digitais para Sistemas Ciberfísicos de Recursos Limitados

Tese de Doutorado submetida ao Programa de Pós-graduação em Informática do Instituto de Matemática e do Instituo Tércio Pacitti da Universidade Federal do Rio de Janeiro - UFRJ, como parte dos requisitos necessários à obtenção do título de Doutor em Informática.

Aprovado em: Rio de Janeiro, 22 de Março de 2017.

Prof. Luiz Fernando Rust da Costa Carmo (Orientador)
Docteur, UPS, França
PPGI/UFRJ

Profa. Luci Pirmez
D.Sc., COPPE/UFRJ, Brasil
PPGI/UFRJ

Prof. Raphael Carlos Santos Machado
D.Sc., COPPE/UFRJ, Brasil
INMETRO

Prof. José Ferreira de Rezende
Docteur, Paris VI, França
COPPE/UFRJ

Prof. Artur Ziviani
Docteur, Paris VI, França
LNCC

Dedico à minha esposa e melhor amiga, Sayonara,
e à nossa amada filha, Laís.

AGRADECIMENTOS

Ao meu estimado orientador Luiz Fernando Rust, por todo seu conhecimento e confiança depositados não somente durante o período de doutorado, mas também por toda a parceria acadêmica, iniciada em 2008.

Aos competentes professores integrantes da Banca de Defesa, Luci Pirmez, Raphael Machado, José Ferreira de Rezende e Artur Ziviani, pela disponibilidade e suas importantes contribuições e correções feitas ao trabalho.

À Universidade Federal do Rio de Janeiro, ao Núcleo de Computação Eletrônica (NCE/UFRJ) e aos colegas e profissionais do Programa de Pós-graduação em Informática (PPGI/UFRJ).

Ao Instituto Nacional de Metrologia, Qualidade e Tecnologia (Inmetro) e aos colegas de trabalho da Divisão de Metrologia em Tecnologias da Informação e Telecomunicações (Dmtic).

Ao Programa Nacional de Apoio ao Desenvolvimento da Metrologia, Qualidade e Tecnologia (Pronametro), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação Parque Tecnológico da Paraíba, pelos recursos financeiros concedidos durante o período de doutorado.

Ao *National Institute of Standards and Technology* (NIST) e aos colegas da *Computer Security Division*. Aos amigos, Victoria Pillitteri, por todo o suporte e dedicação ao meu projeto de pesquisa, e Dhananjay Anand, pelas horas de boas discussões e companheirismo.

Aos meus familiares e amigos próximos, por entenderem as minhas faltas e pelos ótimos momentos juntos de sempre.

Às mulheres da minha vida, Sayonara e Laís, por compartilharem um lar e todo o amor de vocês comigo. Por serem a força que eu preciso para seguir em frente e querer ser cada vez melhor.

Aos meus queridos pais, Ronaldo e Iracema, por estarem sempre lá por mim em todos os momentos, sem hesitar, me ensinando e me educando incansavelmente. Este trabalho é consequência direta de seus esforços em me proporcionar o que, segundo eles, ninguém nunca poderá me tirar: o conhecimento.

A todos vocês, os meus sinceros agradecimentos.

"Por trás de cada problema, há uma oportunidade."

– Galileo Galilei

RESUMO

Câmara, Sérgio de Medeiros. **Autenticação de Evidências Digitais para Sistemas Ciberfísicos de Recursos Limitados**. 2017. 132 f. Tese (Doutorado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2017.

Os Sistemas Ciberfísicos estão presentes em um diverso número de aplicações modernas, p. ex., gerenciamento de energia elétrica, recursos hídricos, monitoramento de tráfego e de assistência à vida. Eles são definidos como os sistemas que oferecem integrações de computação, redes e processos físicos, sendo compostos por vários tipos de dispositivos conectados em rede. De maneira geral, é esperado que alguns componentes de um sistema ciberfísico tenham recursos computacionais, de memória e de rede limitados para executar tarefas específicas de monitoramento e controle dos processos físicos. Dentre as aplicações possíveis, as que demandam resposta em tempo real e, também, as que envolvem risco à vida tornam-se um desafio de implementação, uma vez que as restrições computacionais e de rede entram em conflito com os requisitos de segurança necessários e a qualidade do serviço desejada. Dessa forma, há uma grande necessidade de que esses sistemas operem de forma segura e confiável, mitigando o impacto de falhas e ataques que possam ocorrer e, também, investigando e conhecendo os fatores que ocasionaram eventos indesejados no passado.

A Evidência Digital cumpre um papel principal durante esse processo de investigação da causa e consequências de um evento de falha ou um incidente de segurança. As ações decorrentes de uma investigação forense são importantes para minimizar as consequências negativas, assim como também garantir a segurança de um sistema diante de futuras ameaças. Logo, se faz necessária a utilização de métodos de segurança da informação, a fim de garantir a integridade e a comprovação de origem das evidências coletadas. No entanto, a complexidade existente nos cenários dos Sistemas Ciberfísicos impõe requisitos não-funcionais restritivos aos métodos utilizados, tais como escalabilidade, desempenho computacional, utilização do armazenamento de dados e da rede de comunicação, entre outros.

A presente tese tem como objetivo propor e avaliar métodos de segurança da informação para a autenticação e validação de evidências digitais durante as etapas de armazenamento e transmissão em Sistemas Ciberfísicos que possuem dispositivos de recursos limitados. Apresentamos três propostas em relação à autenticação de

evidências digitais, cada uma com foco específico em uma parte do sistema, são elas: (*i*) um esquema de estruturação de log seguro, (*ii*) um protocolo de autenticação atrasada para uma comunicação *multicast*, e (*iii*) uma *tag* de autenticação baseada em características de dados de monitoramento.

Nossos métodos são avaliados de forma comparativa com outros métodos existentes na literatura em relação à redução do custo computacional, armazenamento interno e sobrecarga na rede de comunicação. Além disso, avaliamos esses métodos, também, para aplicações que apresentam limitações de recursos no domínio dos Sistemas Ciberfísicos. Em particular, escolhemos as Redes Elétricas Inteligentes como o cenário de avaliação principal.

Palavras-chave: Evidência Digital, Sistemas Ciberfísicos, Recursos limitados, Autenticação, Computação Forense, Redes Elétricas Inteligentes.

ABSTRACT

Câmara, Sérgio de Medeiros. **Autenticação de Evidências Digitais para Sistemas Ciberfísicos de Recursos Limitados**. 2017. 132 f. Tese (Doutorado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2017.

Cyber-Physical Systems (CPSs) are found in many modern applications, e.g., power management, water resources, traffic monitoring and life support. CPSs are defined as the systems that provide integrations of computation, networks and physical processes, composed of several types of networked devices. In general, some CPS components are expected to have limited computational, storage, and network resources to perform specific tasks for monitoring and controlling the physical processes. From the possible applications, those requiring real-time response and also life-critical procedures become an implementation challenge, since the computational and network constraints conflict directly with the necessary security requirements and quality of service. So, there is a great need for these systems to operate safely and reliably, by mitigating the consequences of failures and attacks that may occur and, also, by investigating and being aware of the factors that have caused undesired events in the past.

Digital Evidence plays a major role during the investigation process of the cause and consequences of a failure event or a security incident. Resulting actions from forensic investigation are important to minimize the negative consequences, as well as ensuring the security of a system against future threats. Therefore, it is necessary to use information security methods in order to guarantee the integrity and proof of origin of the collected evidences. However, the complexity of the cyber-physical systems scenarios imposes on the used methods some restrictive non-functional requirements, such as scalability, computational performance, data storage and communication network usage, among others.

This thesis aims to propose and evaluate information security methods on the authentication and validation of digital evidence for storage and transmission in the Cyber-Physical Systems that comprise resource-constrained devices. We present three proposals, each one for a specific part of the system, for digital evidence authentication, namely: (*i*) a secure log structure scheme, (*ii*) a delayed authentication protocol for multicast communication, and (*iii*) a characteristics-based authentication tag for monitoring data.

We comparatively evaluate our methods to the ones found in the literature, regarding the reduction of computational cost, internal storage and communication network overhead. In addition, we also evaluate these methods for constrained-resource applications in the Cyber-Physical Systems domain. Particularly, we chose the Smart Grids as our main evaluation scenario.

Keywords: Digital Evidence, Cyber-Physical Systems, Resource-constrained, Authentication, Digital Forensics, Smart Grids.

LISTA DE FIGURAS

2.1 Componentes e camadas de uma arquitetura WAMS. 19
2.2 Modelo Gilbert-Elliot. 21

3.1 Representação dos elementos do esquema SbC-EC MAC. 39
3.2 Concatenação de uma nova entrada $L_{i,m}$ não compactada. 42
3.3 Cenário de simulação. 48
3.4 Redução de tamanho de cada arquivo de log. 52

4.1 Modelo do Sistema. 63
4.2 Uma ilustração do funcionamento no tempo das cadeias de chaves usadas no protocolo *inf*-TESLA. 71
4.3 Os dois modos de divulgação de chaves: *2-chaves* e *Alternado*. . 75
4.4 Exemplo de um ataque homem no meio em um esquema com uma cadeia de chaves (sem compromisso autenticado) e no esquema de autenticação do *inf*-TESLA. 77
4.5 Custo computacional para TESLA e *inf*-TESLA durante 2 dias de comunicação. 82
4.6 Custo computacional para TESLA e *inf*-TESLA, em dois dias de comunicação, ao variar o número de chaves, n, armazenadas pelo emissor. 83

5.1 Modelo do Sistema. 91
5.2 Modos de Operação da *Tag* de Autenticação: (a) *Sobreposto*, (b) *Continuado*, e (c) *Intervalado*. 100
5.3 Campos de dados da *Tag* de Autenticação. 102
5.4 Exemplo de um ataque do tipo *Degrau*. 112
5.5 Exemplo de um ataque do tipo *Alternado*. 114

LISTA DE TABELAS

3.1 Sobrecarga de autenticação dos esquemas de log seguro 46
3.2 Resultado da simulação em 1.000 rodadas 47
3.3 IDs/Categorias de log [20] 50
3.4 Resultados do log de dados total 51
3.5 Resultados do log de dados por arquivo 51

4.1 Cálculo da sobrecarga na comunicação. 80
4.2 Cálculo do custo computacional. 81

5.1 Simulação de perdas na rede (1000 rodadas/*fingerprints*). 111
5.2 Simulação de ataque de injeção de dados do tipo *Degrau* (1.000 rodadas/*fingerprints*). 112
5.3 Simulação de ataque de injeção de dados do tipo *Alternado* (1.000 rodadas/*fingerprints*). 113

LISTA DE ABREVIATURAS E SIGLAS

CPS	Cyber-Physical System
DER	Distinguished Encoding Rules
ECDSA	Elliptic Curve Digital Signature Algorithm
IED	Intelligent Electronic Device
MAC	Message Authentication Code
MTU	Maximum Transmition Unit
PDC	Phasor Data Concentrator
PLC	Programmable Logic Controller
PMU	Phasor Measurement Unit
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SHA	Secure Hash Algorithm
UDP	User Datagram Protocol
WAMS	Wide Area Monitoring System

SUMÁRIO

1 INTRODUÇÃO 2
1.1 Contextualização 2
1.2 Motivação 6
1.3 Objetivos e Contribuições 8
1.3.1 Autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques 10
1.3.2 Autenticação de evidências digitais em trânsito 11
1.3.3 Validação externa de evidências digitais 12
1.4 Organização do Trabalho 13

2 CONCEITOS BÁSICOS 15
2.1 Evidência Digital 15
2.1.1 Autenticação de Evidência Digital 16
2.2 Sistemas de Monitoramento de Grandes Áreas 17
2.2.1 Arquitetura WAMS 18
2.3 Métodos utilizados 20
2.3.1 Filtro de Média Móvel 20
2.3.2 Modelo Gilbert-Elliot 21
2.3.3 Coeficiente de Correlação de Pearson 22

3 ESQUEMA DE ESTRUTURAÇÃO SBC-EC PARA LOG SEGURO 24
3.1 Contextualização do Problema 25
3.2 Trabalhos Relacionados 28
3.3 Modelos 31
3.3.1 Modelo do Sistema 32
3.3.2 Modelo de Atacante 34
3.4 Esquema de Estruturação SbC-EC para Log Seguro 35
3.5 Esquema de Log Seguro SbC-EC MAC 37
3.5.1 Inicialização do log de dados 40
3.5.2 Concatenação e compactação de entradas de log 40
3.5.3 Verificação do log de dados 42
3.6 Discussão de Segurança 43
3.7 Avaliação 45
3.7.1 Análise Comparativa 45
3.7.2 Resultados Experimentais 48
3.8 Conclusão 53

4 PROTOCOLO DE AUTENTICAÇÃO ATRASADA *INF*-TESLA PARA COMUNICAÇÃO *MULTICAST* 55
4.1 Contextualização do Problema 56
4.2 Trabalhos Relacionados 59
4.3 Características do Cenário 62
4.3.1 Modelo do Sistema 63
4.3.2 Latência de Comunicação 64
4.3.3 Considerações de Segurança 66
4.4 Protocolo Proposto 67
4.4.1 Protocolo TESLA 67
4.4.2 Protocolo *inf*-TESLA 70
4.4.3 Configuração Inicial da Comunicação 72
4.4.4 Mecanismo *Dual Offset Key Chains* 73
4.4.5 Divulgação de Chaves 74
4.4.6 Segurança do *inf*-TESLA 75
4.5 Avaliação 79
4.6 Conclusão 84

5 AUTENTICAÇÃO BASEADA EM CARACTERíSTICAS PARA DADOS DE MONITORAMENTO 86
5.1 Definição do Problema 87
5.2 Modelos 90
5.2.1 Modelo do Sistema 90
5.2.2 Modelo de Atacante 92
5.3 Trabalhos Relacionados 93
5.4 *Tag* de Autenticação baseada em Características 97
5.4.1 Modos de operação 99
5.4.2 Campos de dados da *Tag* de Autenticação 101
5.4.3 Geração do *fingerprint* 104
5.4.4 Verificação da *Tag* 105
5.4.5 Discussão de segurança 106
5.5 Avaliação 108
5.6 Conclusão 115

6 CONCLUSÃO 117

REFERÊNCIAS 119

1 INTRODUÇÃO

1.1 Contextualização

O contínuo progresso na área da eletrônica vem contribuindo, cada vez mais, para a introdução de um grande número de novos dispositivos tecnológicos no mundo. A sociedade se tornou dependente das inúmeras tecnologias adotadas e, consequentemente, estão presentes por todos os lugares que frequentamos, tanto nos locais de trabalho, na vida cotidiana ou durante o lazer. Atualmente, os desenvolvedores têm pronto acesso a plataformas de hardware, como o *Raspberry Pi* e o *Arduino*, juntamente a diversos outros sensores e atuadores disponíveis, o que facilita a implementação de novas aplicações. Os potenciais domínios de aplicação desses dispositivos incluem desde simples soluções (p.ex., acessórios tecnológicos e automação da casa) para consumo no varejo e para empresas de fabricação e construção, até sistemas mais complexos como os de gestão de edifícios, logística de transporte, controle de serviços públicos, medicina, monitoramento de saúde e pesquisa científica [84]. A prática de coletar, armazenar e compartilhar informações digitais, assim como tomar decisões e atuar a partir delas, logo tornou-se algo difundido e comum.

Um grande número de sistemas computacionais atuais apresentam uma combinação de sistemas físicos e sistemas virtuais, que são desenvolvidos independentemente. No entanto, a tendência no aumento das exigências de desempenho e de complexos padrões de utilização mudaram, de forma significativa, a dinâmica e as interações entre componentes físicos e os componentes virtuais. Fundamentalmente, essas mudanças exigem novas abordagens de projeto, onde ambas as categorias de componentes estejam integradas em todos os níveis [51, 35]. Essa necessidade tem

conduzido os avanços na área dos Sistemas Ciberfísicos – do inglês, *Cyber-Physical Systems*, ou CPSs.

Os Sistemas Ciberfísicos são definidos como os sistemas que oferecem integrações de computação, redes e processos físicos [22, 90], ou, em outras palavras, como os sistemas onde os componentes físicos e virtuais estão de todas as formas interligados, cada um operando em diferentes escalas espaciais e temporais, exibindo múltiplas e distintas modalidades de comportamento, e interagindo uns com os outros em uma enorme quantidade de formas que mudam com o contexto [68].

Os CPSs possuem aplicações nas mais diversas áreas, incluindo sistemas e dispositivos médicos de alta confiança e de assistência à vida, controle e segurança de tráfego, sistemas automotivos avançados, controle de processo, conservação de energia, controle ambiental, aviação, instrumentação, robótica distribuída (telepresença, telemedicina), sistemas de defesa, manufaturas, estruturas inteligentes e controle de infraestruturas críticas (p.ex, redes de energia elétrica, recursos hídricos, e sistemas de comunicação) [55]. Em geral, é esperado que os CPSs estejam disponíveis de forma ininterrupta, sempre proporcionando um comportamento adequado e mantendo sua operação mesmo sob condições de estresse, incluindo desastres naturais e condições climáticas severas, como nos casos das redes de energia.

Devido à sua complexidade e à sua utilização em aplicações críticas, os CPSs demandam grande necessidade de garantir confiabilidade, resiliência e segurança às suas operações. A partir dessas necessidades, uma série de aspectos relacionados à segurança da informação devem ser levados em consideração. Aspectos como a prevenção e a detecção de intrusão, privacidade, anonimato, integridade e autenticidade dos dados, entre outras [51, 5], podem ser especialmente desafiadores dentro do contexto desses sistemas.

De acordo com Byres e Lowe [13], os primeiros ataques aos sistemas de controle industrial e aos componentes de uma infraestrutura crítica datam do ano de 1988. O ciberataque mais famoso, conhecido atualmente, foi realizado pelo *worm Stuxnet* [32], que foi descoberto em 2010 e, supostamente, operado por mais de três anos sem ser detectado. O aspecto mais notável do *Stuxnet* é ter infligido danos físicos a uma infraestrutura industrial – nesse caso, centrífugas de enriquecimento de urânio – através de uma intrusão virtual. O ataque em março de 2000 contra o *Marochy Water Services* em Queensland, Austrália, é outro exemplo proeminente de um ataque a uma infraestrutura industrial. A intrusão foi realizada através de comandos de rádio ao sistema SCADA (*Supervisory Control And Data Acquisition*) que interromperam as operações de bombeamento e desligaram os alarmes, resultando no vazamento de esgoto sem tratamento nos rios e parques locais [91]. Pesquisadores demonstraram que ciberataques elaborados podem levar a consequências físicas, incluindo a desativação dos freios de um automóvel, o desligamento do motor enquanto o carro está se movendo em alta velocidade, o bloqueio permanentemente das portas e a manipulação do indicador de velocidade [18, 52]. Outro exemplo refere-se ao comprometimento de uma rede interveicular, onde acidentes podem ocorrer, por exemplo, em decorrência da transmissão de uma informação errada de distância entre carros.

Um atacante poderá agir em diversas frentes para explorar vulnerabilidades em diferentes componentes do sistema, as quais, individualmente, poderiam não representar uma séria ameaça, mas que, quando combinadas, podem servir a um propósito maior. Isso, aliado às interdependências existentes dentro de um sistema ciberfísico e, ainda, entre múltiplos sistemas ciberfísicos, os ataques poderão ultrapassar os limites de um domínio, se propagando e amplificando os efeitos, causando danos e resultados catastróficos em diversas outras áreas [107].

Diante desse cenário suscetível a ataques, é fundamental a implementação

de mecanismos de segurança da informação e auditoria a fim de combater as vulnerabilidades introduzidas pelos componentes que integram a tecnologia subjacente aos CPSs. Além de *firewalls* de rede, ferramentas de monitoramento e sistemas de detecção de intrusão, existe a necessidade de que sejam realizadas, a tempo, uma análise forense e uma resposta a incidentes quando esses mecanismos de prevenção falham. Nesse outro lado da questão – o pós ataque –, a habilidade de responder a incidentes de segurança, juntamente com a habilidade de analisar e aprender com o que aconteceu, é crucial.

A Computação Forense é uma área de relevante importância e engloba atributos como a identificação do problema e a descoberta da causa raiz de um comprometimento do sistema, além de teorias e técnicas científicas para a análise e processamento de evidências digitais [16]. Através da análise das evidências digitais, pode-se revelar as ações que aconteceram durante um incidente de segurança, juntamente com as motivações do fato e, até mesmo, a identidade do atacante. As ações decorrentes de uma análise forense são importantes para mitigar as consequências negativas, assim como também garantir a segurança do sistema diante de futuras ameaças. Os especialistas em segurança têm como papel realizar auditorias e resolver o problema, indicando os caminhos para uma melhoria na resiliência do sistema.

A evidência digital pode estar sujeita a todo um processo forense, o qual engloba os seguintes estágios: identificação, coleta, examinação, análise, documentação e apresentação [85]. A preservação da evidência digital é um princípio essencial que deve ser considerado em todos os estágios desse processo. Para essa finalidade, a utilização de métodos de segurança da informação cumpre um papel importante, a fim de garantir a integridade e a comprovação de origem das evidências coletadas. Por outro lado, a complexidade existente nos cenários de operação dos Sistemas Ciberfísicos impõe, às soluções de segurança e aos métodos utilizados, requisitos

não-funcionais restritivos referentes à escalabilidade, desempenho computacional, utilização da rede de comunicação, entre outros.

O presente trabalho tem como objetivo propor e avaliar métodos de autenticação e validação de evidências digitais para os Sistemas Ciberfísicos que possuam limitações quanto aos recursos computacionais, de armazenamento interno e de rede de comunicação. Apresentamos três contribuições para a autenticação de evidências digitais em (*i*) um esquema de estruturação de log seguro, (*ii*) um protocolo de autenticação atrasada para uma comunicação *multicast*, e (*iii*) uma *tag* de autenticação baseada em características de dados de monitoramento.

1.2 Motivação

Um dos princípios essenciais relacionado à Evidência Digital está na sua preservação e resguardo da sua integridade. A evidência digital, por sua própria natureza, é frágil e pode ser alterada, danificada ou destruída por manuseio ou análise inadequados [25]. Algumas precauções especiais devem ser tomadas para preservar esse tipo de dado. É recomendado, portanto, que o procedimento completo de coleta e análise de evidências digitais seja realizado em conformidade com as regulamentações e boas práticas da Computação Forense existentes, tais como a *ACPO Good Practice Guide for Computer-Based Electronic Evidence* [7], de modo que o material encontrado possa ser validado e, por fim, seja utilizado para levantar acusações legais contra os autores identificados de algum crime [95]. Para que isso possa ser realizado, torna-se importante o uso de mecanismos de criptografia (p.ex., algoritmos de *hash* e de autenticação de dados), protocolos seguros de comunicação (p.ex., IPSec, VPN, SSH) e práticas de segurança recomendadas (p.ex., armazenamento seguro em servidores *offline* e em mídias apenas para leitura).

Outros mecanismos de segurança, tais como de monitoramento, auditoria e registro de eventos dos sistemas, são, também, grandes aliados durante a condução de uma investigação forense. A ausência desses mecanismos resulta em uma grande retrabalho por parte do investigador, uma vez que os eventos que precederam um incidente de segurança precisam ser descobertos e correlacionados no tempo de cada acontecimento. Além disso, há relatos de que sistemas que possuem mecanismos de auditoria podem, por vezes, ser implementados com esses recursos desativados [30], simplesmente por serem vistos, por alguns administradores de sistemas, como não necessários ou de baixo custo-benefício. Isso se deve, por muitas vezes, ao impacto negativo que uma funcionalidade a mais pode causar no desempenho de um dispositivo, o que pode interferir nas operações fins do mesmo. Essa preocupação acontece não somente em relação ao desempenho do dispositivo, mas também em outras questões, como a ocupação da memória de armazenamento, utilizada na persistência dos dados de auditoria, e, também, na sobrecarga da rede de comunicação, utilizada na transmissão das evidências. Dado esse cenário, é crucial que os mecanismos de segurança sejam concebidos e implementados de forma a impactarem minimamente o sistema o qual estão apoiando.

De maneira geral, o equilíbrio entre o desempenho e a segurança das informações é um desafio constante no desenvolvimento de dispositivos de recursos limitados e, ainda maior, no âmbito dos Sistemas Ciberfísicos. Fatores, como a utilização de mecanismos de segurança que impactem minimamente o desempenho do sistema como um todo e, também, a escalabilidade da solução de segurança, se fazem obrigatórios diante da complexidade e dimensão dos sistemas em questão [28].

1.3 Objetivos e Contribuições

Ao longo do trabalho, apresentamos métodos de segurança da informação para a autenticação e validação de evidências digitais durante as etapas de armazenamento e transmissão em sistemas de recursos limitados. Nossos métodos são avaliados de duas formas: (*i*) através de comparações com métodos existentes na literatura, e (*ii*) em aplicações de recursos limitados, em relação ao armazenamento interno, desempenho computacional e à rede de transmissão, no domínio dos Sistemas Ciberfísicos. Em particular, escolhemos as Redes Elétricas Inteligentes como o cenário de avaliação principal.

De uma maneira geral, este trabalho pode ser dividido em três propostas de métodos de segurança, cada uma abrangendo um problema específico de uma parte do sistema. Abaixo, enumeramos essas propostas e suas respectivas hipóteses de pesquisa.

1. Autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques.

   Aqui, tratamos sobre o armazenamento seguro de log de dados em dispositivos desprotegidos e com restrições de armazenamento e de comunicação. Para isso, apresentamos o esquema de estruturação para log seguro SbC-EC [14].

   Hipotése 1. Uma estruturação de log seguro que considera categorização e compactação das entradas de log permite que haja uma redução na sobrecarga de segurança, no armazenamento e na comunicação na rede em relação aos outros esquemas de log seguro existentes.

2. Autenticação de evidências digitais em trânsito.

   É abordado o problema da autenticação fim-a-fim de dados de monitoramento

em uma comunicação *multicast*. Apresentamos um protocolo de autenticação atrasada para comunicação *multicast*, o *inf*-TESLA [15]. Essa proposta é validada para uma aplicação de monitoramento nas Redes Elétricas Inteligentes.

Hipotése 2. Um protocolo de autenticação atrasada para comunicações *multicast* que utiliza duas cadeias de chaves simultâneas para a autenticação dos dados permite a redução da sobrecarga computacional e de segurança em comunicações de longo prazo com altas taxas de envio de pacote.

3. Validação externa de evidências digitais.

   Aqui, tratamos sobre a validação de um conjunto incompleto de evidências por um verificador externo ao sistema. Apresentamos um esquema de autenticação baseada em características para dados de monitoramento. Essa proposta é, também, validada para uma aplicação de monitoramento nas Redes Elétricas Inteligentes.

   Hipotése 3. Um esquema de uma *tag* de autenticação baseada em características de um conjunto de dados de monitoramento permite tolerância à perdas de pacotes, redução da sobrecarga de segurança e validação estatística desse conjunto por um verificador externo ao sistema.

Dessa forma, atacamos desafios de autenticação de evidências digitais em diferentes partes de um sistema, incluindo os dispositivos em campo, a rede de comunicação, os servidores de armazenamento de evidências, e, também, um verificador externo. Ressaltamos que, o contexto relacionado à cadeia de custódia após a identificação de uma evidência e, também, relacionado ao processo judicial pelo qual uma evidência digital está sujeita a participar, encontram-se fora do escopo das propostas desta tese. Estamos, principalmente, interessados em propor métodos de segurança da informação a fim de garantir a verificação da integridade e da origem desses dados, os quais podem, eventualmente, dar suporte às etapas judiciais posteriores.

Nossas propostas e suas contribuições são melhor detalhadas nas três subseções seguintes.

1.3.1 Autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques

A proteção torna-se necessária uma vez que alguns tipos de dispositivos precisam operar em ambientes inseguros ou hostis e, portanto, suscetíveis a ataques (p.ex., medidores inteligentes, PLCs, RTUs, etc.). Após o comprometimento de um sistema, atacantes experientes tentarão instalar *backdoors* – para uma futura invasão – e, também, encobrir suas ações através da manipulação maliciosa dos registros de log. Portanto, garantir que os registros de log estejam íntegros e detectar alguma possível manipulação desses (p.ex., deleção, inserção ou modificação), é de grande importância durante uma investigação forense.

Para essa finalidade, apresentamos o esquema de estruturação SbC-EC para log seguro, apropriado para dispositivos com restrições de armazenamento e de comunicação em rede, incluindo duas propriedades: Separação por Categoria e Compactação de Entradas. Descrevemos também o esquema de log seguro SbC-EC MAC, que implementa a nova estruturação proposta em conjunto com primitivas da criptografia simétrica e o esquema de autenticação *FssAgg* [58], a fim de oferecer a proteção adequada aos registros de log. O SbC-EC MAC apresenta um ganho de armazenamento em relação a outros esquemas simétricos de log seguro da literatura, o que é demonstrado na parte de avaliação.

1.3.2 Autenticação de evidências digitais em trânsito

A área de computação forense e a área de resposta a incidentes são diretamente relacionadas. O time de resposta a incidentes deve ser capaz de coletar evidências digitais em tempo hábil. Na maioria dos ambientes dos sistemas de controle modernos, existem tecnologias capazes de atender a requisitos mínimos referentes à coleta desses dados. Contudo, determinados métodos de coleta, tais como os baseados em monitoramento e detecção através das redes de comunicação, introduzem um custo que pode ser bastante alto dependendo da natureza, tamanho e criticidade da rede dos sistemas de controle [30].

Iremos tratar esse desafio no domínio das Redes Elétricas Inteligentes e, mais especificamente, para aplicações WAMS, sigla em inglês para *Wide-Area Monitoring System*. Uma aplicação WAMS tem como objetivo "monitorar a saúde" da rede elétrica, obtendo dados em tempo real de diversos dispositivos espalhados por diferentes pontos da rede. Os *Phasor Measurements Units* (PMUs) são os dispositivos que desempenham um papel significativo na análise forense em relação às perturbações da rede elétrica, pois podem ajudar na visualização em tempo real de uma pré ou pós-perturbação, e a indicação de uma possível falha. Além disso, a análise forense de dados de PMUs pode revelar a localização de nós comprometidos devido a um ataque.

Os dados de medições são comunicados, por cada PMU, em *multicast* para as aplicações de monitoramento. Contudo, a autenticação desses dados está sujeita aos diversos requisitos de projeto dos sistemas de monitoramento das redes elétricas, tais como baixo *overhead* de segurança, tolerância a perdas na rede, criticidade no tempo das operações e alta taxa de transmissão de dados. Para isso, propomos o *inf*-TESLA, *Infinite Timed Efficient Stream Loss-tolerant Authentication*, um protocolo de autenticação atrasada para comunicações *multicast*. Nossa abordagem é

baseada no protocolo de autenticação TESLA [77], mas é adaptada para dar suporte às altas taxas de transmissão de dados por um longo, ou até ilimitado, período de tempo. Fundamentalmente, o nosso protocolo utiliza o mecanismo de geração e gerenciamento de chaves *Dual Offset Key Chains* para reduzir o atraso da autenticação dos dados e o custo computacional associado ao gerenciamento das chaves criptográficas.

Apresentamos uma descrição de dois modos diferentes de divulgação de chaves, e demonstramos a segurança do protocolo contra uma tentativa de ataque homem no meio. Além disso, comparamos a nossa abordagem em relação ao protocolo TESLA, o que resultou em uma redução no custo computacional, para o emissor e o receptor durante o período de operação simulado.

1.3.3 Validação externa de evidências digitais

Esta última parte do trabalho refere-se à validação, em relação à autenticidade, de um conjunto incompleto de evidências digitais coletadas e armazenadas após um período de tempo indeterminado. Essa validação poderá ser realizada por um verificador externo ao sistema, como, p. ex., um investigador forense. É desejável, portanto, que o emissor utilize esquemas de não-repúdio (p. ex., assinatura digital), a fim de garantir a validação externa/pública de seus dados, porém, impactando minimamente o desempenho do sistema como um todo.

Os servidores de armazenamento de dados ("*historians*") são responsáveis por armazenar dados de monitoramento e informações históricas sobre processos industriais. No entanto, seja pelas restrições inerentes ao *link* de comunicação (p. ex., linha telefônica, rádio, cabo, etc.), ou pelos requisitos da aplicação (p. ex, uso do protocolo de transporte UDP), um percentual dos dados transmitidos são perdidos

no caminho até os servidores de aplicação e armazenamento.

Devido à perda de pacotes na rede, o conjunto de dados gerados não é, necessariamente, o mesmo conjunto de dados que o servidor de aplicação recebeu e armazenou. Dessa forma, uma assinatura de um *hash* acumulado ao longo de um intervalo de tempo não tem garantias de funcionar nesse cenário, uma vez que é necessário o conjunto completo de dados para reconstrução do *hash* e validação da assinatura.

Como solução para esse problema, descrevemos um esquema de uma *tag* de autenticação baseado em características para dados de monitoramento. Essa *tag* oferece não-repúdio e garantia de integridade, com um alto grau de correlação estatística, de um fluxo de dados transmitidos por um canal de comunicação suscetível a ataques e perdas de pacotes. A partir da análise dessa *tag*, um verificador externo é capaz de validar um conjunto de evidências guardadas no servidor de armazenamento.

Na parte de avaliação, testamos o esquema para uma aplicação WAMS nas Redes Elétricas Inteligentes. Realizamos a simulação de três cenários para a aplicação em questão, uma levando em conta apenas a perda de pacotes na rede, e as outras duas considerando diferentes ataques de injeção de dados durante a transmissão. Como resultados, obtivemos um alto percentual de validação de evidências ao considerar apenas as perdas na rede, e um alto percentual de invalidação durante os cenários de ataque de injeção de dados.

1.4 Organização do Trabalho

O restante da tese está organizado da seguinte forma:

- No Capítulo 2, descrevemos os conceitos básicos necessários para o entendimento do trabalho.
- No Capítulo 3, detalhamos o esquema de estruturação SbC-EC e o esquema de log seguro SbC-EC MAC, e o avaliamos experimentalmente e comparativamente a outros trabalhos da literatura.
- No Capítulo 4, descrevemos detalhadamente o protocolo de autenticação atrasada inf-TESLA. Apresentamos o mecanismo de geração e gerenciamento de chaves *Dual Offset Key Chains* e a avaliação de desempenho da solução.
- No Capítulo 5, apresentamos o esquema de autenticação baseada em características para dados de monitoramento, e o avaliamos em cenários de ataques e perdas de pacote.
- No Capítulo 6, apresentamos nossas conclusões e considerações finais do trabalho.

2 CONCEITOS BÁSICOS

Neste capítulo, apresentamos os conceitos básicos para um melhor entendimento deste trabalho. Discorremos sobre o conceito de evidência digital, detalhamos o sistema de monitoramento de grandes áreas, e os demais métodos utilizados ao longo do texto.

2.1 Evidência Digital

Casey [16] define a evidência digital como qualquer dado armazenado ou transmitido usando um computador, que apóie ou refute uma teoria sobre como um delito ocorreu, ou que aborde elementos críticos do delito, como intenção ou álibi (adaptado de Chisum [19]).

Tendo em vista a ubiquidade dos computadores e dos dados digitais, podemos considerar que são diversas as fontes de evidências digitais. Henseler [41] categoriza essas fontes basicamente em três conjuntos: os sistemas de computadores, os sistemas de comunicação, e os sistemas embarcados.

Nos dias atuais, é raro o crime que não tenha um dado associado em algum desses sistemas. As evidências digitais podem ser úteis em uma ampla gama de investigações criminais incluindo homicídios, pessoas desaparecidas, abuso infantil, tráfico de drogas, crimes sexuais, fraude e roubo de informações pessoais. Esses registros podem ajudar a estabelecer quando determinados eventos ocorreram, para onde os suspeitos e as vítimas foram e com quem eles se comunicaram, e podem, inclusive, revelar a intenção dos suspeitos de cometer um crime [16].

2.1.1 Autenticação de Evidência Digital

Alguns textos relativos à computação forense afirmam que a autenticação é o processo de assegurar que a evidência recuperada é a mesma que os dados originalmente capturados, porém, esse conceito é mais sutil [16]. Do ponto de vista técnico, nem sempre é possível comparar o dado adquirido com o original. Como exemplo, o conteúdo da memória RAM de um computador em execução muda constantemente. Esse conteúdo refere-se ao estado do computador em execução naquele instante de tempo, e, portanto, não há nenhum original para ser comparado com a sua cópia. Da mesma forma, o tráfego de rede é transitório e deve ser capturado enquanto ele está em trânsito. Após o tráfego de rede ser capturado, somente as cópias permanecem e os dados originais não estarão mais disponíveis para comparação.

Do ponto de vista legal, para demonstrar que uma evidência digital é autêntica, geralmente é necessário convencer o tribunal que o dado foi adquirido de um computador e/ou local específico, e que uma cópia da evidência foi adquirida de forma precisa e completa.

Para autenticar uma evidência digital, também pode ser necessário avaliar a sua confiabilidade. Existem duas abordagens gerais para avaliar se a evidência digital pode ser utilizada em um tribunal. A primeira abordagem é descobrir se o computador que gerou a evidência estava funcionando de forma normal, e a outra abordagem é examinar a evidência digital a fim de encontrar possíveis adulterações e outros danos.

Para Reed [93], a autenticação significa convencer o tribunal que (*i*) o conteúdo da evidência se manteve inalterado, (*ii*) que as informações contidas na evidência são efetivamente originárias da sua suposta fonte, seja humano ou máquina, e (*iii*) que informações, como a data do registro, são exatas. O que ressalta a

importância de garantir a integridade e a origem desses dados.

2.2 Sistemas de Monitoramento de Grandes Áreas

Os Sistemas de Monitoramento de Grandes Áreas – ou do inglês, *Wide Area Monitoring Systems* (WAMS) – foram introduzidos a fim de facilitar o monitoramento dinâmico de uma área extensa de um sistema de energia, utilizando aplicações específicas para proteção do sistema. O termo "monitoramento" indica que se trata de um sistema de aquisição de dados e de uma infraestrutura de comunicação que direcionam os dados para um local central. Como o objetivo pretendido é a proteção do sistema, a coleta de dados para um local central deve ser rápida o suficiente para favorecer esse propósito, o que implica que as latências da transmissão devem ser mínimas. Tradicionalmente, os sistemas WAMS usam concentradores de dados e *links* de comunicação dedicados e rápidos para alcançar o desempenho necessário.

É reconhecido que os sistemas WAMS têm outras aplicações além de proteção e controle do sistema. Uma aplicação importante é a de monitoramento da estabilidade do sistema, outra é a de consciência situacional ("*situational awareness*"), a qual refere-se aos processos de percepção, decisão e ação que permitem a avaliação e reação a determinadas condições do sistema. De fato, um dos principais fatores para a modernização da rede é a melhoria das capacidades de consciência situacional, a fim de gerenciar o sistema de transmissão de energia. Dessa forma, os sistemas de aquisição de dados sincronizados por GPS são a tecnologia-chave para alcançar os objetivos de monitoramento de grandes áreas. Com a introdução das medições sincronizadas de GPS, a tecnologia das WAMS evoluiu de forma efetiva.

Em geral, as aplicações de monitoramento de grandes áreas se enquadram nas três categorias a seguir: (i) assistência na tomada de decisão do operador, isto

é, estimativas de estado baseadas na medição do fasor, (*ii*) análise *off-line* após eventos importantes do sistema, p. ex., análise *post-mortem*, e (*iii*) verificação de modelos e calibração de instrumentos.

2.2.1 Arquitetura WAMS

Em geral, o sistema de tecnologia da informação e comunicação que suporta o monitoramento de grandes áreas inclui quatro componentes básicos: o *Phasor Measurement Unit* (PMU), o *Phasor Data Concentrator* (PDC), o *software* das aplicações, e a rede de comunicação que interliga os três primeiros componentes como um sistema WAMS [112].

Um fasor é um número complexo que representa a magnitude e o ângulo de fase das ondas senoidais encontradas na eletricidade. O dispositivo PMU mede os fasores de corrente e os fasores de tensão a uma frequencia de 30, 60 ou 120 vezes por segundo. Essas medições de fasores são sincronizadas no tempo e, portanto, são chamadas de "sincrofasores". Os sincrofasores são definidos no documento *C37.118.1-2011 – IEEE Standard for Synchrophasor Measurements for Power Systems* [43].

Os PMUs são posicionados em vários locais do barramento para oferecer uma visão otimizada do sistema de energia elétrica, a fim de oferecer uma estimativa precisa do estado da rede. Eles transmitem os sincrofasores para o PDC, que, por sua vez, concentra essas medições e as retransmite para outro PDC, ou para outros pontos finais da rede, como p. ex., os sistemas de gerenciamento de energia, os servidores *historians* e outras aplicações que consomem sincrofasores (ver Figura 2.1).

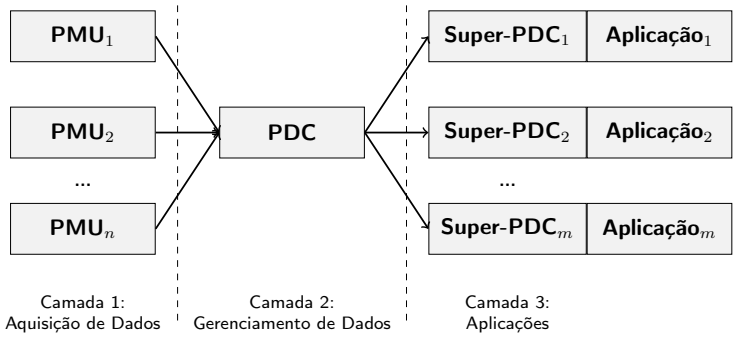

Figura 2.1: Componentes e camadas de uma arquitetura WAMS.

O processo de concentração de dados no PDC acontece a partir do recebimento de vários pacotes individuais de cada PMU e a criação de um único pacote com as múltiplas medições recebidas alinhadas no tempo. Como parte desse processo, o PDC também pode interpolar os dados que estiverem faltando e aplicar algoritmos para ajustar as taxas de envio (p. ex., ajustar um fluxo de 30 amostras por segundo para 60 amostras por segundo). O PDC também pode implementar aplicações de controle de área, aplicações de detecção de eventos no sistema elétrico, checagem de erro e arquivamento de medições para análise de dados históricos.

Atualmente, o PMU e o PDC usam o padrão de comunicação IEEE C37.118 [43] para a transmissão de dados, podendo ser futuramente substituído pelo protocolo IEC/TR 61850-90-5 [47]. Os PMUs são pré-configurados com informações de conexão de rede e detalhes de frequência de medição. Quando um PDC se conecta a um PMU, ele requisita as informações de configuração daquele PMU. Depois que o pacote de configuração é enviado, o PMU começar a transmitir um fluxo contínuo de pacotes de dados para o PDC. Os pacotes de configuração, de comandos, de cabeçalhos e de dados do IEEE C37.118 não incluem mecanismos de segurança de integridade ou de confidencialidade.

Os analistas e investigadores têm usado os dados de PMUs em análises fo-

renses a fim de entender as sequências e causas de falhas na rede elétrica [70]. Tais investigações usam os dados de PMUs e outras informações para criar um registro de "sequência de eventos", a fim de compreender, de que maneira e quando, os equipamentos foram afetados. A partir desse registro de sequência de eventos, os analistas conseguem entrar os principais parâmetros (p. ex., de geração e de carga, de configurações da rede e características operacionais) em um modelo dinâmico do sistema para ver se esse modelo pode reproduzir o incidente e explicar, com precisão, o que aconteceu.

2.3 Métodos utilizados

A seguir, explicamos três métodos utilizados no Capítulo 5.

2.3.1 Filtro de Média Móvel

O Filtro de Média Móvel é um simples filtro passa baixa comumente usado para suavizar um sinal/vetor de dados amostrados. O resultado do filtro é obtido calculando-se a média de um conjunto de valores a partir do sinal de entrada para produzir cada ponto no sinal de saída. O filtro é descrito pela seguinte equação [92]:

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j],$$

onde $x[]$ é o sinal de entrada, $y[]$ é o sinal de saída, e M é o número de pontos na média.

O filtro recebe M amostras de entrada de cada vez, calcula a média dessas

M amostras e produz um único ponto de saída. À medida que o valor do parâmetro M (ou *janela*) aumenta, a suavidade da saída também aumenta.

Como padrão, o índice i é incrementado, a cada iteração, de uma em uma unidade ao longo do vetor $x[]$. No entanto, um parâmetro *passo* pode ser estabelecido para funcionar como um multiplicador do índice i no vetor x para cada iteração do filtro, ou seja, $x[i * passo + j]$. Ao atribuir $passo > 1$, o parâmetro indica um fator de compressão, a fim de reduzir o número do pontos na saída do filtro.

2.3.2 Modelo Gilbert-Elliot

O modelo Gilbert-Elliot [36, 27] é um modelo simples de canal de comunicação. O modelo GE tem sido amplamente utilizado na literatura para caracterizar e representar o padrão de perda de pacotes em rajada, produzido por atenuações no sinal de uma comunicação sem fio. O modelo é baseado em uma cadeia de Markov de dois estados, conforme representado na Figura 2.2. Os dois estados representam os estados bom (S_0) e ruim (S_1) em termos de perdas de pacotes. O padrão de perda de pacotes gerado, devido à transição entre estados, é composto por uma sequência consecutiva de pacotes recebidos e uma sequência consecutiva de pacotes perdidos.

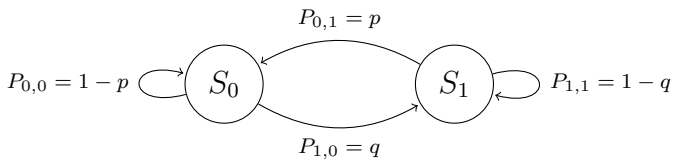

Figura 2.2: Modelo Gilbert-Elliot.

O modelo GE é determinado por dois parâmetros: a probabilidade de perda de pacotes, P_E, e o comprimento médio de rajada, L_B. Esses parâmetros são obtidos a partir da matriz de probabilidade de transição, P, do estado do canal. Essa matriz

é composta por probabilidades condicionais de acordo com a seguinte estrutura:

$$P_{i,j} = P[S(t) = S_j | S(t-1) = S_i]; \quad i, j \in \{0, 1\},$$

onde é representada a probabilidade de transição do estado S_i, no instante $t-1$, para o estado S_j, no instante t. Um exemplo é dado a seguir:

$$\begin{cases} q = P[S(t) = S_0 | S(t-1) = S_1], \\ p = P[S(t) = S_1 | S(t-1) = S_0] \end{cases}$$

Logo, a matriz de probabilidade de transição é dada por

$$P = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix}$$

.

As probabilidades estacionárias são também definidas para representar os estados estacionários: a probabilidade média de recebimento, $\pi_0 = \frac{P_{1,0}}{P_{1,0}+P_{0,1}}$, e a probabilidade média de perda, $\pi_1 = \frac{P_{0,1}}{P_{1,0}+P_{0,1}}$. As expressões matemáticas utilizadas para obter as probabilidades de transição, ou seja $P_{0,1}$ e $P_{1,1}$, são usadas para determinar os parâmetros GE. Dessa forma, $P_E = \pi_1$ e $L_B = 1/q$.

2.3.3 Coeficiente de Correlação de Pearson

O coeficiente de correlação produto-momento de Pearson (ou coeficiente de correlação de Pearson) é uma medida que indica a força de uma associação linear entre duas variáveis. Basicamente, uma correlação de Pearson tenta traçar uma linha de melhor ajuste através dos dados de duas variáveis e o coeficiente de correlação de Pearson indica o quão longe os pontos de dados estão dessa linha de melhor ajuste [75].

O coeficiente de correlação populacional de Pearson, ρ, entre duas variáveis aleatórias X e Y é definido segundo a fórmula:

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$$

O coeficiente de correlação, quando aplicado a uma amostra, é comumente representado pela letra r. Podemos calcular r pela substituição, na fórmula acima, das covariâncias e variâncias estimadas das amostras. O coeficiente de correlação amostral de Pearson, r, entre duas amostras x_i e y_j é definido segundo a fórmula:

$$r = \frac{S_{xy}}{\sqrt{S_{xx}S_{yy}}},$$

onde $S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$.

O coeficiente pode assumir valores no intervalo $[-1, 1]$. O valor 0 indica que não há associação entre as duas amostras. Um valor maior que 0 indica uma associação positiva, ou seja, à medida que o valor de uma amostra aumenta, o mesmo acontece com o valor da outra amostra. Um valor menor que 0 indica uma associação negativa, ou seja, à medida que o valor de uma amostra aumenta, o valor da outra amostra diminui.

3 ESQUEMA DE ESTRUTURAÇÃO SBC-EC PARA LOG SEGURO

A preservação de evidências digitais é uma etapa importante ao longo de uma investigação forense. Esse princípio deve ser respeitado durante todo o tempo de existência de uma evidência para que ela possa ser validada e servir como prova incontestável e legal.

O log de dados é um mecanismo crucial no suporte a uma investigação forense, uma vez que ele é capaz de guardar o histórico de ações e eventos que aconteceram no passado e, portanto, armazenar potenciais evidências. Por esse motivo, o log de dados é um alvo visado por atacantes que pretendem encobrir seus rastros, tornando-se ainda mais vulnerável quando os dispositivos estão localizados em um ambiente hostil, ou não protegido, como no caso de alguns componentes que integram os Sistemas Ciberfísicos e que operam de forma remota ou desassistida (ex., os medidores inteligentes, IEDs, PLCs, sensores sem-fio, etc.). Para uma maior segurança, os dados logados devem ser protegidos assim que gerados e "lacrados" à prova de qualquer alteração maliciosa posterior, até que seja devidamente copiado para um local seguro.

Um investigador pode provar que os arquivos de log são autênticos se ele puder provar que esses arquivos não foram alterados. A detecção de intrusão no log de dados é, portanto, uma importante habilidade para a validação de evidências digitais. Os esquemas de log seguro de dados, por sua vez, apresentam garantia de detecção de possíveis ataques contra o log de dados residente em dispositivos de um ambiente não protegido. Neste capítulo, abordamos a proposta referente à autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques e

que possuem restrições de armazenamento e de comunicação em rede.

3.1 Contextualização do Problema

O log de dados, conforme mencionado, é um importante mecanismo usado na Computação Forense, capaz de oferecer segurança e confiabilidade aos sistemas computacionais. No log de dados podem ser registrados, por exemplo, eventos críticos como o estado e erros de execução de programas, atividades de rede e de usuários e modificação de dados. Dessa forma, um arquivo de log pode ser utilizado para efetuar a análise e diagnóstico de incidentes de segurança, permitindo a rastreabilidade da causa raiz de um problema, ou ataque, no sistema.

Devido aos benefícios obtidos com a implementação do log de dados e por ser de grande valia para uma análise forense, os arquivos de log são constantes alvos de ataques após a invasão de um sistema. Um atacante experiente espera apagar, ou modificar, qualquer informação registrada capaz de revelar sua identidade ou permitir a detecção de traços do comprometimento do sistema. Além disso, o atacante espera manter os métodos de ataque transparentes para o administrador do sistema, escondendo as brechas de segurança para serem utilizadas em um futuro ataque [12]. Dessa forma, torna-se essencial manter o log de dados íntegro e seguro contra qualquer tipo de manipulação maliciosa.

Um requisito do padrão *Common Criteria* [21] estabelece que uma funcionalidade de segurança deve ser capaz de prevenir/detectar modificações não autorizadas aos registros de log armazenados na trilha de auditoria. Uma possível solução a fim de garantir a proteção ao log de dados inclui o uso de técnicas de *hardware* que são resistentes à violação (*tamper-proof*). Entretanto, os componentes de *hardware* demandados por essas técnicas podem onerar muito o custo de produção dos dispo-

sitivos com restrições de recursos, como memória e processamento (p. ex., sensores sem-fio) [59]. Uma técnica alternativa é a adoção de armazenamentos controlados por software do tipo WORM [71], *Write Once Read Multiple*, que garantem que uma informação uma vez escrita não pode ser modificada posteriormente. No entanto, esses armazenamentos são vulneráveis às ações maliciosas de atacantes internos com alto grau de privilégio e acesso físico aos discos.

Uma técnica comum considerada para garantir a proteção ao log de dados é a utilização de um servidor *online* de coleta, responsável por requisitar e armazenar em tempo real as entradas de log dos dispositivos. Essa abordagem possui algumas desvantagens como a dependência de uma comunicação sempre ativa entre os dispositivos e o servidor central de coleta, o que pode não ser factível em determinados tipos de cenários. Com isso, a comunicação incerta, ou não frequente, com o servidor de coleta, além do armazenamento local por tempo indeterminado das entradas de log nos dispositivos, são condições que precisam ser consideradas. Ao longo do tempo, alguns protocolos e técnicas criptográficas foram criados e aprimorados a fim de garantir a proteção dos arquivos de log nos dispositivos, desenvolvendo, portanto, os mecanismos de logs seguros. Os logs seguros são mecanismos de segurança que possuem propriedades como *forward-security*[11] , ser não falsificável, ser a prova de violação, oferecer verificação de integridade (pública ou não), entre outras adicionais como prover confidencialidade e controle de acesso aos registros, verificação individual de entradas e busca por palavras-chave.

Juntamente com a necessidade de segurança, os arquivos de log dos dispositivos tendem a crescer indefinidamente com o passar do tempo, podendo conter milhares de entradas de logs, necessitando cada vez mais espaço de armazenamento [58]. Além disso, as boas práticas do padrão *ISA - Security for Industrial Automation and*

[11]Propriedade que assegura o não comprometimento do uso de chaves utilizadas no passado a partir da revelação de qualquer informação no momento presente [11].

Control Systems [46], antiga ISA-99, determinam, como requisito obrigatório, que sistemas embarcados devem oferecer uma capacidade de armazenamento suficiente para registrar os eventos auditáveis, a fim de que não haja perda de informações entre uma auditoria e outra. Logo, em um cenário onde os dispositivos apresentam restrições de memória e de comunicação em rede, as condições de um arquivo de log sempre crescente e o requisito de um armazenamento suficiente conflitam diretamente.

Com base nos problemas de segurança e armazenamento de arquivos de log citados, o presente capítulo descreve um esquema de estruturação de log seguro apropriado para dispositivos com restrições de armazenamento e de comunicação em rede. O esquema de estruturação SbC-EC apresenta duas propriedades com o objetivo de reduzir o espaço necessário para armazenar entradas de log: (*i*) Separação por Categoria (*Split by Category*, SbC) e (*ii*) Compactação de Entradas de log (*Entry Compaction*, EC). Além disso, apresentamos o esquema de log seguro SbC-EC MAC que utiliza a estruturação proposta. O SbC-EC MAC utiliza o esquema de autenticação *FssAgg* em conjunto com outros mecanismos para estabelecer a segurança necessária ao log de dados. Ao final, apresentamos dois cenários de avaliação. No primeiro, avaliamos comparativamente o esquema SbC-EC MAC em relação aos esquemas de Schneier-Kelsey [88] e FssAgg MAC [59], a fim de mostrar que ele permite um maior ganho de espaço de armazenamento em relação a esses dois. Na segunda parte, simulamos um cenário experimental que demonstra resultados relevantes em relação ao ganho de armazenamento quando aplicamos a categorização e a compactação de entradas de log.

O restante do capítulo está organizado da seguinte maneira. A seção 3.2 destaca os trabalhos importantes na área de logs seguros, mencionando as principais características dos esquemas e possíveis desvantagens. A seção 3.3 descreve o modelo geral do sistema adotado no trabalho e o modelo de atacante, detalhando seus

modos de operação. Na seção 3.4 é descrita a proposta do esquema de estruturação SbC-EC para log seguro e suas propriedades. Em seguida, apresentamos o esquema de log seguro SbC-EC MAC. Cada uma de suas funções e os passos detalhados de funcionamento são descritos na seção 3.5. A seção 3.6 apresenta uma discussão sobre a segurança do SbC-EC MAC, incluindo suas propriedades e possíveis vulnerabilidades. Na seção 3.7, realizamos uma avaliação comparativa entre o nosso esquema de log e outros dois esquemas de log seguro da literatura e, também, mostramos nosso esquema de estruturação aplicado à um cenário experimental. Por fim, concluímos nossa proposta e indicamos nossos esforços futuros na seção 3.8.

3.2 Trabalhos Relacionados

A seguir, destacamos os principais trabalhos relacionados à elaboração de esquemas e características de logs seguros. Diferentemente da abordagem na qual os dispositivos possuem comunicação com um servidor *online* 100% do tempo, esses trabalhos têm como foco os esquemas para proteção dos logs "em repouso" (*logs at rest*). Esses esquemas oferecem soluções tanto no domínio da criptografia simétrica quanto na de chave-pública. Os trabalhos de Bellare e Yee [12, 11] foram os primeiros a incorporar as características de *forward security* aos logs de dados a fim de garantir posteriormente a integridade do fluxo das entradas de log (*forward-secure stream integrity*). Eles montam seu esquema utilizando MACs (*Message Authentication Codes*), onde as entradas de log são indexadas de acordo com períodos de tempo.

Schneier e Kelsey [88, 89] descrevem um esquema de chave simétrica utilizando cadeia de hash autenticada por MACs para estabelecer uma dependência entre as entradas de log, permitindo detectar qualquer alteração feita na ordem das entradas. Os trabalhos também descrevem uma implementação do protocolo para sistemas distribuídos, incluindo troca de mensagens, criação e fechamento de

arquivo de log. Uma das desvantagens do esquema Schneier-Kelsey é a sobrecarga na comunicação e no armazenamento das entradas de log, uma vez que para cada entrada é guardado um MAC para autenticação individual. Além disso, o esquema é suscetível ao ataque de *truncation*, no qual o atacante é capaz de eliminar n últimas entradas do log sem que essa manobra seja percebida em uma verificação posterior.

O Logcrypt [42] é uma extensão do esquema de Schneier-Kelsey utilizando criptografia de chave-pública. Da mesma forma, esse esquema apresenta uma grande sobrecarga na comunicação, por estabelecer frequentes trocas de pares de chaves entre os dispositivos e o servidor central, e no armazenamento, em consequência das assinaturas digitais para uma, ou mais, entradas de log. O Logcrypt também é vulnerável ao ataque de *truncation*. O trabalho de Accorsi [2, 3] descreve o BBox, uma caixa-preta para armazenamento do log de dados em sistemas distribuídos com apoio de componentes de computação confiável, como o TPM (*Trusted Plataform Module*). O esquema de log seguro que compõe o BBox segue a mesma linha do esquema de Schneier-Kelsey, utilizando cadeias de *hash* assinadas digitalmente. Entretanto, essa abordagem também oferece desvantagens em relação à sobrecarga de comunicação/armazenamento devido às assinaturas de cada entrada de log. Além disso, o modelo centralizador adotado não oferece segurança apropriada aos logs nos dispositivos periféricos, os quais podem ser comprometidos de maneiras não previstas no estudo.

Ma e Tsudik [59, 60, 58] desenvolveram o primeiro esquema de autenticação de log seguro apropriado para dispositivos com restrições de comunicação e armazenamento. O esquema de autenticação *FssAgg* (*Forward-Secure Sequential Aggregate*), proposto tanto para a criptografia convencional quanto para a de chave-pública, permite que o dispositivo assinante combine diferentes tags de autenticação criadas em diferentes chaves/períodos de tempo em uma única tag de tamanho constante. Dessa forma, a autenticação do arquivo de log depende de apenas uma tag, resis-

tente ao ataque de *truncation* devido à propriedade "*all-or-nothing*" de verificação de assinatura. O esquema SbC-EC MAC, apresentado no presente trabalho, utiliza a autenticação *FssAgg* a fim de compor a *tag* única para o verificador.

Yavuz et al. [108, 109] desenvolveram o esquema BAF (*Blind Aggregation Forward*), aplicando também a autenticação *FssAgg*. O BAF oferece maneiras menos custosas de realizar a assinatura digital das entradas de log, sendo, portanto, uma opção para dispositivos com baixo poder computacional. Em particular, o nosso trabalho descreve esquemas para sistemas com restrições computacionais, de armazenamento e de rede. Dessa forma, optamos por um esquema de autenticação baseado em criptografia simétrica por se mostrar mais adequada aos nossos propósitos. Yavuz et al. [26] também propõe o esquema LogFAS que oferece eficiência computacional e de armazenamento para o verificador, o qual necessita apenas de uma chave pública pequena e de tamanho constante, independentemente do número de entradas de log. Seu algoritmo de verificação exige menos esforço computacional do que o algoritmo de verificação do BAF, e também permite a verificação de um subconjunto de entradas de log.

Hartung [40] propõe um esquema de log seguro baseado em assinatura digital que pode suportar a verificação de trechos de um arquivo de log. Ele acrescenta marcadores de período ("*epoch*") e números de sequência nas entradas de log, e também permite a categorização das mensagens de log. Crosby e Wallach [23] desenvolvem um projeto de log de auditoria inviolável que permite que um gerador de logs produza provas concisas, ou compromissos ("*commitments*"), de seu comportamento correto. Esses compromissos são enviados regularmente para os auditores que podem verificar o arquivo de log, mesmo com capacidade de armazenamento muito limitada. Além disso, eles propõem uma técnica que permite a exclusão segura e verificável de eventos passados do log de auditoria. Wensheng et al. [106] apresenta um serviço *web* de auditoria segura (SAWS) baseado em Infraestrutura de Chave

Pública, criado a partir do esquema Schneier-Kelsey. Eles utilizam elementos de computação confiável para armazenar as chaves privadas e secretas do serviço de auditoria. Waters et al. [104] descrevem uma abordagem para permitir procura por palavra-chave em um log de auditoria criptografado, e que pode ser combinada com outros esquemas de log seguros.

É importante ressaltar o trabalho de Accorsi [1], que realiza uma revisão da literatura sobre alguns dos artigos sobre log seguro. Mais recentemente, alguns trabalhos [110, 102, 86] concentraram-se em soluções de log seguro abrangendo requisitos de aplicações em um cenário de Computação em Nuvem, como por exemplo, o foco na privacidade dos dados.

O presente trabalho diferencia-se dos demais apresentados na forma de como estruturar o log seguro usado pelos dispositivos. Enquanto os outros trabalhos organizam as entradas de log de acordo com o tempo em que foram geradas e em uma única sequência, nossa estruturação propõe uma organização pelo tempo da geração das entradas, porém separadas em categorias de tipos de log (p. ex., eventos de sistema, erros, alteração de configuração, etc.). Além disso, essa divisão permite que a verificação do log de dados seja feita em subconjuntos de entradas de log, ou seja, por uma categoria de cada vez. Isso oferece, portanto, uma opção intermediária de verificação entre a verificação integral do arquivo de log, quando não necessário, e a verificação individual de entradas, a qual introduz uma sobrecarga de armazenamento para cada uma das entradas.

3.3 Modelos

Nesta seção, é apresentada a descrição do Modelo do Sistema (seção 3.3.1) e do Atacante (seção 3.3.2) utilizados neste trabalho.

3.3.1 Modelo do Sistema

O modelo de sistema utilizado nesse trabalho é baseado no modelo descrito por Schneier e Kelsey [89]. O sistema é composto basicamente por três partes, T (máquina confiável), U (máquina suscetível a ameaças) e V (verificador semi-confiável), que são detalhadas na sequência.

1. T, a máquina confiável. T pode ser instanciado de diversas maneiras, p. ex., como um servidor em um ambiente controlado. T possui armazenamento suficiente para guardar todas as entradas de log de U, e autoriza um verificador V a ter acesso aos logs de U. Por fim, T realiza a verificação final dos arquivos de log, uma vez transmitidas a ele;

2. U, a máquina suscetível a ameaças. Não há garantias de que essa máquina está totalmente segura contra atacantes, erros de software, entre outras ameaças. Nela são geradas e armazenadas, por um período de tempo indeterminado, as entradas de log até que sejam transmitidas a T. Uma vez comprometida, U agirá da forma que o atacante estipular. U é capaz de se comunicar e interagir com T e V;

3. V, o verificador semi-confiável. O verificador pode realizar a função de inspecionar a integridade do arquivo de log de dados, porém sem alterar nenhuma entrada. Em geral, V se caracteriza por um pequeno grupo de entidades autorizadas (p. ex., administradores de sistemas) a obter e ler uma cópia dos arquivos de log de U, quando necessário.

Durante o período de inicialização do sistema é realizada uma comunicação inicial entre o servidor T e o dispositivo U a fim de compartilhar os segredos iniciais do esquema de log seguro. Assumimos que todas as transmissões entre as partes

do sistema sejam estabelecidas por uma comunicação segura, utilizando protocolos reconhecidos de autenticação e de troca-de-chaves.

O dispositivo U gera entradas de log de formato bem definido e as guarda em seu armazenamento não volátil. Em intervalos de tempo não frequentes e não determinísticos, o dispositivo U transmite seus logs armazenados para o servidor T. Uma vez que os dados de log são recebidos por T, eles são assumidos seguros. Por sua vez, T verifica a integridade desses dados e os armazena. Além da capacidade de U de armazenar logs, ele também é capaz de apagar informações a fim de liberar espaço na memória, p. ex. entradas de log enviadas, e a fim de cumprir o funcionamento do esquema proposto, p. ex. apagar antigas chaves criptográficas. Assumimos, também, que U é capaz de gerar números pseudo-aleatórios de maneira robusta.

Abaixo, apresentamos as primitivas criptográficas e suas notações usadas ao longo do capítulo:

1. Função pseudo-aleatória – $prf(K_i)$ recebe como entrada a chave K_i, e tem como saída a chave K_{i+1};

2. Função de sentido único – $H(M)$ recebe uma mensagem M e calcula um resumo criptográfico de tamanho fixo. $H()$ pode ser implementada com o algoritmo SHA-256 [67];

3. *Message Authentication Code* – $MAC(K, M)$ fornece uma *tag* que verifica a autenticidade e integridade da mensagem M, dada a chave criptográfica simétrica K. Pode ser implementada com o algoritmo HMAC [10];

4. Algoritmo de criptografia simétrica – $E(K, M)$ realiza a cifragem simétrica da mensagem M utilizando a chave K. $E()$ pode ser implementado com o algoritmo AES [66].

3.3.2 Modelo de Atacante

Como dito anteriormente, os logs de dados são alvos importantes para atacantes que queiram esconder traços de seu ataque, agindo de forma a apagar, modificar, inserir ou reordenar as entradas de log a fim de atingir seu objetivo. De fato, os esquemas de log seguro não protegem contra ações maliciosas no arquivo de log. A característica principal de um esquema de log seguro é garantir que qualquer modificação indevida realizada seja detectada no ato da verificação do arquivo de log.

A seguir, consideramos as possibilidades de um atacante durante o comprometimento de U. Dada a última transmissão de entradas de log de U para T no instante de tempo t_k e que o comprometimento de U acontece no instante de tempo t_{k+l}, é assumido que:

- O atacante toma conhecimento das chaves $A_{t_{k+l}}$ e $B_{t_{k+l}}$ (definidas na seção 3.5), guardadas em U, no momento do comprometimento e é capaz de derivar todas as evoluções futuras dessas chaves;

- O atacante é capaz de gerar as tags do verificador e da parte confiável, uma vez que o algoritmo de geração de tags é público;

- As entradas de log que foram geradas em U, após o instante de tempo t_k, não podem ser consideradas confiáveis. O atacante tem a capacidade de apagá-las, modificá-las, reordená-las ou inserir novas entradas;

- O atacante desconhece as chaves de autenticação das entradas de log passadas $L_{t_k}...L_{t_{i+j-1}}$. Portanto, qualquer manipulação realizada nessas entradas poderá ser detectada em uma futura verificação do arquivo de log por V ou T;

- Qualquer entrada de log gerada após $L_{t_{k+l}}$, inclusive, poderá ser manipulada a fim de evitar uma detecção posterior.

Apesar do atacante ser capaz de realizar alguma modificação, sem ser detectada, em qualquer entrada após $L_{t_{k+l}}$, espera-se que ele também deseje alterar entradas anteriores a t_{k+l} por conterem possíveis rastros de suas ações. A detecção de um ataque ao log de dados será, portanto, realizada com sucesso para o conjunto de entradas de log criadas anteriormente ao comprometimento.

3.4 Esquema de Estruturação SbC-EC para Log Seguro

Um dos desafios relacionados à geração de logs de dados em dispositivos, em geral, é a necessidade de armazenamento de um grande, e crescente, volume de informação ao longo do tempo de operação. Com o objetivo de diminuir o espaço necessário para o armazenamento dessas informações, propomos um novo esquema de Estruturação para log seguro, denominado SbC-EC, que apresenta uma maneira diferente de organizar semanticamente as entradas de log em relação aos trabalhos da literatura.

A estratégia principal adotada no esquema de estruturação SbC-EC reside na eliminação de informação redundante encontrada nos *payloads* das entradas de log semelhantes. Uma vez que o mecanismo de log seguro detém a propriedade de *forward-integrity*, é impossível realizar alterações nas entradas passadas de forma manter a integridade do fluxo de entradas, logo, qualquer alteração de *payload* deve ser executada na entrada antes de ser logada. Para eliminar informações redundantes do *payload*, a entrada a ser logada L_i deverá referenciar alguma anterior semelhante a ela, porém, isso torna-se impeditivo quando o número de entradas é muito grande, necessitando realizar buscas cada vez maiores para logar uma só entrada. Dessa

forma, limitamos a comparação apenas a um nível, ou seja, caso o *payload* de L_i for semelhante ao de L_{i-1}, L_i é logada de forma compactada. Esse procedimento estabelece a propriedade de *Compactação de Entradas* (*Entry Compaction*, EC) para o esquema de estruturação aqui apresentado.

A *Compactação de Entradas* funciona da seguinte maneira. Consideramos um *payload* bem estruturado P_n, composto pela tupla $(ID_n, M_n, [Params_n], TS_n)$, onde ID_n é o identificador único da mensagem de log, M_n é a mensagem de log, $[Params_n]$ é o conjunto de parâmetros referente à mensagem de log, caso haja, e TS_n é o *timestamp* da geração do *payload*. Quando um novo *payload* P_i está para ser logado, é verificado se $ID_i = ID_{i-1}$, caso seja verdade, o *payload* P_i será composto pela tupla $(TS_{dif}, [Params_i])$, onde TS_{dif} é a diferença horária entre o *timestamp* de P_i e o *timestamp* de P_{i-1}, isto é $TS_{dif} = TS_i - TS_{i-1}$.

Uma vez que os eventos auditáveis são de diversos tipos, é baixa a probabilidade de que, em um único fluxo de entradas, duas entradas semelhantes sejam logadas consecutivamente. A partir disso, desejamos aumentar essa probabilidade e, consequentemente, compactar mais entradas e poupar mais espaço de armazenamento. Para que isso seja possível, a propriedade de *Separação por Categoria* (*Split by Category*, SbC) foi estabelecida. A estruturação SbC-EC classifica cada entrada de log gerada em uma das $1..c$ categorias pré-estabelecidas e a concatena junto às demais da mesma categoria, em diferentes fluxos de entradas de log. Cada fluxo corresponde a um arquivo de log.

Assumimos que um componente de *software* seja capaz de classificar, tomando como base o campo ID, a categoria de cada *payload* de log antes de ser concatenada como uma entrada no arquivo de log. Uma categoria de log corresponde aos grupos de registros de eventos auditáveis semelhantes, tais como eventos de controle de acesso, erros de requisição, eventos de sistema, alteração de configu-

ração, potenciais atividades de reconhecimento e eventos de log [46].

3.5 Esquema de Log Seguro SbC-EC MAC

Nesta seção, é apresentado o esquema de log seguro, o SbC-EC MAC, utilizando criptografia simétrica e com base na estruturação SbC-EC. A criptografia simétrica é aqui aplicada pois oferece uma sobrecarga de autenticação menor do que a criptografia assimétrica em relação ao tamanho de dados gerados, além, também, de exigir menos tempo e custo computacional, o que torna o esquema apropriado para dispositivos com restrições de recursos. Além disso, o SbC-EC MAC utiliza o esquema simétrico de autenticação *FssAgg* [58], adaptado à estruturação SbC-EC, a fim de gerar as tags de autenticação necessárias das entradas de log. O esquema *FssAgg* estipula uma tag única (assinatura agregada) a fim de autenticar todas as entradas de log existentes em um arquivo de log.

A seguir, são descritos os algoritmos presentes no esquema SbC-EC MAC:

- *SbC-EC.Skg – Secret key generation*, o algoritmo de geração de chave secreta é usado para gerar uma chave inicial de tamanho seguro a ser usada pelo algoritmo de criptografia simétrica.
- *SbC-EC.Ckg – Category key generation*, o algoritmo de geração das chaves iniciais de verificador para cada categoria. O algoritmo recebe como entrada uma chave inicial de verificador A_0 e o número de categorias c, e responde, como saída, as chaves de verificador $A_{1,1}, A_{1,2}, ..., A_{1,c}$, uma para cada categoria.
- *SbC-EC.Vsig – Verifier signature*, o algoritmo assina-e-agrega (*sign-and-aggregate*) a fim de formar a tag de autenticação do verificador. O algoritmo recebe, como entradas, uma chave secreta, uma mensagem a ser assinada e a tag de auten-

ticação computada até aquele ponto. Ele calcula a assinatura da mensagem e a agrega à tag já existente. Ao final, o algoritmo realiza uma atualização da chave secreta usada. O *SbC-EC.Kupd*, *Key update*, é realizado dentro do algoritmo de assinatura para se obter melhores garantias de segurança. Esses algoritmos remetem aos *FssAgg.Asig* e *FssAgg.Upd* descritos por Ma e Tsudik.

- *SbC-EC.Tsig – Trusted party signature*, o algoritmo que gera a tag de autenticação da parte confiável. Ele recebe, como entradas, uma chave secreta e c tags de autenticação, $c \geq 1$. O algoritmo concatena e assina as tags da entrada para gerar uma única tag. Por fim, o algoritmo atualiza a chave secreta utilizando *SbC-EC.Kupd*.

- *SbC-EC.Vver – Verification*, o algoritmo de verificação da parte verificadora V recebe um fluxo de entradas de log, o número de sequência da primeira entrada do fluxo, uma chave secreta da categoria e a tag de autenticação correspondente, e responde se a sequência de entradas está íntegra ou não. Esse algoritmo remete ao *FssAgg.Aver*.

- *SbC-EC.Tver – Trusted party verification*, o algoritmo de verificação da parte confiável recebe c fluxos de entradas de log em conjunto com suas c tags de autenticação, $c \geq 1$, a chave secreta da parte confiável, uma tag de autenticação da parte confiável verificada anteriormente e uma tag de autenticação da parte confiável recebida de U a ser verificada. O algoritmo recalcula a tag da parte confiável a ser verificada a partir dos dados inseridos e responde, caso essa tag for igual à tag da parte confiável informada, os fluxos de log estão íntegros, caso contrário, não.

Um dos mecanismos usados pelo SbC-EC MAC é o de evolução de chaves secretas, usada também em outros esquemas de log seguro e em aplicações que exigem propriedades como *forward-security*. Esse mecanismo permite que uma chave

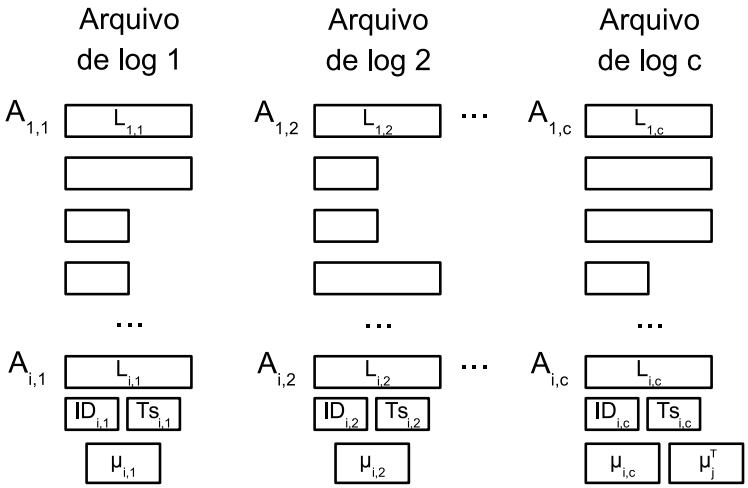

Figura 3.1: Representação dos elementos do esquema SbC-EC MAC.

K_i evolua para K_{i+1} de forma determinística através de uma função pseudo-aleatória $prf(K_i)$. É computacionalmente inviável obter K_i a partir de K_{i+1}.

No SbC-EC MAC temos que, para cada arquivo de log representando uma categoria m, onde $1 \leq m \leq c$, existe uma tag única de verificador. Essa tag de verificador é representada por $\mu_{i,m}$, correspondente à i-ésima entrada de log da categoria m. Existe, também, uma única tag da parte confiável μ_j, correspondente à j-ésima evolução da chave secreta da parte confiável B_j. A Figura 3.1 mostra uma representação dos diferentes arquivos de log, cada um com suas correspondentes entradas de log, compactadas ou não, as chaves secretas de evolução e tags de verificador, e a única tag da parte confiável existente no esquema.

A seguir, descrevemos os passos do esquema SbC-EC MAC desde a preparação e inicialização dos arquivos de log, a concatenação e compactação de novas entradas de log e, por fim, as maneiras de realizar as verificações do log de dados para as diferentes partes do sistema.

3.5.1 Inicialização do log de dados

O dispositivo U deve efetuar alguns passos de preparação do arquivo de log antes de criá-lo. Primeiramente, U deve utilizar o algoritmo *SbC-EC.Skg* para gerar duas chaves secretas: (i) A_0, chave inicial do verificador V e (ii) B_0, chave inicial da parte confiável T. U deve definir c, o número de categorias de logs, podendo esse dado estar predefinido e protegido no dispositivo. A partir disso, U deve informar a T esses 3 dados (A_0, B_0, c). Assumimos que a comunicação entre U e T seja segura, autenticada e à prova de ataques.

Após a confirmação de que T recebeu os dados enviados, U utiliza a função *SbC-EC.Ckg* para gerar todas as chaves iniciais de verificador de cada categoria. A partir disso, inicializa-se c arquivos de log, um para cada categoria. Para garantir a detecção de um ataque de deleção total do arquivo de log, antes de qualquer entrada legítima ser guardada, uma entrada "*dummy*" é gerada e concatenada em todos os arquivos. Essa medida evita que, após um atacante comprometer U e apagar todas as entradas de log existentes até o momento, ele não consiga esconder esse fato reivindicando que o arquivo de log nunca foi inicializado.

3.5.2 Concatenação e compactação de entradas de log

Para cada última entrada de cada arquivo de log, são guardados em claro (ou seja, sem estar criptografados) os dados $ID_{i-1,m}$ e $TS_{i-1,m}$, isso se justifica pelo processo de compactação de entradas, como explicado na seção 3.4. Descrevemos, a seguir, os procedimentos necessários durante o procedimento de concatenação de uma nova entrada de log. Primeiramente, é descrita a concatenação de uma entrada normal (quando $ID_i \neq ID_{i-1}$) e, em seguida, a concatenação de uma entrada compactada (quando $ID_i = ID_{i-1}$).

A partir da geração de um novo *payload* P, uma função decisora de categoria é responsável por escolher qual arquivo de log m deve concatenar a nova entrada. Uma vez definida a categoria, inicia-se o procedimento de concatenação da entrada. Assumimos que o arquivo de log da categoria m já possui as entradas $L_{1,m}$, $L_{2,m}$, $L_{3,m}$, ..., $L_{i-1,m}$. Os passos necessários para concatenar a entrada $L_{i,m}$ não compactada são descritos a seguir (Figura 3.2):

1. ID_i é comparado com ID_{i-1}. Como são diferentes neste caso, ID_i é guardado e ID_{i-1} é apagado.

2. TS_i é guardado e TS_{i-1} é apagado.

3. $L_{i,m} = E(A_{i,m}, P_i)$, gera a nova entrada a partir da cifragem simétrica do *payload* P_i com a chave $A_{i,m}$.

4. Gera a i-ésima assinatura agregada $\mu_{i,m}$, tag do verificador, calculada conforme a Equação 3.1 [58]:

$$\mu_{i,m} = H(H(...H(\mu_{1,m}||MAC(A_{1,m}, L_{1,m}))...)||MAC(A_{i,m}, L_{i,m})) \tag{3.1}$$

5. *SbC-EC.Kupd($A_{i,m}$)*, evolui a chave $A_{i,m}$ para $A_{i+1,m}$.

6. Gera a j-ésima tag da parte confiável, calculada conforme a Equação 3.2:

$$\mu_j^T = MAC(B_j, \mu_{j-1}^T||\mu_1||\mu_2||...||\mu_c) \tag{3.2}$$

   onde μ_m é a tag atual de verificador da categoria m, $1 \leq m \leq c$.

7. *SbC-EC.Kupd(B_j)*, evolui a chave B_j para B_{j+1}.

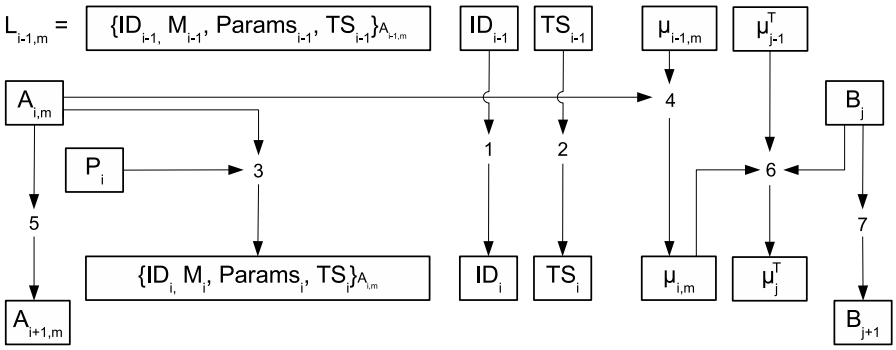

Figura 3.2: Concatenação de uma nova entrada $L_{i,m}$ não compactada.

Para a concatenação de uma nova entrada de log $L_{i,m}$ em que ID_i é igual ao da última entrada $L_{i-1,m}$, a nova entrada será compactada. Os passos necessários para concatenar $L_{i,m}$ compactada são descritos abaixo:

1. ID_i é comparado com ID_{i-1}. Como são iguais nesse caso, ID_i é descartado e ID_{i-1} é mantido.

2. $TS_{dif} = TS_i - TS_{i-1}$, a diferença horária entre as entradas. TS_i é guardado e TS_{i-1} é apagado.

3. $L_{i,m} = E(A_{i,m}, P_i)$, gera a nova entrada a partir da cifragem simétrica do *payload* P_i com a chave $A_{i,m}$. Onde $P_i = (TS_{dif}, [Params_i])$.

O restante do procedimento é idêntico aos passos 4 à 7 descritos para a concatenação de uma entrada não compactada.

3.5.3 Verificação do log de dados

A verificação do log de dados pode ser executada tanto pelo verificador V, quanto pelo servidor T. Para iniciar o procedimento, V requisita e obtém, de forma

segura, de T a chave inicial de verificador. Com a chave em posse, V consegue calcular todas as chaves iniciais de todas as categorias de log através da função *SbC-EC.Ckg*. V requisita os arquivos de log de seu interesse a U e recalcula a tag de verificador $\mu_{i,m}$, pela Equação 3.1, de cada arquivo de log m. V então verifica se as tags são idênticas às que U armazena. Se sim, o log de dados está íntegro, caso contrário, isso implica que ao menos um elemento de toda a sequência não está correto e, portanto, não está íntegro.

Como dito anteriormente, a estratégia de separação por categorias oferece uma vantagem de que, se V estiver interessado em apenas um tipo de log (p. ex., atividades de rede), ele só precisará requisitar e verificar o arquivo de log da categoria desejada, sem a necessidade de verificar todas as entradas de todos os arquivos de log. Na verificação feita por T, é necessário requisitar e verificar todos os arquivos de log. T realiza o mesmo procedimento de recalcular as tags de verificador para cada arquivo de log e, por fim, recalcula a tag da parte confiável pela Equação 3.2. Por fim, T compara essa tag com a tag da parte confiável informada por U. Se as duas tags forem iguais, os arquivos de log estão íntegros, caso contrário, não estão.

3.6 Discussão de Segurança

O esquema SbC-EC MAC utiliza o esquema *FssAgg* como camada de autenticação para criar as tags de verificador. A cada nova entrada de log gerada, o protocolo recria a tag de autenticação única, e de tamanho constante, para o arquivo de log correspondente. A segurança do esquema *FssAgg* é provada nos trabalhos de Ma e Tsudik [59] [60] [58]. Além disso, o esquema SbC-EC MAC é resistente ao ataque de *truncation* (i.e., a deleção das entradas mais recentes do arquivo de log) devido à propriedade "*all-or-nothing*" de verificação do arquivo de log herdado da autenticação *FssAgg*. A propriedade "*all-or-nothing*" estabelece que, se ao menos

uma entrada tenha sido alterada, a verificação de todo o arquivo de log falhará.

Ataque de detecção atrasada. Por possuir a chave secreta inicial A_0, um usuário/verificador, V, autorizado e mal-intencionado é capaz de realizar alterações nas entradas de log e recriar as assinaturas agregadas – ou seja, as tags de verificador – com o objetivo de esconder alguma manipulação feita. Outros usuários autorizados, portanto, não poderão detectar essas alterações feitas por V e verificariam com sucesso os arquivos de log. A detecção da manipulação por V é possível apenas para o servidor T com ajuda da tag da parte confiável μ_j.

Dessa forma, o esquema SbC-EC MAC é suscetível ao tipo de ataque denominado *delayed detection* (detecção atrasada), devido ao fato de que nenhum V pode detectar prontamente uma manipulação realizada por outro V, e que a comunicação entre U e T só acontece em períodos de tempo indeterminados. Por outro lado, se o ataque não é realizada por um usuário privilegiado, V pode detectar prontamente qualquer alteração indevida no log de dados.

Importância da tag da parte confiável. Por razões de segurança, μ^T deve ser atualizada após cada nova entrada de log concatenada. A partir do procedimento de atualização (Equação 3.2), a tag μj^T é calculada a partir de uma função de $MAC()$, onde o parâmetro é a concatenação de todas as atuais tags de verificador com a última tag da parte confiável μ^T_{j-1}. Esta estratégia garante que quaisquer alterações às tags de verificador são detectáveis por T.

Além disso, uma vez que a chave da parte confiável B evolui através de uma função pseudo-aleatória, $prf()$, após cada atualização da *tag*, μ^T não pode ser alterada de forma despercebida por V ou um atacante, porque eles não têm acesso aos valores anteriores de B. Em particular, a adição do valor da *tag* anterior, μ^T_{j-1}, na geração da próxima *tag* (Equação 3.2) assegura a detecção de ataques realizados

por usuários privilegiados ou verificadores maliciosos.

Demais propriedades. Por fim, salientamos que o esquema SbC-EC MAC oferece a propriedade de *forward-secrecy* e *forward secure stream integrity*. A propriedade *forward-secrecy* oferece a permanência da confidencialidade das entradas de log concatenadas ao arquivo antes de uma intrusão, ou seja, mesmo que um atacante consiga invadir o sistema no tempo t, a confidencialidade das entradas geradas antes de t é mantida. Da mesma forma, a propriedade *forward secure stream integrity* também garante a integridade da sequencia das entradas do arquivo de log antes do momento t. Essas duas propriedades têm como base o mecanismo de evolução de chaves simétricas.

3.7 Avaliação

Nesta seção, avaliamos a nossa proposta através de uma análise comparativa e, depois, em um cenário experimental.

3.7.1 Análise Comparativa

Para esta análise, o esquema SbC-EC MAC é avaliado de acordo com o ganho de armazenamento em memória em relação a outros esquemas de log seguro de criptografia simétrica existentes na literatura: (i) Schneier-Kelsey, e (ii) *FssAgg* MAC. A seguir, descrevemos algumas premissas sobre o armazenamento necessário de cada item presente nos esquemas e, por fim, uma simulação é realizada com fins comparativos.

Sobrecarga de autenticação. Seja M_{key} o espaço de armazenamento necessário

Tabela 3.1: Sobrecarga de autenticação dos esquemas de log seguro

	Sobrecarga	Ordem
Schneier-Kelsey	$M_{key} + N(M_{hash} + M_{mac})$	$O(N)$
FssAgg MAC	$2M_{key} + 2M_{hash}$	$O(1)$
SbC-EC MAC	$(c+1)M_{key} + cM_{hash} + M_{mac}$	$O(c)$

para guardar as chaves privadas, M_{mac} o espaço necessário para guardar uma saída obtida de uma função $MAC()$ (p. ex., assinaturas, tags), M_{hash} o espaço necessário para guardar uma saída obtida de uma função *hash* $H()$, dado N o número total de entradas de log geradas e c o número de categorias de log, verificamos na Tabela 3.1 a sobrecarga de armazenamento de autenticação associada a cada esquema. Pela tabela, conclui-se que a sobrecarga de autenticação do esquema de Schneier-Kelsey está diretamente ligada ao número de entradas logadas, uma vez que esse esquema mantém uma tag de autenticação para cada entrada. O esquema FssAgg MAC é o que implica na menor sobrecarga, utilizando apenas duas chaves e duas tags para autenticação de todo o arquivo de log. O esquema SbC-EC MAC apresenta uma sobrecarga de autenticação diretamente relacionada ao número de categorias de log estipuladas, pois, para cada categoria, uma chave e uma tag são armazenadas.

Ganho de espaço de armazenamento. Para avaliar o ganho de espaço de armazenamento do esquema SbC-EC MAC, realizamos uma simulação de geração de entradas de log, em MATLAB, a fim de calcular o tamanho total ocupado pelo log de dados protegido pelos esquemas aqui testados. Para essa simulação, definimos um dispositivo capaz de registrar 400 mensagens de log diferentes – ou seja, 400 IDs – e que segue uma distribuição normal para a geração de logs. Assumimos que M_{key}, M_{mac} e M_{hash} são equivalentes a 1 unidade de armazenamento, o tamanho médio de uma entrada de log é de 2 unidades de armazenamento e o tamanho médio de uma entrada de log compactada é de 1 unidade de armazenamento. Os demais itens que compõem os logs de dados de cada esquema puderam ser desconsiderados, por não impactar significativamente no total de armazenamento ocupado.

Tabela 3.2: Resultado da simulação em 1.000 rodadas

	Tamanho médio (un. de armazenamento)	Desvio padrão (σ)	Ganho de arm. (%)
Schneier-Kelsey	40.000	0	-
FssAgg MAC	20.000	0	50,00
SbC-EC MAC (c =1)	19.719	16,60	50,70
SbC-EC MAC (c =8)	17.743	42,56	55,64
SbC-EC MAC (c =10)	17.181	46,42	57,05
SbC-EC MAC (c =20)	14.379	60,39	64,05

Com o intuito de obter o tamanho médio e o ganho de armazenamento dos esquemas FssAgg MAC e SbC-EC MAC em relação ao esquema de Schneier-Kelsey, realizamos 1.000 rodadas de simulação para $N = 10.000$ entradas de log. Os resultados são mostrados na Tabela 3.2. Para os esquemas Schneier-Kelsey e *FssAgg* MAC, o armazenamento se mantém constante para todas as rodadas, uma vez que eles detêm o mesmo comportamento independente das entradas de log geradas. O FssAgg MAC oferece um ganho total de armazenamento de 50% em relação ao esquema anterior. Para o esquema SbC-EC MAC, observa-se que, quanto maior o número de categorias c estipulado, menor as unidades de armazenamento necessárias para guardar o log de dados. Isso é explicado pelo fato de que há mais probabilidade para a condição $ID_i = ID_{i-1}$ ser satisfeita quando há mais divisões, consequentemente agrupando entradas de log de categorias, ou sub-categorias, cada vez mais semelhantes. Dessa forma, nosso esquema permite um maior ganho de armazenamento em relação ao *FssAgg* MAC. Essa avaliação evidencia a importância da categorização de mensagens de log no sistema para o esquema SbC-EC MAC e que, quanto maior a granularidade dessa categorização, maior o ganho de armazenamento.

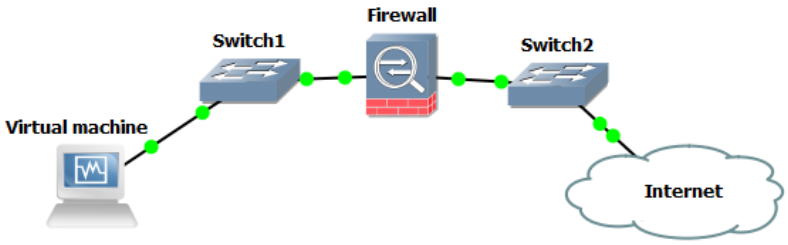

Figura 3.3: Cenário de simulação.

3.7.2 Resultados Experimentais

Nesta parte da avaliação, testaremos o nosso esquema de estrutura de log seguro SbC-EC em um cenário experimental. Desta vez, nosso objetivo é demonstrar o funcionamento do esquema em um dispositivo real, exemplo de modificações causadas nas entradas de log gerados por esse dispositivo e o ganho total de memória de armazenamento após o processamento do esquema.

Preparação e configuração da simulação. O dispositivo escolhido para a geração de logs é o *Adaptive Security Appliance* (ASA)[2], software version 8.4, da fabricante Cisco Systems. O cenário consiste no *firewall* ASA conectando uma máquina virtual à comunicação externa com a internet, ver Figura 3.3. Para implementar o cenário da simulação, utilizamos o simulador *Graphical Network Simulator-3* (GNS3) versao 1.3.14dev1, com Python 3.4.2 e Qt 4.8.6, instalado em uma máquina física rodando o Windows 7 64-bit com 6 gigabytes de RAM. O *firewall* ASA é do modelo ASA 5520, representado por um Pentium II 1000 MHz com 1 gigabyte de RAM e 256 megabytes de memória flash, enquanto a máquina virtual roda o sistema Ubuntu 14.04.4 com 2 gigabytes de RAM.

O *firewall* está operando inspeções padrão de tráfego e políticas com configu-

[2] *Adaptive Security Appliance* website: *http://www.cisco.com/c/en/us/products/security /adaptive-security-appliance-asa-software/index.html*

rações padrão. A configuração de tamanho máximo do *buffer* interno foi ajustada para 1 megabyte, que é o máximo valor suportado por esse modelo. Quando o *buffer* interno é preenchido totalmente por dados de log, seu conteúdo é salvo em um arquivo de log na memória flash e, então, o buffer é apagado para que possa guardar novos dados. O *firewall* ASA utiliza o padrão *syslog* para a gravação de mensagens de log. O formato de mensagem *syslog* (*Syslog Message Format*) para o *firewall* ASA é composto dos sequintes dados {*Timestamp*, palavra "ASA", *Severity Level*, *Message_number*, *Message_text*}[3]. O nível de severidade (Severity Level) pode ser configurado para estabelecer o nível de granularidade de gravação de mensagens de log. Os valores possíveis são: 0-*emergencies*, 1-*alert*, 2-*critical*, 3-*error*, 4-*warning*, 5-*notification*, 6-*informational*, 7-*debugging*. Quanto maior o nível, maior o volume de dados gerados nos arquivos de log. Para a nossa simulação, estabelecemos o nível de severidade igual a 6.

O valor *Message_number* representa um número único de seis dígitos que identifica uma mensagem syslog. Os três primeiros dígitos do *Message_number* representam a categoria de log da qual essa mensagem faz parte. Essas categorias são pré estabelecidas pela fabricante. Utilizamos a categorização existente para dividir as mensagens de log em nosso esquema.

Os dados de log experimentais foram gerados na máquina *firewall* através de atividades *online* usuais realizadas a partir da máquina virtual. Algumas dessas atividades são: navegação de páginas web, atualização e instalação de *software*, *streaming* de vídeos, mídias sociais e *chats*, *upload/download* de arquivos via protocolos torrent e ftp, configuração remota e comandos administrativos do *firewall* ASA. Essas atividades foram conduzidas por, aproximadamente, 5 dias, suficientes para gerar 100 arquivos de logs no *firewall*. Cada arquivo de log possui 1 megabyte

[3]ASA system logs details: *http://www.cisco.com/c/en/us/td/docs/security/asa/asa91/configuration/general/asa_91_general_config/monitor_syslog.html*

Tabela 3.3: IDs/Categorias de log [20]

106	Access Lists
111	Command Interface
742	Password Encryption
302, 305, 502, 710	User Session
199, 411, 604, 711	System
733	Threat Detection
713	IKE and IPsec
720	VPN Failover
721	WebVPN Failover

de tamanho.

Exemplo de compactação de entrada. Abaixo, mostramos dois exemplos de entradas de log não compactadas e três exemplos de entradas compactadas. Cada entrada não compactada pertence a uma categoria diferente Os trechos foram retirados da simulação, e são mostrados sem a aplicação do esquema de log seguro SbC-EC MAC, ou seja, apenas com os resultados da estruturação proposta, sem cifragem das informações. Para os exemplos compactados, o valor antes de ";" é a diferença de tempo em relação à última entrada (TS_{dif}), e os valores após ";" são os parâmetros específicos de cada entrada de log ($Params_i$).

Categoria 604 – System:

Jul 08 2016 15:59:44: %ASA-6-604103: DHCP daemon interface inside: address granted 0800.2728.e9f4 (172.16.1.100)
1500.0; inside 0800.2728.e9f4 172.16.1.100

Categoria 302 – User Session:

Jul 08 2016 15:59:46: %ASA-6-302016: Teardown UDP connection 5 for outside:8.8.8.8/53 to inside:172.16.1.100/61448
duration 0:00:00 bytes 96
0.0; 6 outside 8.8.8.8 53 inside 172.16.1.100 47579 0:00:00 92
0.0; 7 outside 8.8.8.8 53 inside 172.16.1.100 41122 0:00:00 121

Tabela 3.4: Resultados do log de dados total

Número total de entradas de log	691.913
Número total de entradas compactadas	452.400
Tamanho do log de dados original	102,40 MB
Tamanho do log de dados pós compactação	74,27 MB

Resultados do log de dados total. Após a geração dos arquivos de log, aplicamos as transformações do esquema SbC-EC nos arquivos através de um script Python[4]. Para separar todo o log de dados, o esquema SbC-EC utilizou 15 categorias, ver Tabela 3.3. Em um total de 691.913 entradas de log, 452.400 entradas foram compactadas ao longo da simulação, o que representa 65,38% do total. O tamanho de armazenamento, originalmente ocupado pelo arquivo de log, era de 102,40 megabytes, enquanto, após o processo do esquema, passou a ser 74,27 megabytes, 72,52% (ver Tabela 3.4).

Resultados do log de dados por arquivo . Verificamos, também, os resultados em relação a cada arquivo de log em separado, como se cada arquivo fosse um log de dados independente. Os resultados dessa avaliação encontram-se na Tabela 3.5, onde adicionamos o cálculo do desvio padrão para cada resultado obtido. A sobrecarga de armazenamento dos meta dados de cada arquivo no sistema não foi considerado nos cálculos do tamanho dos arquivos. No sistema de arquivos *ext4*, por exemplo, o tamanho padrão de um *inode* – utilizado para guardar meta dados de um arquivo – é de 256 bytes [29] e, portanto, podem ser neglicenciados no cálculo final.

[4]O script Python e os arquivos de log podem ser baixados em *https://www.dropbox.com/s/t76prx23zsjyx2z/sbc-ec_simulation.zip*

Tabela 3.5: Resultados do log de dados por arquivo

	Média	Desvio padrão
Número de entradas de log	6.919,13	183,25
Número de entradas compactadas	4.524	770,30
Tamanho dos arquivos compactados	742,73 KB	51,54 KB

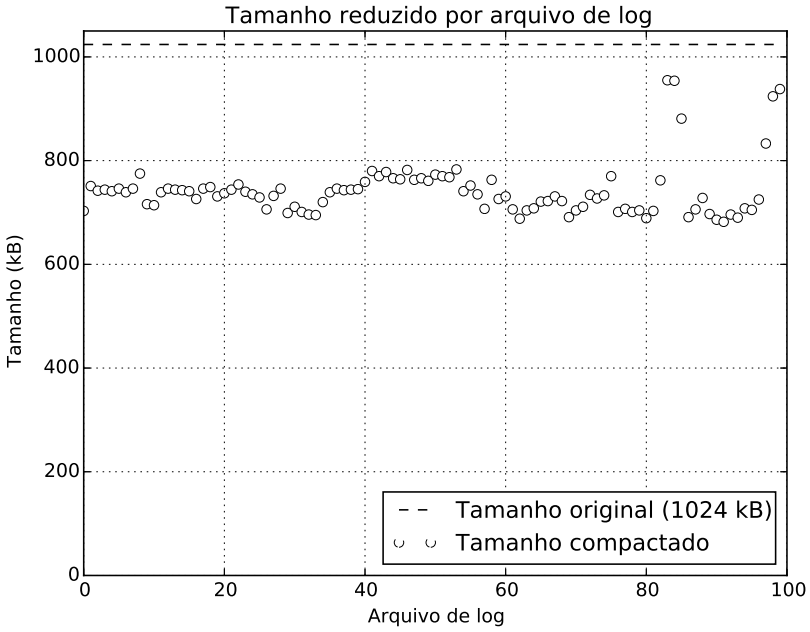

Figura 3.4: Redução de tamanho de cada arquivo de log.

A Figura 3.4 mostra a redução de tamanho de armazenamento para cada arquivo de log considerado na simulação. Em geral, para cada arquivo, os resultados de redução de tamanho obtidos são similares. Entretanto, podemos observar que, para alguns arquivos, a redução do tamanho não foi tão significativa, ex. 84° e 85° arquivos de log. Ao examinar esses arquivos em particular, verificamos que neles estão gravadas entradas de log sobre a atividade de rede causada intencionalmente pelo uso prolongado do comando *ping*. Nesses arquivos existe uma grande quantidade de duas entradas de log específicas relacionadas a "Building outbound connection" e "Teardown connection", IDs ASA-6-302020 e ASA-6-30202 respectivamente, do protocolo ICMP (*Internet Control Message Protocol*), que aparecem de forma alternada e levando, portanto, a condição de compactação de entradas a falhar.

Um dos pontos relevantes dessa simulação é evidenciar que há, por muitas

vezes, a repetição das mesmas mensagens de log, e que a maioria de todas as mensagens de log possíveis não são usadas com frequência pelo sistema. Sendo assim, possibilitando o ganho de armazenamento do esquema SbC-EC.

3.8 Conclusão

Neste Capítulo, apresentamos uma proposta de autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques e que possuem recursos limitados, como no caso de alguns componentes que integram os Sistemas Ciber-físicos e que operam de forma remota ou desassistida. Descrevemos o esquema de estruturação de log seguro SbC-EC, apropriado para dispositivos com restrições de armazenamento e de comunicação em rede. O esquema de estruturação SbC-EC é fundamentado em duas propriedades com o objetivo de reduzir o espaço necessário para armazenar entradas de log: (*i*) Separação por Categoria (*Split by Category*, SbC) e (*ii*) Compactação de Entradas de log (*Entry Compaction*, EC).

Apresentamos, também, um novo esquema de log seguro baseado nessa estruturação, o SbC-EC MAC. O esquema SbC-EC MAC utiliza as primitivas de criptografia simétrica e o esquema de autenticação *FssAgg*, criado por Ma e Tsudik [59], para criar as tags de autenticação do log de dados. Descrevemos as funções presentes no SbC-EC MAC, como é realizada a inicialização e verificação dos arquivos de log, a concatenação e a compactação das entradas de log. Além disso, discutimos algumas considerações sobre a segurança do esquema, onde relatamos características como a *forward-secrecy*, a *forward secure stream integrity* e a suscetibilidade ao ataque de *delayed detection*, causado por um usuário privilegiado. Por fim, avaliamos a nossa proposta em duas etapas. Primeiramente, demonstramos uma avaliação do esquema SbC-EC MAC em relação ao ganho de armazenamento, comparando-o a outros dois esquemas simétricos existentes na literatura. Em seguida, realizamos

uma simulação de um cenário experimental, onde pudemos observar a relevância das duas propriedades da estruturação proposta em relação ao ganho de armazenamento no dispositivo testado.

Como trabalhos futuros, consideramos adicionar outras características ao esquema de log seguro, levando em conta a criticidade da informação do log e desempenho computacional necessário para proteção dos arquivos de log. Assim como também elaborar, baseado na estruturação SbC-EC, um esquema de log seguro para o domínio da criptografia de chave-pública.

4 PROTOCOLO DE AUTENTICAÇÃO ATRASADA *INF*-TESLA PARA COMUNICAÇÃO *MULTICAST*

Neste Capítulo do trabalho, apresentamos a proposta de autenticação de evidências digitais em trânsito. A coleta de evidências digitais é um dos estágios mais críticos do processo de uma investigação forense, uma vez que o dado precisa ser adquirido e preservado de forma apropriada. Para a preservação das evidências, é necessário assegurar a sua autenticação e integridade, a fim de ser utilizado como prova legal em futuras investigações.

Dentre as dificuldades técnicas das aplicações de monitoramento e controle, como as encontradas nos Sistemas Ciberfísicos, podemos destacar a de realização de uma análise "ao vivo" (*live analysis*) e a escalabilidade dos métodos forenses. Registrar todas as ações de controle e dados de medições em um banco de dados central pode ser viável em escala industrial. No entanto, em grandes infraestruturas, como as da rede de energia elétrica de energia, isso resultaria em uma sobrecarga excessiva na comunicação e tornaria problemática a transmissão de mensagens críticas de monitoramento e controle [28].

Os dispositivos *Phasor Measurement Units* (PMUs) desempenham um papel significativo na análise forense de perturbações da rede elétrica. Os dados gerados nesses dispositivos, e transmitidos à diversas aplicações de monitoramento, podem fornecer informações valiosas sobre as falhas e imagens em tempo real da rede, de antes e depois das perturbações [28]. Neste capítulo, investigamos o desafio da autenticação desses dados em uma comunicação *multicast* no cenário das Redes Elétricas Inteligentes, no qual a infraestrutura de comunicação deve atender aos requisitos temporais críticos das aplicações, assim como uma alta taxa de envio de

dados, e atender requisitos de segurança, como criptografia "leve", e a garantia de integridade e de origem das informações.

4.1 Contextualização do Problema

As redes de energia elétrica são consideradas um dos maiores Sistemas Ciber-Físicos da atualidade. Ao longo dos últimos anos, a infraestrutura das redes elétricas vem sendo transformada a partir da concepção e desenvolvimento de aplicações avançadas para registro, visualização e controle em tempo real dos eventos que acontecem na rede [74]. Como parte desses esforços, a instalação dos dispositivos *Phasor Measurement Units* (PMUs) é um exemplo de investimento na infraestrutura da rede para a finalidade de controle. Antes da instalação dos PMUs, a falta de visibilidade de áreas de longa distância foi um dos fatores que impediram, por exemplo, a identificação de falhas no nordeste norte-americano e os apagões na Itália, ambos durante o ano de 2003 [100, 101].

Os dispositivos PMUs realizam medições de dados críticos da rede elétrica, tais como tensão, corrente e frequência, sincronizados no tempo. Essas medições são conhecidas como "sincrofasores". Os sincrofasores originados de múltiplos PMUs são utilizados como evidências a fim de auxiliar a análise em tempo real ou *post-mortem*, planejamento, ações corretivas e controle automatizado para a segurança e resiliência da rede elétrica. Em geral, os PMUs são posicionados em locais específicos da infraestrutura de forma a proporcionar boa visibilidade sobre grandes áreas da rede. Atualmente, as redes de comunicação de alta velocidade, juntamente aos PMUs, estão sendo utilizadas como infraestrutura de suporte para aplicações *Wide Area Monitoring System* (WAMS), a fim de fornecer consciência situacional ("*situational awareness*") do estado das redes elétricas situadas na América do Norte, China, Canadá, Brasil e em toda a Europa [74].

Ao longo dos anos, novos padrões de segurança e protocolos precisaram ser criados para que permitissem a interoperabilidade dos dispositivos PMUs [38]. Isso se deve ao fato de que determinados ataques à rede elétrica, entre eles a alteração maliciosa de dados de PMUs, podem resultar em decisões de controle imprecisas e, consequentemente, prejudiciais à segurança, confiabilidade e resiliência da rede elétrica. Idealmente, as informações geradas pelos PMUs necessitam de autenticação, garantia de integridade e, opcionalmente, de confidencialidade.

Os esquemas de autenticação utilizados nas Redes Elétricas Inteligentes devem ser capazes de oferecer suporte à comunicação *multicast* de maneira eficiente. O padrão IEC/TS 62351-6 [45] especifica o uso de assinaturas digitais e criptografia assimétrica. No entanto, a utilização dessas técnicas gera preocupações quanto ao custo e impacto no desempenho do microprocessador, dada a arquitetura de *hardware* dos PMUs. Ademais, essas preocupações tendem a crescer dentro de um cenário onde os dados são transmitidos em altas taxas. Outra solução, como a sugerida em IEC/TR 61850-90-5 [47], engloba a utilização de algoritmos de autenticação HMAC para assinar os sincrofasores. Porém, o compartilhamento de apenas uma chave de sessão através de um grupo *multicast* não garante a segurança adequada dos dados compartilhados e, além disso, essa solução apresenta problemas de escalabilidade. Esquemas de *Multiple-Time Signature* (MTS) também possibilitam a autenticação em comunicação *multicast*, porém geram assinaturas relativamente grandes e possuem requisitos restritivos relacionados à sincronização de tempo entre os dispositivos na rede [54]. Por sua vez, os esquemas *One-Time Signature* (OTS) – um subconjunto de MTS – apresentam desvantagens relacionadas à sobrecarga no armazenamento de memória e na comunicação, e ao gerenciamento complexo de chaves criptográficas [102].

A abordagem da autenticação atrasada de pacotes apresenta vantagens sobre os problemas anteriormente citados, oferecendo suporte ao fluxo de dados em co-

municação *multicast*, criptografia simétrica e leve, e detecção de dados corrompidos e ataques. Ela também permite escalabilidade nas soluções de autenticação e no gerenciamento de chaves, tolera perda de pacotes, e proporciona uma baixa sobrecarga nas comunicações e alta eficiência computacional. Embora alguns trabalhos prévios na literatura assumam, em geral, que a autenticação atrasada não é apropriada para aplicações que exigem transmissão de dados em tempo-real [54] [56], esse método ainda se mostra elegível para aplicações de monitoramento e controle que permitem margens relativamente maiores de atraso de pacotes (p.ex., *wide-area oscillation damping control application*) [112], e também para aplicações em tempo quase real ("*near real-time*"). Na literatura, é também sugerido que, para algumas classes de comandos, o receptor execute a ação de controle imediatamente após receber a mensagem e armazene uma informação de *roll-back*, caso a verificação da mensagem venha a falhar instantes depois [31]. Para maiores considerações sobre a elegibilidade da abordagem de autenticação atrasada, ver Seção 4.3.2.

A contribuição principal deste capítulo é a proposta do *inf*-TESLA, *Infinite Timed Efficient Stream Loss-tolerant Authentication*, um protocolo de autenticação atrasada para comunicação de dados em *multicast*. O nosso esquema é baseado no protocolo de autenticação TESLA [77], alterado de forma a comportar comunicações de longo prazo com altas taxas de envio de pacote, de maneira mais eficiente e escalável. Essas características tornam, portanto, o nosso esquema apropriado para o procedimento de coleta de dados de monitoramento nos Sistemas Ciberfísicos, como em uma aplicação WAMS para as Redes Elétricas Inteligentes.

O protocolo *inf*-TESLA utiliza o mecanismo *Dual Offset Key Chains* para geração das chaves criptográficas, a fim de permitir o mínimo atraso de autenticação dos pacotes e a redução do custo computacional associado à substituição de cadeias de chaves criptográficas. Descrevemos, também, dois modos de divulgação de chaves e demonstramos a sua segurança contra uma tentativa de ataque homem

no meio. Por fim, apresentamos uma avaliação comparativa entre a nossa abordagem e o protocolo original TESLA em um cenário de simulação, na qual o primeiro demostra uma redução no custo computacional exigido dos participantes envolvidos na comunicação.

4.2 Trabalhos Relacionados

A autenticação para comunicação *multicast* é um campo de pesquisa ativo nos últimos anos e tem sido utilizada em diversas aplicações. Em sistemas de energia elétrica, por exemplo, há uma grande necessidade desse mecanismo para monitoramento, proteção e divulgação de informações na rede [102]. Nesta seção, primeiro revisaremos todos os esquemas de autenticação em *multicast* baseados no protocolo TESLA e, em seguida, outros esquemas de autenticação em *multicast* também utilizados para sistemas de energia elétrica.

Para abordar o desafio da autenticação de dados em fluxo contínuo para múltiplos receptores em uma rede sujeita a perdas, o protocolo *Timed Efficient Stream Loss-tolerant Authentication* (TESLA) foi apresentado por Perrig *et al.* [77]. TESLA é um protocolo de autenticação para comunicação *broadcast* que estabelece a estratégia de divulgação atrasada das chaves criptográficas usadas na autenticação de mensagens anteriormente enviadas, e a retenção (*buffering*) de pacotes por parte do receptor. Ele é baseado no protocolo Guy Fawkes [6] e permite uma sincronização relaxada, em relação ao tempo, entre os emissores e receptores. TESLA suporta ambas taxas, fixa ou dinâmica, de pacotes, e apresenta escalabilidade e robustez em uma rede com perda de pacotes. Algumas vantagens do TESLA incluem baixo custo de computação, baixa sobrecarga por pacote na comunicação, tolerância à perda de pacotes arbitrários, fluxo unidirecional de dados, e alto grau de confiabilidade na autenticação dos dados. Os trabalhos posteriores desses autores proporam algumas

modificações e melhorias para o TESLA, entre elas um modo no qual os emissores realizam retenção de pacotes enquanto os receptores autenticam os pacotes conforme são recebidos. Eles também sugerem soluções para uma redução nas sobrecargas geradas pelo protocolo e maior robustez a ataques de negação de serviço [78].

Studer et al. descreve o protocolo TESLA++ [96], uma versão modificada do TESLA resiliente a ataques de negação de serviço baseados em memória. Os autores combinam o TESLA++ com assinaturas digitais ECDSA (*Elliptic Curve Digital Signature Algorithm*) para construir um arcabouço de autenticação para Redes Ad Hoc Veiculares.

O protocolo μTESLA [79] é uma adaptação do TESLA que o torna prático para a autenticação em comunicação *broadcast* em ambientes com recursos restritos, tais como as Redes de Sensores Sem Fio. Algumas das adaptações apresentadas sugerem o uso apenas de mecanismos de criptografia simétrica, frequência menor em relação à divulgação das chaves criptográficas e restrição do número de emissores autenticados. Liu e Ning [57] reduziram a sobrecarga necessária para a divulgação dos *commitments* de cadeia de chaves e apresentam estratégias para combater os ataques de negação de serviço. Seu protocolo Multinível μTESLA considera diferentes níveis de cadeias de chaves a fim de abrangir todo o ciclo de vida de um sensor.

Falk e Fries [31] propõem melhorias para o TESLA, tais como as cadeias de *hash* hierárquicas para evitar os frequentes usos de assinaturas digitais para estabelecer novas cadeias. Esta abordagem difere da nossa proposta uma vez que o nosso mecanismo de gerenciamento de chaves apresenta uma maneira de autenticar todos as fluxos de cadeias de *hash* posteriores a partir do estabelecimento de apenas duas cadeias durante o início da comunicação. Além disso, a abordagem hierárquica pode ser suscetível aos ataques de negação de serviço, através da interferência no

processo de estabelecimento de novas cadeias de *hash* por parte dos receptores.

Além dos métodos baseados em TESLA, podemos destacar a família *Multiple-Time Signature* (MTS), e o subconjunto *One-Time Signature* (OTS), que ganharam popularidade nos últimos anos e são aplicáveis à autenticação em *multicast* e também à aplicações WAMS. O autor em [76] descreve um protocolo OTS para autenticações em *broadcast* com base no protocolo BiBa. O BiBa utiliza funções *one-way* sem *trapdoors* e explora o paradoxo do aniversário para oferecer segurança e eficiência na verificação de assinaturas. Suas desvantagens incluem uma chave pública de tamanho grande e uma alta sobrecarga no processo de geração da assinatura.

O protocolo HORS [87] é descrito por Reyzin *et al.* como um esquema OTS que possui agilidade durante os processos de geração e de verificação das assinaturas. Ele utiliza uma função de *hash* para obter subconjuntos aleatórios para a mensagem assinada e, também, para verificá-la, no entanto, o protocolo apresenta desvantagens em relação às frequentes distribuições da chave pública. O TV-HORS [103], um protocolo de autenticação em multicast, utiliza cadeias de *hash* para conectar vários pares de chaves a fim de autenticar simultaneamente vários pacotes. Além disso, o autor sugere assinar os primeiros l bits do *hash* da mensagem, mostrando ser mais eficiente na geração de assinatura do que seu antecessor. Como desvantagem, o TV-HORS necessita de uma chave pública de tamanho que chega a 10 Kbytes.

O trabalho em [56] descreve o TSV, um protocolo MTS que gera assinaturas menores que a do protocolo HORS e tem requisitos menos restritos de armazenamento em detrimento, por outro lado, do aumento do custo de computação na geração e verificação das assinaturas. Ele usa a abordagem de gerenciamento de chaves chamado "*nonuniform chain traversal*", a qual foi demonstrada, posteriormente, ser potencialmente insegura perante aos ataques de negação de serviço nos receptores. O TSV+ [54], um protocolo de autenticação em *multicast* a partir de

uma versão mais segura do TSV, usa a abordagem "*uniform chain traversal*" para gerenciamento de chaves e suporta múltiplas assinaturas dentro de um determinado período ("*epoch*"). O protocolo SCU [99] é um esquema MTS projetado para atualização segura de código em Redes de Sensores Sem-Fio e a sua variante SCU+ [54] o adaptou para ser um esquema de autenticação em multicast para os sistemas de energia, utilizando, também, a abordagem *uniform chain traversal*.

Além dos trabalhos citados acima, uma série de referências refletem os esforços da indústria, do governo norte-americano e das organizações internacionais em relação à segurança de dados na rede elétrica. A iniciativa *North American Synchrophasor Initiative* (NASPI) estabelece um grupo de compartilhamento de informações voltado para a indústria, que busca soluções para questões estratégicas e entraves técnicos, operacionais, institucionais, desenvolvimento de normas, entre outros [69]. O padrão IEEE C37.118 define a medição dos sincrofasores, e fornece um método para quantificar essas medições e as especificações para testes de qualidade. Por não incluir mecanismos de autenticação, este protocolo é vulnerável a ataques de falsificação ("*spoofing*"). O padrão IEC/TR 61850-90-5 [47] apresenta uma maneira para troca de dados sincrofasores entre os dispositivos PMU, PDCs e as aplicações do centro de controle, enquanto que o padrão IEC/TS 62351 [45] aborda os requisitos de segurança da informações necessários e sugere soluções, como a utilização de funções HMAC, para autenticação dos dados sincrofasores.

4.3 Características do Cenário

O cenário adotado para esta parte do trabalho é determinado pelo crescente uso de aplicações de monitoramento e controle baseadas em medições de sincrofasores, onde são conhecidos os limites tolerados de atraso de pacotes na rede [17]. No entanto, ao contrário de infraestruturas tradicionais de tecnologias da informação e

comunicação, como as dos bancos e das telecomunicações, o contexto dos Sistemas Ciberfísicos compreende restrições significativas aos recursos computacionais e de comunicação disponíveis, resultando na priorização do uso de soluções "leves" de segurança.

4.3.1 Modelo do Sistema

O modelo do sistema baseia-se na infraestrutura de rede utilizada nos sistemas de monitoramente de grandes áreas (WAMS) nas Redes Elétricas Inteligentes (detalhado na seção 2.2). Os componentes principais desse modelo são: a máquina emissora S (PMU), as máquinas intermediárias $I_1, I_2, ..., I_L$ (PDCs), e as máquinas receptoras $R_1, R_2, ..., R_m$ (Super-PDCs), conforme mostrados na Figura 4.1.

Cada *link* de comunicação compreende um PMU S, capaz de enviar pacotes em *multicast* para m Super-PDCs R_k, onde $1 \leq k \leq m$, seguindo o modelo publicador/assinante. O PMU S envia pacotes com dados sincrofasores, com assinatura de tempo, a uma taxa de 10 a 120 pacotes por segundo. A rede possui n PDCs intermediários entre S e R_k, sendo $n > 0$. Os PDCs agregam e alinham cronologicamente os sincrofasores recebidos, reempacotam e retransmitem esses dados para o próximo nó, sendo esse outro PDC ou Super PDC. Um pacote retransmitido pelo PDC é, comumente, maior do que os enviados pelos PMUs, pois seu quadro de dados pode conter diversos valores enviados por diferentes PMUs dentro de um intervalo de tempo.

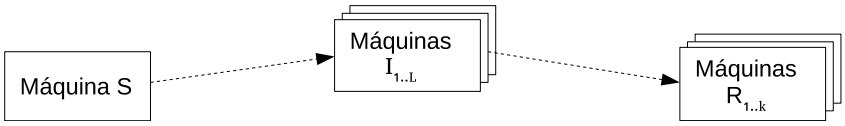

Figura 4.1: Modelo do Sistema.

A transmissão está sujeita a perda de pacotes, uma vez que a necessidade de altas taxas de envio de dados inibe o uso da verificação de recebimento de pacotes, nos dois sentidos da comunicação. Quando as medições são perdidas na rede, ou demoram além do intervalo de tempo esperado para chegar, o PDC pode, a fim de manter o fluxo de comunicação dos dados e com a devida indicação, interpolar as medições de sincrofasores e preencher seu quadro de dados com esses valores interpolados. O Super-PDC R_k, destinatário final do pacote, verifica a autenticidade das valores recebidos e os repassa para a devida aplicação que irá consumir esses dados.

4.3.2 Latência de Comunicação

Inerente à arquitetura de comunicação de múltiplos nós da rede PMU/PDC, existe uma latência, ou atraso, de comunicação entre os dispositivos. Esse atraso se dá em função da topologia, do volume de tráfego e da capacidade dos componentes, e afeta tanto os sistemas de controle quanto a autenticação das medições enviadas (assumindo a solução de autenticação atrasada apresentada neste capítulo). O atraso da rede, portanto, estabelece um limite inferior sobre o atraso de autenticação. Embora seja possível que o atraso de autenticação se adaptasse ao atraso da rede e, consequentemente, fosse dinâmico, assumiremos, para fins de referências nas notações, que o atraso da rede possui limite superior igual a d_{NMax}.

Existem diferentes aplicações WAMS que consomem dados sincrofasores e possuem diferentes requisitos em relação ao intervalo de atraso aceitável para a qualidade do serviço. Como exemplo, aplicações como o *Situational Awareness Dashboard*, *Small-Signal Stability Monitoring*, e *Voltage Stability Monitoring/Assessment* aceitam até 500 milissegundos na latência da comunicação, enquanto outras aplicações como *Long-term Stability Control*, *State Estimation*, e *Disturbance Analy-*

sis Compliance podem suportar até 1000 milissegundos. Para a lista completa de aplicações e seus requisitos de atraso de pacotes, ver Tuffner [98].

O trabalho de Zhu *et al.* [112] apresenta simulações da latência de rede para aplicações de monitoramento na rede de transmissão sueca e obteve resultados dentro do intervalo de 150 a 220 milissegundos. Para aplicações de controle centralizado, a latência calculada se mostrava bem abaixo de 500 ms. O tempo de um *round-trip time* (RTT) para qualquer hospedeiro na internet intra-planetária raramente excede os 500 ms, se o roteamento se mantiver estável [81]. Do ponto de vista da autenticação atrasada, o atraso mínimo para uma confirmação segura da autenticação dos pacotes por parte de R_k é igual a latência da rede, ou seja d_{NMax}, ou ainda $RTT/2$. Como desejamos que o atraso para a confirmação de autenticação seja o mínimo possível, podemos assumir o dado valor de forma aproximada. Ainda assim, os protocolos de autenticação atrasada são capazes de atender os requisitos para as aplicações citadas anteriormente.

Pelo documento C37.118 [43], é requerido que um PMU possua uma fonte de sincronização de tempo UTC ("*Coordinated Universal Time*"). Para isso, podem ser utilizados o sistema GPS ("*Global Positioning System*") ou um relógio local que use protocolos padrão, como o *Inter-range instrumentation group time code B* (IRIG-B) ou o *Precision Time Protocol* (PTP) [44]. Esses protocolos podem ser usados para estimar e minimizar os atrasos de relógio entre os dispositivos da rede. Nosso esquema de autenticação atrasada requer que S e R_k sejam, no mínimo, fracamente sincronizados no tempo. Ou seja, R_k deve ser capaz de calcular um limite superior δ_{tMax} do máximo erro de sincronização do seu relógio em relação a S. Assumimos que δ_{tMax} é mínimo, dada a atuação dos protocolos de sincronização de tempo citados.

4.3.3 Considerações de Segurança

A integridade e autenticidade são as principais preocupações de segurança em relação aos dados sincrofasores. Assumimos o mesmo modelo de ataque utilizado em [103], onde o adversário não é limitado pela largura de banda da rede de comunicação e tem total controle para descartar, reenviar, capturar e manipular pacotes. Assumimos, também, que os ataques do tipo homem no meio ("*man-in-the-middle*") são possíveis, a fim de manipular os valores dos dados ou simular um PMU legítimo na rede. Espera-se que o adversário possua recursos computacionais relativamente poderosos, porém não ilimitados, e que ele não possa, com uma probabilidade não negligenciável, inverter uma função pseudo-aleatória.

Outra consideração a ser feita, tendo em conta uma solução de autenticação atrasada, é em relação ao tempo de exposição à ameaça que o receptor está sujeito. Cada receptor R_k é capaz de autenticar o conteúdo e a origem dos dados enviados após um atraso mínimo de d_{NMax}. No entanto, se um pacote falha a autenticação no tempo t, logo um ataque poderá estar ativo e não detectado pelo menos desde $t - d_{NMax}$. É trivial concluir que, quanto maior for o atraso de autenticação, maior a janela de exposição à ameaças.

As primitivas criptográficas usadas ao longo deste capítulo são as seguintes:

- *Função de Hash criptográfico* – $H(M)$ opera em uma mensagem M de tamanho arbitrário, retornando $h = H(M)$. $H()$ pode ser implementado, por exemplo, pela família de algoritmos *Secure Hash Algorithm* [67];

- *Message Authentication Code* – $MAC(K, M)$ fornece uma *tag* que verifica a autenticidade e integridade da mensagem M dada a chave compartilhada K. $HMAC(K, M)$ é uma construção específica que inclui uma função de *hash*

criptográfico para a criação da tag de autenticação [10];

- *Cadeia de Hash* – $H^n(M)$ denota n sucessivas aplicações de uma função de *hash* criptográfico $H()$ em uma mensagem M.

Para o uso de comunicação *multicast*, o padrão IEC/TR 61850-90-5 requer que um Centro de Distribuição de Chaves (KDC - *Key Distribution Center*) seja utilizado, a fim de gerenciar as chaves simétricas entre S e R_k. Assumimos que cada S é o seu próprio KDC, o que também é aceito pelo padrão citado. Além disso, como nosso esquema exige que S prove sua identidade para R_k uma vez durante a etapa de inicialização da comunicação entre eles, é necessário que R_k possa validar as assinaturas digitais enviadas por S e manter uma cópia de seu certificado de chave pública. Para tal efeito, supomos também que os dispositivos na rede possuem acesso a uma Infraestrutura de Chaves Públicas (ICP).

4.4 Protocolo Proposto

Nesta seção, propomos o protocolo *inf*-TESLA, baseado no esquema do TESLA, e o mecanismo de gerenciamento de chaves *Dual Offset Key Chains*. Primeiramente, revisamos o protocolo TESLA para, em seguida, apresentar a nossa proposta.

4.4.1 Protocolo TESLA

O *Timed Efficient Stream Loss-tolerant Authentication* (TESLA) [77] [78] [80] [81] é um protocolo de autenticação em *broadcast* que oferece baixa sobrecarga de computação e na comunicação, tolerância à perda de pacotes, e necessita apenas de

uma fraca sincronização de tempo entre o emissor e os receptores. A sincronização é chamada "fraca" porque não é necessário que a diferença de sincronização entre os dois seja a mínima possível. O único requisito é que o receptor saiba o limite superior do erro de sincronização entre ele e o emissor, o qual pode ser calculado através de um algoritmo simples, apresentado nos trabalhos de referência do protocolo.

O protocolo TESLA baseia-se na divulgação atrasada das chaves simétricas e, como consequência, o receptor deve reter as mensagens recebidas antes de poder autenticá-las. As chaves são geradas a partir de uma cadeia de sentido único ("*one-way chain*") e são utilizadas e divulgadas na ordem inversa da sua geração. Durante a etapa de configuração inicial da comunicação, o emissor deve, primeiramente, definir n como o índice do primeiro elemento da cadeia K_n. Para gerar a cadeia de chaves, o emissor escolhe aleatoriamente um número para K_n e, utilizando uma função pseudo-aleatória f, ele constrói a função de sentido único $F : F(k) = f_k(0)$. Logo, o emissor é capaz de gerar recursivamente todas as chaves subsequentes da cadeia utilizando $K_i = F(K_{i+1})$. Através dessa fórmula, o último elemento da cadeia é representado por $K_0 = F^n(K_n)$, e todos os outros elementos podem ser calculados utilizando $K_i = F^{n-i}(K_n)$.

Cada elemento K_i aparenta ser pseudo-aleatório e um adversário é incapaz de inverter F e calcular qualquer K_j, onde $j > i$. No caso da perda de um pacote contendo K_i, um receptor poderá calcular K_i a partir de qualquer pacote subsequente contendo K_j, uma vez que $K_j = F^{i-j}(K_i)$. Dessa forma, o protocolo TESLA demonstra tolerância à perda esporádica de pacotes.

O esquema de autenticação de fluxo do TESLA é segura, desde que a seguinte condição de segurança seja atendida: um pacote de dados P_i chegou *em segurança*, se o receptor pode decidir de forma inequívoca, com base na sua sincronização no tempo e no erro máximo tolerável de sincronização no tempo, que o emissor ainda

não enviou o pacote P_j contendo a chave criptográfica correspondente [77].

O protocolo também suporta dois tipos de comunicação: as com taxa fixa de pacotes, e as com taxa dinâmica. Para as comunicações com taxa fixa, o emissor divulga a chave K_i, correspondente ao pacote P_i, em pacote subsequente P_{i+d}, onde d é o parâmetro de atraso configurado e anunciado pelo emissor durante a etapa de configuração inicial da comunicação. O emissor determina o atraso d de acordo com a taxa de pacote r, o erro máximo tolerável de sincronização δ_{tMax}, e o atraso máximo tolerável da rede d_{NMax}, sendo $d = \lceil(\delta_{tMax} + d_{NMax})r\rceil$. Em comunicações com taxa fixa de pacote, o esquema pode alcançar atrasos de autenticação menores.

Para as comunicações com taxa dinâmica, o emissor seleciona uma chave por cada intervalo de tempo T_{int}. Cada chave é atribuída a um intervalo uniforme de tempo, T_0, T_1 ,..., T_n, isto é, a chave K_i será ativada durante o período T_i. O emissor utiliza a mesma chave K_i para calcular o MAC de todos os pacotes que são enviados durante T_i. Da mesma forma, todos os pacotes enviados em T_i divulgam a chave $K_{i-d'}$. Neste caso, $d' = \lceil(\delta_{tMax} + d_{NMax})/T_{int}\rceil$. Nós utilizamos a designação d e d' para representar o parâmetro de atraso nas comunicações com taxa fixa e taxa dinâmica, respectivamente. O parâmetro d é referenciado em número de pacotes, enquanto o parâmetro d' é em unidades de intervalo de tempo.

Cada novo receptor (assinante, ou *subscriber*) que integra a comunicação em fluxo deverá, primeiramente, enviar uma requisição de sincronização ao emissor. Após receber a requisição, o emissor cria um pacote de sincronização autenticado que contém parâmetros como a informação do intervalo de tempo, o parâmetro d de divulgação das chaves, e também uma chave criptográfica divulgada – que é um compromisso ("*commitment*") da cadeia de chaves ativa. O emissor deve, então, assinar digitalmente este pacote de sincronização e enviá-lo ao novo integrante. Quando não há a necessidade de sincronização direta entre os participantes (por te-

rem disponível, por exemplo, um servidor de sincronização de tempo), ainda assim o compromisso da cadeia deve ser enviado, digitalmente assinado, ao receptor. Em geral, o compromisso é enviado, por conveniência, no pacote de sincronização. A partir da etapa de sincronização e envio do compromisso, o receptor estará apto a validar a autenticação da comunicação em fluxo.

4.4.2 Protocolo *inf*-TESLA

O *inf*-TESLA, uma abreviação para *infinite TESLA*, é um protocolo para autenticação em comunicação *multicast* baseado em TESLA, apropriado para utilização em comunicações com alta taxa de pacotes por um extenso período. Como no protocolo TESLA, o *inf*-TESLA também depende da robustez das funções de *hash* e de criptografia simétrica, e na divulgação atrasada das chaves criptográficas como um meio de autenticar as mensagens enviadas pelo emissor. Além disso, o *inf*-TESLA também necessita apenas de uma fraca sincronização no tempo entre o emissor e os receptores, para ambos os tipos de comunicações, com taxa fixa ou taxa dinâmica de pacotes.

Ao operar em comunicações com taxa fixa de pacotes, não há necessidade de configurar intervalos de tempo que contém, durante toda a duração do intervalo, apenas uma chave criptográfica ativa para realizar o MAC das mensagens. Cada chave é, portanto, utilizada apenas uma vez para autenticar a mensagem atual e divulgada após d pacotes. Apesar desse modo de operação poder minimizar o período de tempo necessário para autenticar mensagens anteriores, ele apresenta uma desvantagem de rapidamente consumir a cadeia de chaves criptográficas, dependendo da frequência de envio dos pacotes.

Uma vez que a geração de chaves criptográficas é feita a partir da suces-

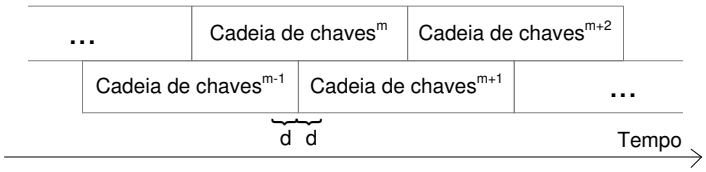

Figura 4.2: Uma ilustração do funcionamento no tempo das cadeias de chaves usadas no protocolo *inf*-TESLA.

siva aplicação da função *hash* para construir uma cadeia de chaves independente, toda vez que uma cadeia de chaves chega ao fim – significando que ela foi totalmente consumida durante o processo de autenticação das mensagens – o emissor deve automaticamente gerar, armazenar e utilizar uma nova cadeia de chaves para substituir a antiga. No protocolo original TESLA, o emissor deve restabelecer um novo compromisso de cadeia aos receptores após o término da atual cadeia de chaves, causando, portanto, uma sobrecarga na rede de comunicação e, também, uma sobrecarga computacional para todos os participantes. Isso se deve ao fato de que o compromisso precisa ser assinado digitalmente, através de mecanismos de criptografia assimétrica. Essa sobrecarga, portanto, não pode ser negligenciada.

O protocolo *inf*-TESLA aborda esse problema utilizando chaves criptográficas geradas pelo mecanismo *Dual Offset Key Chains* (DOKC), ou, em português, Dupla Desalinhada de Cadeias de Chaves. A partir das chaves geradas por este mecanismo, o *inf*-TESLA utiliza um par de chaves para cada mensagem a ser autenticada e garante a continuidade do processo de autenticação em *multicast* sem a necessidade de, periodicamente, assinar e enviar um pacote com o compromisso da nova cadeia de chaves aos receptores. O mecanismo DOKC sempre disponibiliza um par ativo de cadeias de chaves em um determinado tempo. Como princípio fundamental, o nosso protocolo de autenticação em *multicast* sempre usará uma cadeia m como suporte durante o processo de substituição da cadeia de chaves $m-1$ pela cadeia $m+1$. A Figura 4.2 ilustra esse processo de substituição de cadeias de chaves. Uma descrição detalhada do mecanismo *Dual Offset Key Chains* é apresentada na

Seção 4.4.4.

Neste trabalho, iremos focar na comunicação com taxa fixa de pacotes por potencialmente adequar-se melhor ao cenário que abordamos e pela questão de notação. De toda forma, o protocolo *inf*-TESLA é compatível com ambas taxas fixa e dinâmica de pacotes.

4.4.3 Configuração Inicial da Comunicação

A etapa de configuração inicial da comunicação é similar ao do protocolo TESLA. Antes da comunicação em fluxo iniciar, o emissor S primeiramente determina algumas informações essenciais sobre as condições da rede de comunicação (como a latência máxima da rede, d_{NMax}), sincronização no tempo (como o erro máximo tolerável de sincronização, δ_{tMax}), e escolhe a dupla de chaves que serão o compromisso das cadeias. Ainda nesta etapa, S define o parâmetro de atraso d (apresentado na Seção 4.4.1), o qual irá basear a decisão do receptor R_k em aceitar, ou não, um pacote enviado por S.

Para integrar um novo receptor à comunicação em fluxo, S constrói e envia o pacote de sincronização para o novo assinante do fluxo, como resposta à requisição de sincronização do novo receptor. Para as comunicações com taxa de pacotes dinâmica, o pacote de sincronização contém os seguintes dados [78]: o tempo inicial de um intervalo específico T_j juntamente com a sua identificação I_j, a duração do intervalo T_{int}, o parâmetro de atraso d', um compromisso para a cadeia de chaves K_i^m e a cadeia de chaves K_i^{m+1} ($i < j - d'$, onde j é o índice do intervalo atual). Para as comunicações com taxa fixa, sejam j_1 e j_2 as chaves atuais originadas das cadeias m e $m+1$ respectivamente, o pacote de sincronização contém: o parâmetro de atraso d, e o compromisso para as cadeias de chaves $K_{i_1}^m$ e $K_{i_2}^{m+1}$ ($i_1 < j_1 - d$ e

$i_2 < j_2 - d$).

4.4.4 Mecanismo *Dual Offset Key Chains*

O mecanismo *Dual Offset Key Chains* (DOKC), ou em português Dupla Desalinhada de Cadeias de Chaves, permite ao esquema de autenticação em *multicast*, proposto nesta seção, a continuidade da autenticação em fluxo sem a necessidade do restabelecimento periódico do compromisso entre S e R_k. Duas cadeias de chaves são utilizadas simultaneamente pelo mecanismo para prover as chaves criptográficas de autenticação. Essas cadeias são ditas desalinhadas pois elas serão iniciadas e terminadas em tempos distintos. Dado um tempo t, seus elementos ativos sempre terão índices diferentes (a exceção para isso é na primeira inicialização das cadeias, ver explicação mais abaixo). Há, portanto, sempre duas cadeias de chaves existentes e, de cada cadeia, uma chave não utilizada e disponível para o processo de autenticação.

Para a construção das duas cadeias de chaves, primeiramente o emissor escolhe n, o total de elementos em uma única cadeia. Seja l^m o número atual de elementos restantes na cadeia de chaves m, pois assumimos que todas as chaves criadas são apagadas logo após serem usadas no processo de autenticação. Seja M o máximo de memória disponível para o armazenamento das chaves, e que M é suficientemente grande para armazenar as duas cadeias de chaves, m e $m+1$, em qualquer tempo. O valor de n deve ser escolhido de acordo com as seguintes restrições:

1. $n \geq l^{m-1} + 2(d+1)$
2. $n \leq \frac{M}{2} + d$

A primeira restrição define um valor mínimo para n, que é o tamanho inicial

mínimo de uma cadeia de chaves qualquer. Durante a criação das duas primeiras cadeias, consideramos $l^{m-1} = 0$. A segunda restrição define o número máximo de elementos em uma cadeia. Se uma cadeia m não atender a esse limite superior, a cadeia de chaves $m+1$ não poderá ser armazenada de forma suficiente para satisfazer a Condição de Segurança-2, para o procedimento de substituição de cadeias de chaves (ver Seção 4.4.6). Na prática, pode não ser viável para S calcular uma cadeia inteira de chaves durante do intervalo de tempo entre o envio de dois pacotes. Sendo assim, S pode calcular e armazenar a cadeia $m+1$ bem antes do término da cadeia $m-1$.

Um pacote P_j enviado por S é formado pelos seguintes dados: $P_j = \{M_j, i_1, i_2, K^m_{i_1-d}, K^{m+1}_{i_2-d}, MAC(K^m_{i_1}||K^{m+1}_{i_2}, M_j)\}$. Todo pacote contém a mensagem atual M_j, o índice do elemento atual de cada cadeia i_1 e i_2, as chaves de autenticação divulgadas $K^m_{i_1-d}$ e $K^{m+1}_{i_2-d}$ (ver discussão na Seção 4.4.5), e a tag de autenticação calculada através de uma operação MAC que utiliza a concatenação das chaves atuais de cada cadeia. Em particular, durante o início da utilização de uma nova cadeia de chaves $m+1$, a notação $K^{m+1}_{i_2-d}$ refere-se às últimas chaves da cadeia $m-1$.

4.4.5 Divulgação de Chaves

O protocolo *inf*-TESLA oferece dois modos de operação para a divulgação das chaves: *2-chaves* e *Alternado*. No modo *2-chaves* – ou modo padrão, descrito anteriormente – cada pacote P_j divulga duas chaves de autenticação, uma de cada cadeia de chaves, utilizadas na autenticação da mesma mensagem M_i. Portanto, o pacote P_j contém as seguintes informações, $P_j \rightarrow K^m_{i_1-d}, K^{m+1}_{i_2-d}$.

O modo *Alternado* divulga uma chave por pacote, selecionada alternadamente entre as duas cadeias Formalmente, dois pacotes consecutivos terão as seguintes informações de chaves, $P_j \rightarrow K^m_{i_1-d}$ e $P_{j+1} \rightarrow K^{m+1}_{i_2+1-d}$, onde os índices $i_1 - d$ e $i_2 - d$

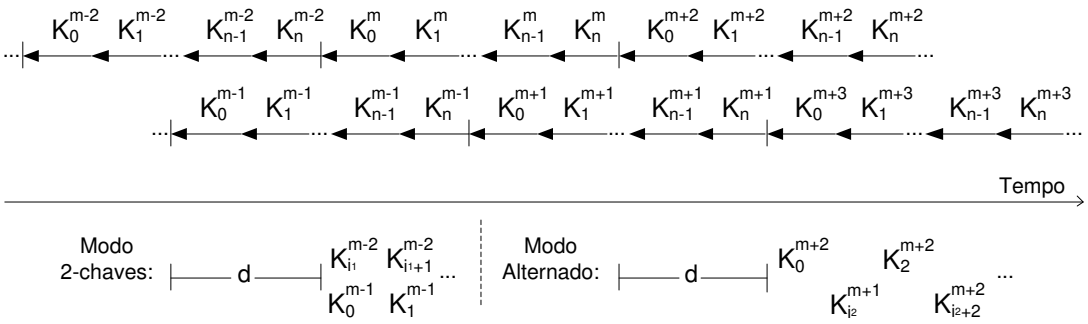

Figura 4.3: Os dois modos de divulgação de chaves: *2-chaves* e *Alternado*.

correspondem aos índices das mesmas chaves que seriam divulgadas no modo de divulgação *2-chaves*. A Figura 4.3 mostra as cadeias de chaves no tempo e os dois modos de divulgação de chaves. Na Seção 4.5, apresentamos uma comparação mais detalhada entre estes dois modos em relação à sobrecarga na comunicação, o custo computacional e o atraso de autenticação.

O parâmetro de atraso d, que indica o atraso na divulgação das chaves, é diretamente afetado pelo atraso máximo tolerável da rede, d_{NMax}, de modo que cada receptor R_k irá apresentar valores de atraso distintos. O emissor S, portanto, deve definir d como sendo o maior atraso previsto, a fim de atender à Condição de Segurança-1 (ver Seção 4.4.6).

4.4.6 Segurança do *inf*-TESLA

A segurança da cadeia de chaves baseia-se na primitiva criptográfica amplamente utilizada: a cadeia de sentido único ("*one-way chain*"). A cadeias de sentido único foram utilizados pela primeira vez por Lamport para uma solução de *one-time password* [53] e tem servido há diversas outras aplicações na literatura.

Condição de Segurança-1. É referente à decisão do receptor de autenticar, ou

não, um pacote que chegou – também usada no protocolo TESLA. Essa condição estabelece que: Um pacote de dados P_i é recebido de forma segura, se o receptor pode claramente decidir, baseado nas suas informações de sincronização de tempo e o erro máximo de sincronização entre ele e o emissor, que o emissor ainda não enviou o pacote P_j com a chave divulgada correspondente.

Condição de Segurança-2. É referente ao procedimento de substituição de cadeias de chaves do protocolo *inf*-TESLA. Essa condição estabelece que duas cadeias não podem ser substituídas dentro de um intervalo de tempo d/r (ou dentro da contagem de d pacotes). Se isso acontecer, o receptor deve recusar os pacotes seguintes e pedir ressincronização da comunicação com o emissor. Esta condição do protocolo garante a inviolabilidade de autenticação e deve ser observada a todo momento pelo receptor. O receptor é o único responsável por monitorar o procedimento de substuição de cadeias de chaves, e aceitar, ou rejeitar, a nova cadeia.

Ataque Homem no Meio. Na Figura 4.4, é apresentado um exemplo de uma tentativa de ataque do tipo homem no meio no esquema de autenticação proposto, e a importância da Condição de Segurança-2 na detecção do ataque. Para esse exemplo, consideramos $d = 9$, como o número mínimo de pacotes que o emissor precisa esperar para a divulgação de uma chave, o índice do último elemento $n = 50$ para todas as cadeias de chaves, e o símbolo "asterisco" indica um item malicioso inserido pelo atacante. Os pacotes são apresentados sem índices "i" para uma apresentação mais limpa.

Primeiramente, é apresentado como um ataque homem no meio pode obter sucesso em um esquema de autenticação utilizando apenas uma cadeia de chaves, sem compromissos autenticados, da seguinte forma: O atacante passa a escutar a comunicação e testa toda chave divulgada. A partir do momento em que é detectada uma mudança na cadeia de chaves utilizada para fazer a autenticação das mensagens,

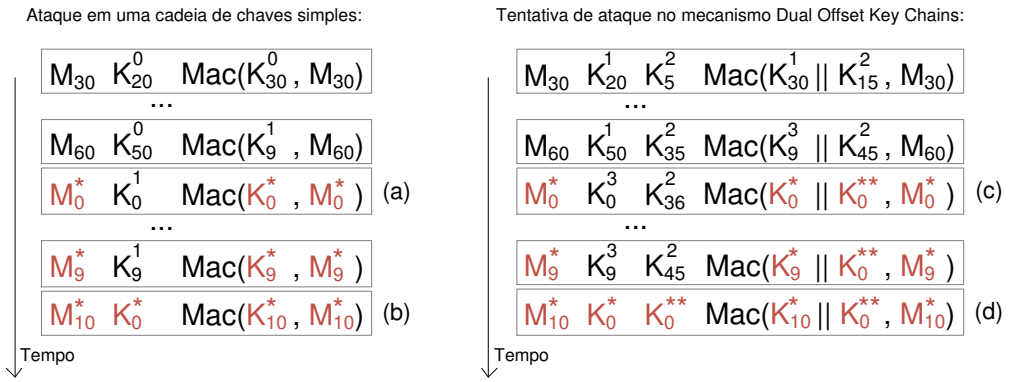

Figura 4.4: Exemplo de um ataque homem no meio em um esquema com uma cadeia de chaves (sem compromisso autenticado) e no esquema de autenticação do *inf*-TESLA.

o atacante insere M_0^* como a primeira mensagem manipulada e a autentica utilizando a função MAC, ver *(a)*. A chave utilizada nesse passo é o primeiro elemento K_0^* de uma cadeia de chaves forjada pelo atacante. O atacante continua manipulando as mensagens desejadas, e seus MACs, até a última chave autêntica do emissor ser divulgada. A partir deste ponto, o atacante é capaz de assumir o controle total da comunicação sem ser detectado, ver *(b)*.

Na segunda parte da Figura 4.4, a mesma tentativa de ataque homem no meio é realizada contra o nosso esquema de autenticação. O atacante é também capaz de detectar quando uma cadeia de chaves divulgadas chega ao fim e alterar as mensagens e os MACs nos pacotes. No entanto, quando ele tenta assumir o controle sobre a cadeia de chaves sendo divulgada através da inserção da chave forjada K_0^{**} na cadeia $m = 2$, isso indica ao receptor que houve uma violação da Condição de Segurança-2. Ou seja, o receptor detectou que houve a substituição das duas cadeias de chaves dentro do intervalo de pacotes especificado pelo parâmetro d.

Perda de pacotes consecutivos. Outra preocupação relacionada à segurança do protocolo é quantos pacotes consecutivos podem ser perdidos pelo receptor, sem

que isso seja necessariamente um ataque. A partir da Condição de Segurança-2, é necessário também que o receptor esteja ciente de que o limite máximo de pacotes perdidos consecutivamente deverá ser também o parâmetro d. Isso deve-se ao fato de que algum atacante pode provocar um ataque de negação de serviço no receptor e, então, retomar a comunicação com pleno controle das cadeias de chaves inseridas nos pacotes.

Se, por algum motivo, mais do que d pacotes consecutivos sejam perdidos na comunicação, o receptor deve assegurar que, após a normalização do recebimento, as chaves divulgadas nos pacotes seguintes são elementos autênticos de, pelo menos, uma das cadeias de chaves existentes anteriormente. Caso isso não seja verdade, o receptor não será capaz de validar a autenticidade de qualquer pacote recebido após as perdas consecutivas. A partir deste ponto, o receptor deve recusar este fluxo de pacotes e solicitar ao emissor uma nova sincronização, ou um compromisso autenticado das cadeias de chaves.

Perda das últimas chaves da cadeia. Outro problema na segurança do esquema pode ocorrer quando a última chave K_n na cadeia de chaves é perdida, o que pode causar a não validação da autenticidade de um pacote anterior P_n. Quando determinada chave K_i é perdida, essa pode ser calculada, a partir de qualquer chave seguinte da cadeia de chaves, através da função F (Section 4.4.1). No entanto, quando $i = n$, não há nenhuma chave subsequente pertencente a essa cadeia.

Este problema pode ser estendido para os últimos d elementos da cadeia de chaves (caso todas esses sejam perdidos), o que significa que até d pacotes poderão não ser devidamente autenticados e, como consequência, devem ser descartados pelo receptor. Para o modo *Alternado* de divulgação de chaves, o receptor descartaria $d + 1$ pacotes no pior caso. Esse problema a respeito da perda das últimas chaves da cadeia é uma vulnerabilidade também presente no protocolo TESLA original.

Além disso, ataques mais elaborados, como o descarte seletivo de pacotes, podem causar ainda mais atraso na autenticação dos pacotes. Por exemplo, no caso do modo de divulgação *Alternado*, um intruso pode induzir um descarte alternado de pacotes que impede o receptor de autenticar alguns pacotes sequenciais. Para mitigar esses tipos de ataques, o receptor deve definir um limite máximo para o número de pacotes não autenticados que ele deve ignorar, antes de ressincronizar com o emissor.

4.5 Avaliação

Para esta avaliação comparativa, verificaremos a sobrecarga na comunicação, o atraso de autenticação e custo computacional em uma comunicação de longo prazo para cada um dos seguintes esquemas: TESLA original, *inf*-TESLA *2-chaves* e *inf*-TESLA *Alternado*. Utilizamos a linguagem Python, e suas bibliotecas disponíveis, para simular os cenários aqui descritos.

A avaliação é realizada para um cenário de uma aplicação WAMS nas Redes Elétricas Inteligentes. Onde, devido às características do cenário de operação dos PMUs, estamos considerando apenas a comunicação com taxa fixa de pacotes. Além disso, as seguintes restrições para a simulação são assumidas:

- O *data frame* enviado pelo PMU possui tamanho de 60 *bytes*, de acordo com o padrão C37.118 [112]. O envio é feito sobre o protocolo da camada de transporte *User Datagram Protocol* (UDP).
- A função pseudo-aleatória e a função HMAC são implementadas pelo algoritmo HMAC-SHA-256-128 [33], que gera *tags* de tamanho trucado de 128 *bits*. O tamanho da chave HMAC também é de 128 *bits*.

- A assinatura digital é implementada pelo algoritmo *Elliptic Curve Digital Signature Algorithm* (ECDSA) sobre corpos finitos GF(p) de 256 *bits*. Apesar de o protocolo TESLA considerar o uso do algoritmo de assinaturas RSA, para os propósitos desta avaliação comparativa usaremos o algoritmo ECDSA. O tamanho das chaves e das assinaturas digitais é baseado no documento NIST SP 800-131A [8], o qual possui recomendações de algoritmos criptográficos e tamanhos de chaves criptográficas.

- O máximo número de chaves que podem ser armazenadas, em um dado momento, na memória cache de um dispositivo é de $n = 10.000$ chaves.

- A taxa de envio de pacotes pelo emissor é de 60 pacotes/segundo.

- O tempo simulado de operação é de 2 dias. Referências da literatura [54] estabeleceram um *baseline* de 1.024 cadeias de chaves para avaliar os esquemas de *One-Time Signatures*. No entanto, como o *inf*-TESLA gera aproximadamente 4 vezes o número de cadeias de chaves do que o TESLA precisa gerar para autenticar o mesmo número de pacotes, a avaliação comparativa será realizada baseada em um tempo simulado fixo, em vez do número total de cadeias de chaves utilizadas.

Sobrecarga na comunicação. A Tabela 4.1 indica as fórmulas utilizadas no cál-

Tabela 4.1: Cálculo da sobrecarga na comunicação.

	Fórmula
TESLA (fixo)	$C * (sKey + sSig) + P * (sKey + sMac)$
inf-TESLA *2-chaves*	$2 * sKey + sSig + P * (2 * sKey + sMac)$
inf-TESLA *Alternado*	$2 * sKey + sSig + P * (sKey + sMac)$
	2 dias de operação (MBytes)
TESLA (fixo)	331,825
inf-TESLA *2-chaves*	497,664
inf-TESLA *Alternado*	331,776

culo da sobrecarga na comunicação referente ao conteúdo de segurança introduzido por cada um dos três esquemas. Seja C o número de compromissos autenticados (assinados digitalmente), P o número total de pacotes transmitidos, e $sKey$, $sMac$ e $sSig$ serem o tamanho da chave criptográfica, o tamanho da *tag* MAC e o tamanho da *tag* de assinatura digital, respectivamente. O esquema *inf*-TESLA *2-chaves* apresenta a maior sobrecarga na comunicação devido às duas chaves divulgadas por pacote, enquanto TESLA e *inf*-TESLA *Alternado* apresentam uma pequena diferença, porém não relevante, de sobrecarga durante os dois dias de comunicação.

Custo computacional. Para calcular o custo computacional de cada esquema, utilizamos as fórmulas apresentadas na Tabela 4.2. O custo de processamento em ciclos por cada operação de *hash*, MAC, geração e verificação de assinatura digital é representado por $cHash$, $cMac$, $cSig$ e $cVer$ respectivamente. A partir do gráfico na Figura 4.5, podemos observar um custo computacional maior do emissor e do receptor que operam com o esquema TESLA em relação *inf*-TESLA. Isso se deve às constantes operações de geração e verificação de assinatura necessárias aos participantes que operam o protocolo TESLA, a fim de manter o fluxo da comunicação.

Ao longo de dois dias de simulação nessa configuração, o emissor operando o protocolo TESLA com taxa fixa de pacotes precisa assinar 1.036 pacotes de compromisso e gasta, em média, 0,373117 gigaciclos/hora. Enquanto a operação do *inf*-TESLA gasta, em média, 0,314087 gigaciclos/hora de operação, o que significa

Tabela 4.2: Cálculo do custo computacional.

	Emissor
TESLA (fixo)	$C * cSig + P * (cMac + cHash)$
inf-TESLA (ambos)	$cSig + 2 * P * (cMac + cHash)$
	Receptor
TESLA (fixo)	$C * cVer + P * (cMac + cHash)$
inf-TESLA (ambos)	$cVer + 2 * P * (cMac + cHash)$

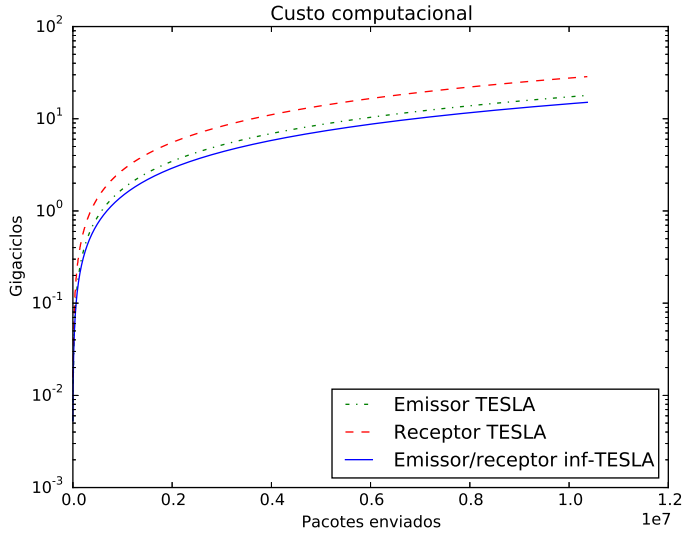

Figura 4.5: Custo computacional para TESLA e *inf*-TESLA durante 2 dias de comunicação.

uma redução de 15,82% no custo computacional para o emissor.

No outro lado da comunicação, um receptor operando o TESLA gasta, em média, 0,596289 gigaciclos/hora, enquanto operando o *inf*-TESLA seriam necessários 0,314303 gigaciclos/hora, o que significa uma redução de 47,29% no custo computacional para o receptor. Todos os valores de ciclos gastos por operação das primitivas de segurança são retirados dos testes realizados em Crypto++ Library 5.6.0 Benchmarks [24].

Custo computacional (variação de n). Outro resultado que queremos calcular é a comparação entre os dois protocolos quando é variado o número de chaves n que um emissor é capaz de armazenar. Na Figura 4.6, consideramos novamente uma simulação de operação por dois dias, incluindo TESLA e *inf*-TESLA, apresentando o custo computacional para o emissor e o receptor quando n é variado no intervalo $[1.000, 70.000]$.

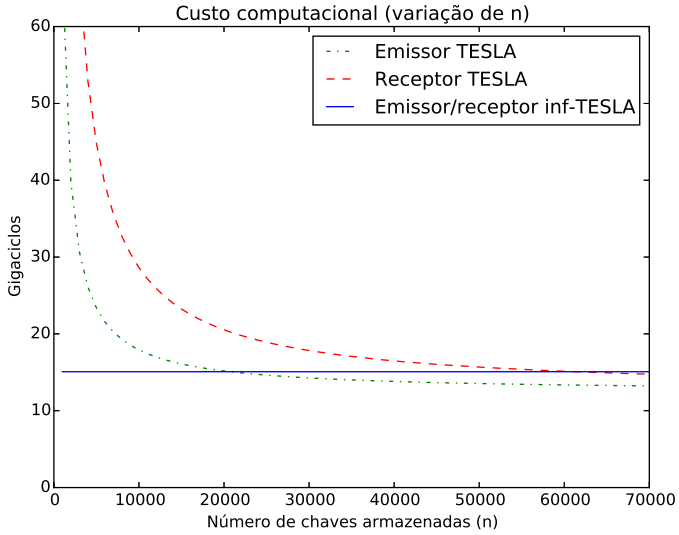

Figura 4.6: Custo computacional para TESLA e *inf*-TESLA, em dois dias de comunicação, ao variar o número de chaves, n, armazenadas pelo emissor.

Da simulação, constatamos que o emissor e o receptor usando o *inf*-TESLA mantêm constante o custo computacional quando n varia. Por outro lado, o emissor usando TESLA se torna mais eficiente quando ele é capaz de armazenar $n \approx 21.000$ chaves, e o receptor apenas quando $n \approx 62.000$. Considerando o custo de computação geral (emissor + receptor), TESLA atingiria, na média, uma melhor eficiência quando $n > 41.000$. Como uma desvantagem do TESLA, o tempo necessário para a geração e verificação de assinaturas digitais durante a substituição das cadeias de chaves pode afetar negativamente a continuidade do fluxo de dados em comunicações com alta frequência de envio de pacotes.

Atraso de autenticação. Embora o modo de divulgação *Alternado* apresente bons resultados nas avaliações anteriores, esse modo aumenta o atraso de autenticação em um pacote. Isso acontece porque a segunda chave necessária para autenticar um pacote P_i, ou seja, $K_{i_2}^{m+1}$, só será divulgado em P_{j+1}, onde $j > i + d$. Além disso, se P_{j+1} for perdido antes de alcançar o receptor, a autenticação de P_i somente será

validada quando o receptor receber a chave divulgada enviada em P_{j+3}. Em ambos os outros esquemas, a autenticação de um pacote P_i é validada, normalmente, após o recebimento de P_j, $j > i+d$. Caso P_j seja perdido, as chaves não recebidas podem ser calculadas a partir das chaves contidas em P_{j+1}.

Conclusão da avaliação. Observa-se, portanto, nesta avaliação comparativa em relação ao protocolo original, que o protocolo *inf*-TESLA no modo de divulgação de chaves *Alternado* pode proporcionar uma sobrecarga na comunicação ligeiramente menor. Em ambos os modos de divulgação, *2-chaves* e *Alternado*, o *inf*-TESLA oferece uma redução de 15,82% e 47,29% no custo computacional durante o período de operação, para o emissor e o receptor, respectivamente.

Embora o protocolo TESLA seja eficiente e apresente uma baixa sobrecarga de segurança, ele não foi originalmente concebido para funcionar de maneira eficiente em comunicações de longo prazo a altas taxas de dados e com o mínimo atraso de autenticação possível. O protocolo *inf*-TESLA se mostra eficiente dentro das condições citadas, além de se mostrar, de maneira geral, apropriado para dispositivos que possuem restrições quanto aos recursos computacionais e de armazenamento.

4.6 Conclusão

Neste capítulo, apresentamos a proposta de autenticação de evidências digitais em trânsito com foco nas aplicações de monitoramento existentes nas Redes Elétricas Inteligentes. Descrevemos o *inf*-TESLA, um protocolo de autenticação atrasada para comunicação *multicast*, apropriado para comunicações de longo prazo a altas taxas de envio de pacote. Para autenticar as mensagens do emissor, o esquema de autenticação do *inf*-TESLA utiliza duas chaves criptográficas para gerar o MAC da mensagem e divulga ambas as chaves após um intervalo de tempo d/r,

considerando uma comunicação com uma taxa fixa de pacotes, r, e o parâmetro de atraso, d.

Também desenvolvemos o mecanismo *Dual Offset Key Chain*, a fim de gerar as chaves de autenticação utilizadas no esquema de autenticação. Dessa forma, o esquema é capaz de autenticar dados em fluxo contínuo, sem a necessidade de periodicamente assinar digitalmente um pacote de sincronização, contendo os compromissos para as cadeias de chaves subsequentes. Demonstramos a segurança do nosso esquema durante a tentativa de um ataque homem no meio e descrevemos as condições de segurança que devem ser sempre observadas pelo receptor da comunicação.

São apresentados dois modos para divulgação de chaves, *2-chaves* (duas chaves divulgadas por pacote) e *Alternado* (uma chave divulgada por pacote, selecionada alternadamente entre as duas cadeias). Realizamos, também, uma avaliação comparativa entre esses dois modos e o TESLA, dentro de um cenário de uma aplicação WAMS. Embora o modo de divulgação de chaves *Alternado* aumente o atraso de autenticação em um pacote, os dois modos oferecem menos sobrecarga na comunicação e uma redução no custo computacional durante o período de operação. O *inf*-TESLA se mostra apropriado para ser utilizado por dispositivos que possuem restrições em relação aos recursos computacionais e de memória de armazenamento.

Como trabalhos futuros, consideramos a simulação do nosso protocolo em um simulador de eventos discretos. Assim como, também, a modelagem computacional do protocolo a fim de obter resultados em relação ao atraso de autenticação – mínimo, máximo, média, etc – em uma aplicação específica de monitoramento dos Sistemas Ciberfísicos.

5 AUTENTICAÇÃO BASEADA EM CARACTERÍSTICAS PARA DADOS DE MONITORAMENTO

Dados coletados podem gerar evidências digitais e indicar conclusões sobre algum fato. Como exemplos disso, podemos citar um retrato falado de um indivíduo concebido a partir de descrições de uma testemunha, ou a pegada de um calçado deixada em um cenário de crime, que pode revelar o modelo do sapato e, em última análise, associar ao sapato que a gerou [16]. Ambos os exemplos citados baseiam-se em características específicas de um conjunto de dados para determinar o todo.

Outro exemplo pode ser observado em um cenário de autenticação biométrica, onde as características físicas de uma pessoa são extraídas e interpretadas por um *software* de verificação, a fim de conceder acesso, ou não, a um determinado recurso. Dados sobre o glóbulo ocular ou impressão digital de um usuário são, portanto, transformadas digitalmente para servir como entrada de uma função computacional. Por sua vez, essa função classifica e correlaciona probabilisticamente os dados de entrada com as informações existentes em uma base de dados. Caso haja uma forte correlação com as informações de um usuário autorizado, o acesso é concedido. A partir desse exemplo, percebemos que os dados que garantiram a autenticação do usuário partiram de características físicas únicas do mesmo.

Nesta parte do trabalho, consideramos que, a partir das características únicas de um conjunto de dados, seja possível determinar com certa significância estatística o conjunto de dados como um todo. A autenticação baseada em características de um conjunto de dados é, aqui, considerada em um cenário de baixos recursos computacionais, onde é impraticável a obtenção e verificação de todos os dados necessários. Nossos estudos concentram-se em fluxos de dados de monitoramento

nos Sistemas Ciberfísicos, a fim de autenticar e oferecer não-repúdio, a um verificador externo, às evidências coletadas de dispositivos com restrições computacionais.

5.1 Definição do Problema

Os Sistemas Ciberfísicos são compostos por um diverso número de aplicações e tipos de dispositivos. Comumente, é esperado que alguns dispositivos da rede de comunicação tenham recursos computacionais, de memória e de rede limitados para executar tarefas específicas de monitoramento e controle dos processos físicos. Dentre as aplicações possíveis, as que demandam resposta em tempo real e, também, as que envolvem risco à vida tornam-se um desafio de implementação, uma vez que as restrições computacionais e de rede entram em conflito com os requisitos de segurança necessários e a qualidade do serviço desejada.

Neste parte do trabalho, consideramos um cenário onde as evidências digitais são retransmitidas e reautenticadas a cada salto da rede de comunicação. Isto é, ao longo da rede, as evidências são desempacotadas e asseguradas com a chave simétrica combinada entre os dois nós vizinhos. Para um verificador externo F, não há garantias de que esses dados não foram alterados no caminho desde o nó de origem (emissor S) até o nó destino (receptor R), uma vez considerado que os nós intermediários possam estar comprometidos. Por sua vez, o verificador F deseja validar os dados transmitidos e armazenados em R como evidências autênticas, a partir de provas suficientes da origem e integridade desses dados. Além disso, duas limitações do sistema precisam ser consideradas no problema aqui abordado: (i) a sobrecarga de segurança (computacional e na rede) não deve afetar o desempenho do sistema como um todo, e (ii) a não garantia da entrega de pacotes pela rede de comunicação.

A primeira limitação estabelece a preferência pelo uso de criptografia "leve", p.ex. a criptografia simétrica. Para não degradar o desempenho do sistema, torna-se impeditivo o uso intensivo de mecanismos de assinatura digital, pelo dispositivo emissor S, para autenticar todos os pacotes transmitidos pela rede ao receptor R. O uso da criptografia simétrica pressupõe que os nós vizinhos da rede compartilham uma mesma chave para autenticar e verificar, respectivamente, cada pacote. A assinatura digital pode ser usada, porém de forma moderada e dentro das possibilidades do sistema. A segunda limitação indica que nem todos os dados enviados por S serão armazenadas por R, pois alguns deles serão perdidos durante a transmissão. Dessa maneira, soluções que criam uma *tag* única de autenticação a partir de cadeias de *hash*, como visto na solução para log seguro apresentada no Capítulo 3, não funcionariam adequadamente pois o receptor R e o verificador F não seriam capazes de recriar essa *tag*, uma vez que o conjunto de dados encontra-se incompleto e a cadeia de *hash* não possa ser reconstruída.

De maneira geral, o problema de autenticação de dados, em uma transmissão suscetível a perdas, poderia ser estendido a qualquer sistema onde os dados são retransmitidos e reautenticados por nós intermediários apenas com o uso de chaves simétricas. Isso é verdade pois não há como comprovar que os dados foram originados no emissor, ou seja, não há garantia de não-repúdio dos dados, mesmo quando o receptor é considerado confiável. No entanto, para os sistemas que não possuem recursos limitados e detém características diferentes das abordadas neste trabalho, a solução desse problema é mais direta, como p.ex., adicionando uma assinatura para cada dado, ou bloco de dados, e retransmitindo pacotes quando houver perda.

No capítulo anterior, apresentamos uma solução para a autenticação fim-a-fim da transmissão, suscetível a perdas, de um fluxo de dados. No entanto, o esquema de autenticação apresentado é válido apenas para a verificação do receptor e não pode ser considerada para uma futura verificação desses dados por um verifi-

cador externo, pois o receptor tem conhecimento das mesmas chaves criptográficas simétricas usadas pelo emissor. A abordagem desta parte do trabalho é diferente da anterior. Aqui buscamos a "persistência" da autenticação desses dados através de mecanismos de não-repúdio, como a assinatura digital, porém respeitando as limitações do sistema.

Neste capítulo, propomos um esquema de uma *tag* de autenticação que oferece não-repúdio e garantia de integridade, a partir de um alto grau de correlação estatística, de um fluxo de dados transmitidos por um canal de comunicação suscetível a ataques e perdas de pacotes. O nosso esquema acrescenta uma camada a mais de segurança para a aplicação sendo executada, não interferindo nos pacotes da aplicação transmitidos. Um dos objetivos da solução é garantir, a um verificador externo, a detecção de alterações maliciosas no fluxo de dados após serem transmitidas por seu emissor.

A ideia básica sugere que um bloco de dados pode ser validado a partir da autenticação de características, ou um *fingerprint*, desse bloco. Uma *tag* contendo o *fingerprint* do bloco é assinada digitalmente e transmitida após os dados. Dessa forma, pode-se inferir se existe, ou não, uma diferença estatisticamente significante entre o *fingerprint* gerado no emissor e o *fingerprint* recriado no receptor. Caso a correlação entre os dois *fingerprints* for acima de um valor limite, o bloco de dados pode ser validado como autêntico. Mais uma vez, utilizaremos como caso de uso uma aplicação de monitoramento nas Redes Elétricas Inteligentes para a avaliação da nossa proposta.

O restante do capítulo está organizado da seguinte maneira. Na seção 5.2, descrevemos o modelo do sistema e seus participantes, o modelo de atacante, e as premissas consideradas. A seção 5.3 destaca os trabalhos que abordam soluções encontradas na literatura para o problema aqui definido. Na seção 5.4, apresentamos

a proposta de uma *tag* de autenticação baseada em características para dados de monitoramento, seus modos de operação, a geração e verificação do *fingerprint*, e uma discussão de segurança. Em seguida, realizamos uma avaliação experimental na seção 5.5 e, por fim, apresentamos nossas conclusões e trabalhos futuros na seção 5.6.

5.2 Modelos

Nesta seção, são apresentadas as nossas premissas, o Modelo do Sistema e o Modelo de Atacante utilizados nesta parte do trabalho.

5.2.1 Modelo do Sistema

O Modelo do Sistema utilizado neste capítulo é similar ao modelo descrito no capítulo anterior (seção 4.3.1), porém simplificado e acrescentado de um participante verificador (ver Figura 5.1). O sistema é composto basicamente por quatro partes, R (máquina receptora confiável), S (máquina emissora confiável), F (verificador externo) e I (máquina intermediária não confiável), que são detalhadas em seguida:

1. R, a máquina receptora confiável. R pode ser instanciada de diversas maneiras, p. ex., como um servidor *historian*[1] em um contexto SCADA. R possui armazenamento suficiente para guardar, de forma segura, todos os dados enviados por S;

2. S, a máquina emissora confiável. Nela são gerados os dados que serão enviados

[1]Um servidor operacional *historian* se refere a uma aplicação de base de dados que registra e armazena dados de processos baseados em tempo. Os *historians* são usados para registrar tendências e informações históricas sobre processos industriais para referência futura [97]

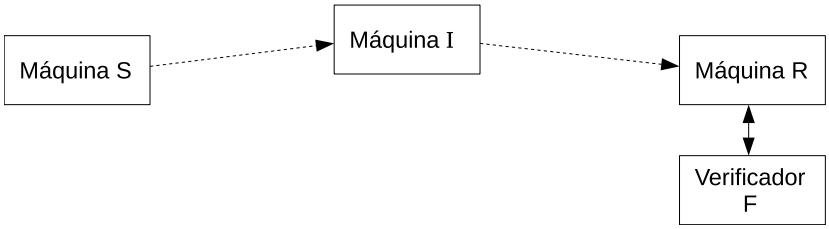

Figura 5.1: Modelo do Sistema.

à R. S possui recursos computacionais limitados, como memória, armazenamento e processamento, mesmo assim, S ainda é capaz de gerar uma assinatura digital. S não realiza *buffering* de pacotes, isto é, após o envio de um pacote, o mesmo é apagado da memória. Dessa forma, impossibilitando a retransmissão de qualquer pacote perdido. S não é capaz de responder requisições ou interagir com R, I ou F, sendo apenas possível a interação durante a troca de segredos ao início de uma transmissão, ou dada a implantação do sistema. A máquina S pode ser implementada, p.ex., como um sensor sem-fio ou como um *Phasor Measurement Unit* (PMU);

3. F, o verificador externo. O participante F pode ser instanciado, p.ex., como um analista, investigador ou perito. Em geral, F quer ter garantias e provas da origem de determinado conjunto de dados armazenados em R. F é capaz de se comunicar apenas com R e obter cópias de seus dados de forma segura e sem perdas;

4. I, a máquina intermediária não confiável. Não há garantias de que essa máquina está totalmente segura contra atacantes, erros de *software*, entre outras ameaças. Uma vez comprometida, I agirá da forma que o atacante estipular. I tem o papel de receber os pacotes de dados de S, reempacotar esses dados, reautenticá-los e repassá-los a R. Eventualmente, I poderá preencher algum dado que não tenha recebido, p. ex., através de técnicas de interpolação linear, e repassá-lo a R, indicando a sua intervenção em uma *flag* específica do pacote. Por fim, I atende aos requisitos de tempo de operação da camada de

aplicação.

Como parte importante do problema tratado, a rede de comunicação está sujeita a perda de pacotes. Com exceção da etapa de configuração inicial e da comunicação entre o par R e F, as máquinas não interagem entre si a fim de obter um dado/pacote perdido durante a transmissão.

Consideramos que a comunicação entre os participantes é protegida e garante, ponto a ponto, a confidencialidade e a integridade dos dados transmitidos com sucesso. A proteção dos dados é obtida através de mecanismos de criptografia simétrica, onde as máquinas S e I compartilham uma chave K_1, enquanto as máquinas I e R compartilham outra chave K_2. Por padrão, o j-ésimo pacote é formado pelas seguintes informações, $P_j = \{M_j, MAC(K_x, M_j)\}$, onde M_j representa a j-ésima mensagem (ou dado) a ser transmitida, acompanhada de seu respectivo *message authentication code* (MAC), criado em conjunto com a chave compartilhada K_x entre dois nós da rede.

5.2.2 Modelo de Atacante

Dentre as ameaças esperadas, assumimos que a máquina I pode ser comprometida, enquanto as outras máquinas permanecem confiáveis. A seguir, consideramos as possibilidades de um atacante durante o comprometimento de I:

- O atacante toma conhecimento das chaves de autenticação dos pacotes, armazenadas em I;
- O atacante é capaz de realizar manipulação/injeção de dados nos pacotes e reautenticá-los, sem necessariamente ativar a *flag* que indica a alteração do

dado no pacote;

- O atacante pode descartar qualquer pacote, impedindo-o de chegar ao seu destino;
- Ainda que atue maliciosamente, a máquina I comprometida é capaz de agir de acordo com a aplicação em execução, com a finalidade de não ser percebida por S ou R.

A fim de maximizar o dano causado, o objetivo do atacante é manipular a extensão dos dados o máximo possível sem que a detecção seja possível. A manipulação de dados de monitoramento pode causar a alteração do estado do sistema, conduzindo a ações de controle erradas e de maiores consequências, como p. ex., impacto ambiental e econômico. Desconsideramos qualquer outro ataque do tipo homem no meio nos *links* de comunicação, com exceção dos ataques realizados a partir da máquina I.

5.3 Trabalhos Relacionados

Um vasto número protocolos de autenticação de fluxo de dados existem na literatura. Seus diferentes propósitos incluem a autenticação por *link* ou fim-a-fim, e também a natureza do fluxo de dados: *offline*, ou seja, os dados são conhecidos pelo emissor de antemão, ou *online*, que é gerado em tempo real [4]. Na Seção 4.2, destacamos trabalhos para autenticação de um fluxo de dados em uma comunicação *multicast*. Nesta seção, destacamos os principais trabalhos relacionados a autenticação de dados para fluxos de dados que oferecem não-repúdio.

É importante ressaltar que as assinaturas digitais são computacionalmente onerosas para gerar e verificar. Logo, as abordagens que oferecem não-repúdio para

a autenticação de um fluxo de dados se classificam em duas: desenvolver esquemas de assinatura mais rápidos, ou amortizar cada operação de assinatura usando uma única assinatura digital para autenticar vários pacotes Aqui, observamos apenas os trabalhos da segunda abordagem.

O esquema proposto por Perrig et al. [77], denominado *Efficient Multi-chained Stream Signature* (EMSS), utiliza uma combinação de funções de *hash* e assinaturas digitais para autenticar pacotes. O *EMSS* é uma extensão da técnica de assinatura de Gennaro e Rohatgi [34]. A idéia básica é a seguinte: um *hash* do pacote P_1 é anexado ao pacote P_2, cujo *hash* é, por sua vez, anexado a P_3. Se um pacote de assinatura, contendo o *hash* do pacote de dados final (isto é, P_3) juntamente com uma assinatura do *hash*, é enviado após P_3, então o não-repúdio é obtido para todos os três pacotes. Em essência, os valores de *hash* atuam como se fossem cadeias entre os pacotes, de modo que eles formam uma única sequência que pode ser assinada por uma assinatura digital. Para reduzir o atraso na verificação, um fluxo de pacotes é dividido em blocos e o processo descrito é repetido para cada bloco. O *EMSS* alcança taxas de verificação relativamente altas, ao custo do aumento da sobrecarga de espaço de armazenamento e do atraso na verificação.

Golle e Modadugu [37] propõem um esquema semelhante ao *EMSS* chamado de técnica de cadeia aumentada (*augmented chain*). Eles propõem um método sistemático de inserir *hashes* em locais estratégicos a fim de que a cadeia de pacotes, formada pelos *hashes*, seja resistente a perdas de pacotes em rajada. A cadeia de pacotes construída usando este método é parametrizada pelas variáveis inteiras a e p. Uma cadeia de pacotes pode suportar uma perda em rajada de comprimento até $p(a-1)$ pacotes. Para reduzir o atraso de verificação, um fluxo é dividido em blocos de pacotes. Cada bloco é construído usando a técnica de cadeia aumentada, e apenas o último pacote do bloco é assinado.

Zhang et al. [111] propôs um esquema de autenticação de fluxos baseado em um grafo borboleta (*butterfly graph*) para redes com perdas, onde os pacotes são perdidos de forma aleatória ou em rajada. O esquema divide o fluxo de dados em blocos, onde apenas uma assinatura é gerada por cada bloco. Os pacotes de cada bloco, mais o pacote de assinatura, são conectados usando o grafo borboleta. Os resultados da análise teórica e da simulação mostram pouca sobrecarga e alta probabilidade de autenticação.

Park et al. [73] propuseram o *SAIDA* que utiliza códigos *erasure* para amortizar uma única operação de assinatura por vários pacotes. No *SAIDA*, um bloco – uma série de pacotes contíguos – de a pacotes carrega o *hash* codificado e a assinatura do bloco. A assinatura e os *hashes* são recuperáveis se o receptor receber qualquer $b \leq a$ pacotes. A codificação FEC (*Forward Error Correction*) usada para o *hash* é robusta contra perdas de pacotes em rajadas até um certo nível. Para reduzir a sobrecarga, o FEC é usado apenas para codificar os *hashes*, mas não para os dados. Esse esquema é estendido por Pannetrat e Molva [72], onde o número de pacotes de *hash* transmitidos é ainda mais reduzido, melhorando a eficiência sem impactar muito o desempenho. Isso é possível porque o próprio receptor gera determinados valores de *hash* de pacotes de dados recebidos com sucesso.

Song et al. [94] propuseram um esquema de autenticação baseado no grafo expansor (*expander graph*) e, também, demonstraram o limite inferior de sua probabilidade de autenticação. No entanto, a proposta adiciona uma sobrecarga de comunicação muito grande, o que torna-se impeditivo para uma aplicação real. O esquema de Miner e Staddon [63] é baseado no grafo aleatório (*random graph*). A assinatura é colocada no primeiro pacote, e cada pacote contém os *hashes* de todos os pacotes subsequentes com certa probabilidade. Dessa forma, essa solução também resulta em uma alta sobrecarga na comunicação.

Wong e Lam [105] utilizam as árvores de autenticação *Merkle* [62] para reduzir o tamanho das informações de autenticação e assinar pacotes em uma comunicação *multicast*. Eles propõem fazer *buffering* de pacotes em grupos. Uma única assinatura é calculada para cada grupo em função de todos os *hashes* dos pacotes. Além desta assinatura em comum, cada pacote contém informações auxiliares, das quais é possível que o valor pode ser recuperado. Isso permite que cada pacote possa ser verificado independentemente dos outros. Na configuração mais simples, os *hashes* de todos os pacotes do grupo são concatenados em uma sequência de caracteres. A informação auxiliar consiste dos *hashes* de todos os outros pacotes. Uma abordagem variante apresentada utiliza uma árvore binária para reduzir o tamanho dessa informação auxiliar. Os *hashes* dos pacotes são colocados nas folhas da árvore, enquanto os nós internos contêm um *hash* dos valores concatenados de seus filhos.

As árvores de *hash* também são usados por Habib et al. [39] para autenticar arquivos de mídia em redes *peer-to-peer offline*. Um arquivo é dividido em blocos de dados e uma árvore é construída sobre o todo o conjunto de dados. A raiz assinada é comunicada inicialmente ao cliente de *download*. O cliente, por sua vez, solicita aos *peers* as informações codificadas em FEC, para que possa reconstruir a árvore inteira e, dessa forma, permitindo que ele verifique os blocos de dados conforme forem chegando.

Ali et al. [4] desenvolveram um esquema de autenticação leve apropriado para monitoramento *online* de saúde e projetado para dispositivos de recursos limitados. O esquema utiliza uma árvore de *hash Merkle* para amortizar os custos da assinatura digital e uma codificação de rede para tornar o esquema tolerante à perda de pacotes. Essa codificação é implementada por pacotes de recuperação de dados. Eles propõe um arcabouço de otimização para que os pacotes de recuperação sejam configurados para maximizar a probabilidade da verificação dos dados transmitidos. Como resultados, eles obtém uma autenticação de mais de 99% dos dados com até

5% de sobrecarga na comunicação. O esquema proposto neste capítulo consegue obter mais de 99% de correlação nos dados apenas com o envio da *tag*, sem precisar de pacotes extras de recuperação de dados.

Os trabalhos relacionados apresentados anteriormente abordam o problema da autenticação de um fluxo de dados de forma criptográfica – por cadeias de *hash* e árvores de *hashes* – ou aumentando a chance de verificação do maior número de pacotes, seja posicionando informações de maneira específica ou criando redundância dessas informações. Diferentemente das propostas anteriores, a proposta deste capítulo tem como objetivo um esquema de autenticação baseado em características dos blocos de dados. As características oferecem provas estatisticamente significantes da integridade dos dados em questão. Dessa forma, a abordagem da nossa solução encontra-se diretamente associada ao conteúdo dos dados transmitidos. Aos nossos conhecimentos, este é o primeiro trabalho a considerar a autenticação de um fluxo de dados baseado em características do conjunto e que utiliza a similaridade entre as características como fator de decisão.

Por fim, o esquema proposto, apresentado a seguir, oferece uma proteção complementar àquela vista no capitulo 4, o protocolo *inf*-TESLA, que realiza a autenticação durante a transmissão de dados, porém não garante autenticidade e não-repúdio dos dados após a revelação das chaves de autenticação.

5.4 *Tag* de Autenticação baseada em Características

Nesta seção, descrevemos a nossa proposta para um esquema de uma *tag* de Autenticação baseada em Características para Dados de Monitoramento. O objetivo da proposta é oferecer não-repúdio e garantia probabilística de integridade de um fluxo de dados transmitidos por um canal de comunicação suscetível a ataques e

perdas de pacotes. Essa garantia é obtida a partir de um alto grau de correlação estatística entre as características dos dados originais e dos dados recebidos pelo receptor.

A *tag* serve como uma prova persistente e autenticada dos dados gerados pelo emissor. Ela carrega informações sobre determinado bloco de dados, incluindo seu *fingerprint*, descrito na seção 5.4.3. Em qualquer momento no futuro, ao ser verificada, a *tag* é capaz de prover detecção de possíveis ataques de injeção e manipulação maliciosa ocorridos durante a transmissão de dados. O receptor que armazenar de forma segura os dados de monitoramento, em conjunto com suas respectivas *tags*, poderá comprovar estatisticamente, a um verificador externo, a legitimidade dos dados que ele armazena.

Abaixo, destacamos as propriedades do esquema de autenticação proposto:

- Tolerante a perda de dados. A verificação da *tag* não depende que todos os dados que geraram o *fingerprint* estejam presentes.
- Publicamente verificável. Através da assinatura digital e de uma Infraestrutura de Chaves Públicas, a *tag* pode ser verificável por todos os interessados. Ao contrário de dados autenticados através de uma chave simétrica, a *tag* oferece integridade, não-repúdio e garantia de origem das informações nela contida.
- Baixa demanda computacional. O esquema prevê o uso de assinatura digital de forma reduzida, somente para cada *tag*, e não para todo dado enviado. A função geradora do *fingerprint* dos dados não depende de cálculos matemáticos complexos.
- Ajustável à aplicação. Pode ser configurado para exigir menos/mais esforço computacional (ver seção 5.4.1), de acordo com as possibilidades dos dispositivos disponíveis no sistema.

Como esse esquema de autenticação adiciona uma camada de segurança a mais na aplicação de monitoramento em execução, ele pode operar de diferentes formas de acordo com o nível de detecção e qualidade de serviço que se deseja obter. Na seção seguinte, introduzimos os três modos de operação possíveis do esquema proposto. Nas seções subsequentes, apresentamos os campos de dados que compõe a *tag* de autenticação, como é feita a geração do *fingerprint* de dados, a verificação da *tag*, e, por fim, apresentamos uma discussão de segurança.

5.4.1 Modos de operação

É possível que uma única transmissão de fluxo de dados possua um número maior de informação que uma única *tag* pode autenticar. Por isso, o emissor deve gerar uma sequência de *tags* $S = \{T_1, T_2, ..., T_i, ..., T_n\}$, independentes entre si, a fim de prover autenticação para todo o conjunto de dados transmitidos. Logo, a sequência S estende-se de acordo com o total de dados transmitidos.

O modo de operação determina a cobertura das *tags* de autenticação em S sobre o fluxo de dados transmitidos. As *tags* podem ser geradas de acordo com os três modos de operação disponíveis: (i) *Sobreposto*, (ii) *Continuado*, e (iii) *Intervalado*.

O modo de operação *Sobreposto* (Figura 5.2.a) especifica que as *tags* T_i e T_{i+1} autenticam blocos de dados, B_i e B_{i+1} respectivamente, que possuem intersecção entre si. T_i e T_{i+1} são geradas concorrentemente e, para cada uma, o emissor deve possuir armazenamento suficiente para guardar os dados referentes a cada *fingerprint*. Esse modo de operação exige maior poder computacional do dispositivo emissor pois demanda que uma frequência maior de assinaturas digitais sejam computadas ao longo da transmissão de dados. No entanto, ele estabelece uma maior segurança dos dados e melhor detecção de alterações maliciosas, em relação aos

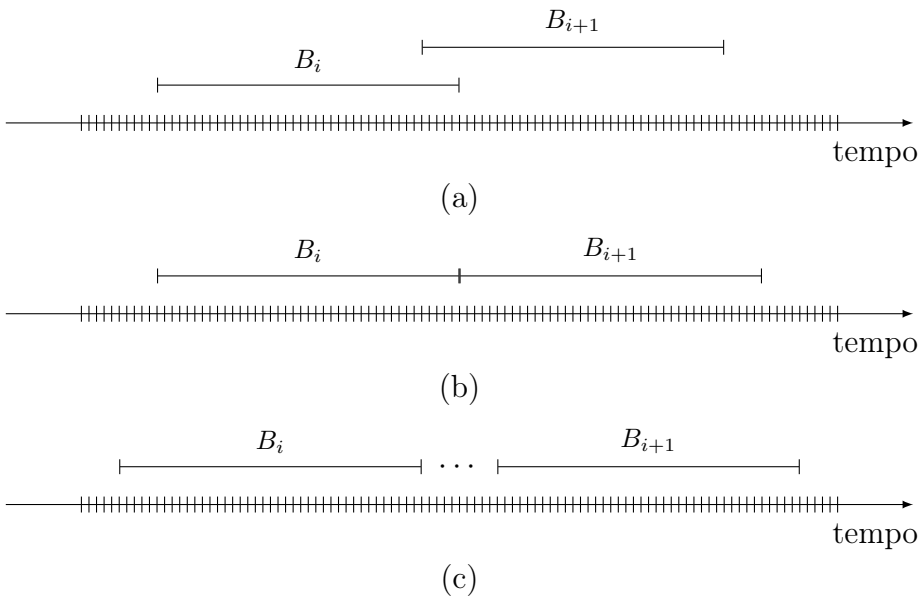

Figura 5.2: Modos de Operação da *Tag* de Autenticação: (a) *Sobreposto*, (b) *Continuado*, e (c) *Intervalado*.

outros modos de operação, pois oferece uma redundância na autenticação.

O modo de operação *Continuado* (Figura 5.2.b) especifica que as *tags* T_i e T_{i+1} autenticam blocos de dados, B_i e B_{i+1} respectivamente, sem interseção. As *tags* são geradas uma após a outra, proporcionando autenticação para todos os dados do fluxo. Esse modo apresenta um equilíbrio entre a segurança de dados e a exigência computacional do emissor, e é considerado o modo de operação padrão do esquema proposto.

O modo de operação *Intervalado* (Figura 5.2.c) especifica que as *tags* T_i e T_{i+1} autenticam blocos de dados, B_i e B_{i+1} respectivamente, sem interseção porém sem abranger todos os dados do fluxo. O intervalo de pontos não protegidos, localizados entre B_i e B_{i+1}, pode ser pré-definido e redefinido durante a transmissão, dado o nível de segurança desejado. Dessa forma, esse modo pretende desonerar computacionalmente o emissor em troca de menos *tags* geradas e menos dados protegidos. Esse

modo é aconselhável para dispositivos com grandes restrições de memória e poder computacional, ou quando a proteção parcial do fluxo de dados seja suficiente.

Para todos os modos de operação, cada *tag* T_i é sempre transmitida pelo emissor posteriormente ao envio do bloco de dados B_i.

5.4.2 Campos de dados da *Tag* de Autenticação

A *tag* de autenticação foi projetada para se ajustar inteiramente dentro de um único pacote de dados, a fim de evitar sobrecargas excessivas na rede de comunicação. O tamanho total de uma *tag* é esperado que não ultrapasse o MTU[2] da rede, considerando ainda os cabeçalhos das camadas 2 e 3 da rede. Cada *tag* T_i possui os seguintes campos de dados (Figura 5.3):

- *Timestamp* início, TS_{i-1}^{ini}. Referente ao primeiro elemento do bloco B_{i-1}. Tamanho do campo: 8 bytes.
- *Timestamp* fim, TS_{i-1}^{fim}. Referente ao último elemento do bloco B_{i-1}. Tamanho do campo: 8 bytes.
- *Timestamp* início, TS_{i}^{ini}. Referente ao primeiro elemento do bloco B_i. Tamanho do campo: 8 bytes.
- *Timestamp* fim, TS_{i}^{fim}. Referente ao último elemento do bloco B_i. Tamanho do campo: 8 bytes.
- ID de origem, ID_{code}. Código identificador do dispositivo emissor. Tamanho do campo: 2 bytes.

[2] *Maximum Transmission Unit*, o máximo tamanho de um datagrama que pode ser transmitido pela rede [83].

<table>
<tr><th colspan="3">0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</th></tr>
<tr><td colspan="3">$TimeStamp$ ínicio última tag (TS_{i-1}^{ini})</td></tr>
<tr><td colspan="3">$TimeStamp$ fim última tag (TS_{i-1}^{fim})</td></tr>
<tr><td colspan="3">$TimeStamp$ ínicio atual (TS^{ini})</td></tr>
<tr><td colspan="3">$TimeStamp$ fim atual (TS^{fim})</td></tr>
<tr><td colspan="2">Identificador (ID_{code})</td><td>Número de pontos (Num^{pt})</td></tr>
<tr><td>$Filter_{id}$</td><td>Num_{pa}</td><td>$[Params]$</td></tr>
<tr><td colspan="3">$Fingerprint$ (fp)
...</td></tr>
<tr><td colspan="3">Valor mínimo (val^{min})</td></tr>
<tr><td colspan="3">Valor máximo (val^{max})</td></tr>
<tr><td colspan="3">Assinatura digital (Sig^{Priv})
...</td></tr>
</table>

Figura 5.3: Campos de dados da *Tag* de Autenticação.

- Número de pontos, Num_i^{pt}. Indica o número de dados total do bloco B_i. Tamanho do campo: 2 bytes.

- Identificador do Filtro, $Filter_{id}$. Indica qual o filtro usado na geração do *fingerprint*. Tamanho do campo: 1 byte.

- Número de parâmetros, Num_{pa}. Número de parâmetros que a função de filtro

possui. Tamanho do campo: 1 byte.

- Vetor de parâmetros do filtro, $[Params_i]$. Parâmetros de entrada para a função de filtro. Tamanho do campo: $(2 * Num_{pa})$ bytes.

- *Fingerprint*, fp_i. Representação das n características do bloco B_i. Tamanho do campo: ∼1000 bytes.

- Valor mínimo, val_i^{min}. Menor valor da série de dados em B_i. Tamanho do campo: 4 bytes.

- Valor máximo, val_i^{max}. Maior valor da série de dados em B_i. Tamanho do campo: 4 bytes.

- Assinatura digital, Sig_i^{Priv}. Assinatura digital de todos os campos anteriores, a partir da chave privada única do dispositivo emissor, e codificada no formato DER[3]. Tamanho do campo: ≤ 72 bytes.

Os algoritmos de *hash* criptográfico e de assinatura digital, além de parâmetros específicos e certificados digitais, são conhecidos previamente pelas partes do sistema a partir de uma mensagem inicial de configuração. Assumimos que essa mensagem é transmitida de forma segura entre o emissor e o receptor no início da comunicação. Assumimos também que, após recebida pelo receptor, a *tag* pode ser armazenada de forma legível, ou mesmo codificada em um formato específico. A formatação das informações para fins de armazenamento encontra-se fora do escopo deste trabalho.

[3] *Distinguished Encoding Rules* [48]

5.4.3 Geração do *fingerprint*

O campo de *fingerprint* contém a informação mais significativa do nosso esquema de autenticação. Nesse campo, são armazenadas as características relevantes de um bloco do conjunto total do fluxo de dados transmitidos. As características, c's, são geradas a partir de um filtro de dados específico, com saída determinística e de funcionamento publicamente conhecido.

O emissor de dados gera o i-ésimo *fingerprint* fp_i seguindo a seguinte equação:

$$fp_i = Filter([Params_i], B_i) \tag{5.1}$$

onde a função $Filter()$ é determinada pelo identificador $Filter_{id}$. O resultado da Equação 5.1 é uma sequência de características $fp = \{c_1, c_2, ..., c_n\}$, onde n é o número total de características.

Para este trabalho, o Filtro de Médias Móveis (*Moving Average Filter*, ou MA *filter*) (Seção 2.3.1) foi escolhido para instanciar a função $Filter()$. O filtro MA atende aos requisitos abordados neste trabalho por exigir pouca memória e baixo esforço computacional, e possuir simplicidade algorítmica e fácil implementação. Os dois parâmetros que o filtro MA recebe como entrada são: (i) a *janela*, o tamanho das amostras, e (ii) o *passo*, o fator de compressão. Esses parâmetros podem variar de acordo com a aplicação que está sendo executada.

Utilizando o filtro MA, as características de cada bloco B_i são, portanto, a média aritmética dos n subconjuntos de dados de tamanho igual à $janela$. Ao longo da geração e transmissão dos dados do bloco B_i, o emissor calcula as n médias e as armazena em uma memória *buffer*. Ao fim do cálculo, o *fingerprint* pode ser

adicionado ao pacote da *tag* de autenticação correspondente e, por fim, transmitido ao receptor.

5.4.4 Verificação da *Tag*

Ao ser recebida pelo receptor, a *tag* deverá ser armazenada de forma segura. Em um dado momento após o recebimento dos dados e sua *tag* correspondente, um verificador externo F realizará a verificação das informações contidas na *tag* a fim de validar corretamente as evidências armazenadas. F, portanto, deve obter de R uma cópia da *tag* T_i e de todos os respectivos dados no intervalo $[TS_i^{ini}, TS_i^{fim}]$. Esse bloco de dados, $B_i^{'}$, pode ser ligeiramente diferente de B_i e conter dados interpolados, em consequência da perda de pacotes durante a transmissão desde S. Conforme dito anteriormente, apenas a comunicação entre R e F é feita de forma segura e sem perdas.

A partir da Equação 5.1, F aplica o filtro indicado por $Filter_{id}$ em $B_i^{'}$ para criar o *fingerprint* $fp_i^{'}$. Considerando que o *fingerprint* original e o recriado por F, fp_i e $fp_i^{'}$ respectivamente, sejam duas variáveis aleatórias e possuam uma correlação entre si, a validação do *fingerprint* é dada pela seguinte equação:

$$corr(fp_i, fp_i^{'}) \geq corr_{lim} \tag{5.2}$$

onde a função $corr()$ é representada pela correlação de *Pearson* (Seção 2.3.3). Se a correlação for maior ou igual a $corr_{lim}$, assumimos que não existe diferença significativa entre as representações e, portanto, o *fingerprint* $fp_i^{'}$ é validado, caso contrário, não. A correlação limite $corr_{lim}$, em geral, assume valores iguais a 0,98, 0,99, 0,999, etc.

Além da verificação do *fingerprint*, F deve testar se todo o bloco de dados está de acordo com a equação abaixo:

$$B_i^{'} \subseteq [val_i^{max}, val_i^{min}] \tag{5.3}$$

a qual mostrando-se falsa, a *tag* T_i não pode validar o bloco $B_i^{'}$.

Por fim, caso as Equações 5.2 e 5.3 sejam verdadeiras, e a assinatura digital Sig_i^{Priv} for válida, o bloco de dados guardado em R e verificado por F, $B_i^{'}$, é validado pela *tag* T_i.

5.4.5 Discussão de segurança

Nesta subseção, discutimos alguns aspectos de segurança do esquema proposto, apontando fraquezas e possíveis mitigações.

Importância do t_{min}. Um ponto que deve ser destacado é em relação ao momento no qual a *tag* T_i deve ser transmitida pelo emissor. T_i deve ser transmitida após um período mínimo de tempo, t_{min}, a partir do último elemento do bloco de dados B_i. O período t_{min} está fortemente vinculado à aplicação que está sendo executada. Isso se faz importante do ponto de vista da segurança, pois nenhum atacante no meio da comunicação terá acesso à T_i durante a transmissão dos dados de B_i e, consequentemente, às suas informações, incluindo como o *fingerprint* fp_i foi gerado.

Tendo em vista que o fluxo de dados é constante de S para R, a máquina intermediária I deve estar sempre retransmitindo os dados. Logo, I não pode reter os pacotes e modificá-los após a leitura de T_i, pois isso levantaria suspeita por parte

do receptor. Dessa forma, o requisito de que haja um período t_{min} para o envio da *tag*, colabora para que o atacante tenha menos probabilidade de alterar, de forma indetectável, esses dados.

Ataque de Negação de Serviço e Descarte Seletivo. No entanto, a máquina I pode realizar um ataque de negação de serviço nos pacotes de dados da aplicação e, inclusive, nos pacotes das *tags* de autenticação. Para os pacotes da aplicação, esse ataque é difícil de ser evitado porém, um limite superior de perda de pacotes na rede pode ser estipulado a fim de que esse tipo de ataque seja mitigado. No caso de um destarte seletivo de pacotes realizado por I, esse ataque pode tornar-se indetectável. Caso I seja capaz de descartar pacotes e invalidar a Equação 5.2, ele conseguirá prejudicar o esquema de autenticação proposto. No caso de um descarte seletivo de pacotes com as *tags* de autenticação, cada *tag* contém uma referência para outra exatamente anterior a ela. Os campos TS_{i-1}^{ini} e TS_{i-1}^{fim} informam ao receptor R a referência para a *tag* T_{i-1}. Esse mecanismo não evita o ataque, mas oferece uma informação extra para controle de perdas e possível descarte de *tags*.

Ataques específicos. Outra estratégia que o atacante pode adotar está em somar e subtrair, alternadamente, um determinado valor x dos dados transmitidos. A isso, iremos chamar de ataque *Alternado*. O ataque *Alternado* é definido da seguinte forma: para um janela de dados $j = \{d_1, d_2, d_3, d_4, ...\}$, o atacante o modifica em tempo real para $j' = \{d_1 + x, d_2 - x, d_3 + x, d_4 - x, ...\}$. Isso garantiria que a média aritmética dos dados da janela j' seja igual à original j. Além disso, somas e subtrações pontuais de um mesmo valor x no fluxo de dados também podem esconder a modificação no *fingerprint*, p.ex. $j' = \{d_1 + x, d_2, d_3 - x, d_4, ...\}$.

Para mitigar o sucesso dessas estratégias de ataque, o filtro de média móveis poderia usar média ponderada para o cálculo de cada janela de dados. Isso torna-se mais eficaz porque o atacante não sabe, de antemão, quando ou a partir de qual

ponto uma janela de dados foi considerada, tornando-se mais difícil falsificar os dados do *fingerprint*. Além disso, a perda de dados, esperada durante a transmissão, e a interpolação, consequente dos dados perdidos, podem deixar mais evidente esse tipo de modificação maliciosa durante a verificação do *fingerprint*.

Por fim, os valores dos parâmetros *janela* e *passo* devem ser variáveis. O atacante em I pode aprender os valores dos parâmetros, caso eles não sejam modificados com o passar do tempo. A variação dos parâmetros aumenta a dificuldade de realizar modificações intencionais no *fingerprint* sem que sejam detectadas, uma vez que não se sabe qual o tamanho da *janela* está sendo usado.

5.5 Avaliação

Nesta seção, avaliaremos nossa proposta para uma aplicação de monitoramento nas Redes Elétricas Inteligentes, também conhecida como uma aplicação WAMS (*Wide Area Monitoring System*). Realizamos a simulação dos três seguintes cenários para a aplicação em questão: (i) considerando apenas a perda de pacotes na rede de comunicação, (ii) máquina intermediária comprometida e ataque de injeção de dados do tipo *Degrau*, e (iii) máquina intermediária comprometida e ataque de injeção de dados do tipo *Alternado*.

A ideia principal da avaliação é saber se o receptor R tem condições de validar com sucesso a *tag* de autenticação, ou de invalidá-la sob um cenário de ataque. Dessa forma, para cada cenário avaliamos quantos *fingerprints* recriados em R, $fp^{'}$, possuem alta correlação com os *fingerprints* originais, fp, gerados em S. Para cada cenário, discutimos os resultados e apontamos a necessidade de utilizar as informações val^{min} e val^{max}.

Configuração da simulação. Para compor a base de dados de testes, utilizamos o *dataset* disponível livremente em [64], representando as medições realizadas no campus da *École Polytechnique Fédérale de Lausanne*, na Suíça [82]. Especificamente, utilizamos os dados de magnitude (em *volts*) da fase A, referentes aos dias 2, 3 e 4 de Maio/2016, do PMU $ID = 2$. O restante dos dados não são considerados nessa simulação.

As demais características dos cenários simulados estão descritas a seguir:

- As máquinas emissora S, intermediária I e receptora R são, respectivamente, os dispositivos *Phasor Measuremente Unit* (PMU), *Phasor Data Concentrator* (PDC), e um Super-PDC em conjunto com um servidor *historian*, de uma aplicação WAMS.
- A frequência de envio/geração de pacotes pelo dispositivo PMU é igual a 50 pacotes/segundo.
- O padrão da perda de pacotes durante a transmissão é descrito de acordo com o modelo *Gilbert-Elliot* (Seção 2.3.2), capaz de simular perdas em rajadas. O modelo *Gilbert-Elliot* recebe como entrada dois parâmetros, p e r, que sugerem as probabilidades de transição do estado *bom* para *ruim*, e *ruim* para *bom*, respectivamente.
- O MTU utilizado é de 1.500 bytes, como o padrão do protocolo Ethernet [50].

A *tag* de autenticação deve caber inteiramente dentro de um pacote de dados para evitar a sua fragmentação na rede. Dos 1.500 bytes do MTU, 20 bytes são reservados para o cabeçalho IP e 8 bytes são reservados para o cabeçalho UDP. Portanto, os 1.472 bytes restantes são reservados para o *payload* de dados, ou seja, a

tag de autenticação. Abaixo, seguem as características da *tag* de autenticação para esta seção de avaliação:

- O *fingerprint* é calculado a partir do Filtro de Média Móveis, com $janela = 60$ e $passo = 30$. Cada *fingerprint* possui 250 médias calculadas.
- O tamanho de cada valor contido no *fingerprint* é de 4 bytes.
- O algoritmo de assinatura digital utilizado é o ECDSA-256 [49], o qual gera uma assinatura digital com tamanho final de 72 bytes (codificado em DER).

A partir das características do primeiro item, o número total de pontos de dados em um bloco, suficientes para formar um *fingerprint*, é de 7.530 pontos. Consequentemente, cada nova *tag* necessita de 150,6 segundos (ou 2,51 minutos) para reunir todos os valores usados na geração do *fingerprint* – relacionado diretamente com a frequência de geração de dados, 50 pacotes/segundo, no modo de operação *Continuado*. A *tag* de autenticação possui um tamanho total de: 1.000 bytes (*fingerprint*) + 72 bytes (assinatura) + 50 bytes (campos restantes) = 1.122 bytes.

Para cada um dos três cenários, simulamos a geração de 1.000 *tags* de autenticação distintas, cada *tag* referente à um bloco de dados diferente. Toda a simulação foi implementada utilizando a linguagem Python, versão 2.7.6, e suas bibliotecas disponíveis.

Cenário de apenas perdas na rede. Para o primeiro cenário de simulação, consideramos a perda de pacote na rede de comunicação como o único agente causador da diferença entre o bloco de dados emitido pelo PMU, B_i, e o bloco de dados armazenados no Super-PDC, $B_i^{'}$. Os valores perdidos durante a comunicação são substituídos através de uma função de interpolação linear.

Os sistemas que utilizam dados de PMU devem operar com taxas de perdas de pacote (PLR, *packet loss rate*) menores que 0.1% [61]. Outros trabalhos [9] indicam que, de fato, o PLR nos sistemas WAMS pode variar entre 10^{-5}% até 0.03%. A fim de avaliar nossa proposta com diferentes PLRs, iremos utilizar os resultados de [65] para parametrizar o modelo *Gilbert-Elliot* com diferentes valores de p e r, como é mostrado na primeira coluna da Tabela 5.1.

As demais colunas da Tabela 5.1 demonstram os resultados obtidos após a geração de 1.000 *fingerprints*. Para cada *fingerprint* gerado, obtivemos a média de pacotes perdidos e o número de correlações acima de cada limite (0,99, 0,999, 0,9999) observado. Nota-se diretamente que, quanto maior a taxa média de perda de pacotes (PLR médio), menor será o número de validações que respeitam a correlação limite, $corr_{lim}$, entre fp e fp'. Os resultados mostram-se favoráveis quando a correlação limite $corr_{lim} = 0,99$, para a qual é obtida uma alta percentagem de validação, 99,9% e 99,7%, considerando os maiores PLRs médios testados, 0,1627 e 0,2213 respectivamente. Para os demais PLRs, a percentagem de validação chega a 100% dos *fingerprints* testados.

Para os dois cenários restantes nesta simulação (ataques de injeção de dados), adotamos um PLR médio de 0,0247.

Tabela 5.1: Simulação de perdas na rede (1000 rodadas/*fingerprints*).

PLR médio (p, r)	Média de pacotes perdidos $\pm\sigma$	Número de correlações > 0.99	> 0.999	> 0.9999
0,0247 (0,0082, 0,323)	186,03 ± 30,79	1.000	1.000	946
0,0303 (0,0097, 0,310)	228,39 ± 34,65	1.000	1.000	920
0,0436 (0,0109, 0,238)	328,354 ± 46,43	1.000	996	735
0,0835 (0,0144, 0,158)	629,04 ± 77,09	1.000	943	270
0,1627 (0,0238, 0,122)	1225,733 ± 116,32	999	705	90
0,2213 (0,0317, 0,111)	1666,919 ± 127,32	997	487	50

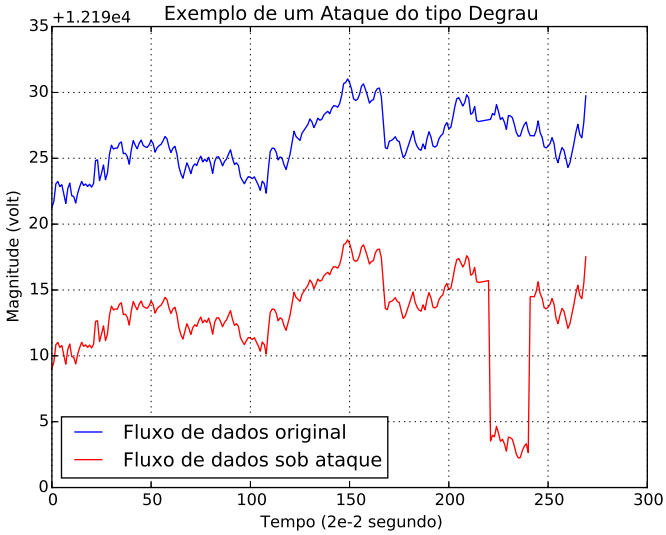

Figura 5.4: Exemplo de um ataque do tipo *Degrau*.

Cenário sob ataque do tipo Degrau. A Tabela 5.2 mostra os resultados após a geração de 1.000 *fingerprints* submetidos a um ataque do tipo *Degrau* (ver Figura 5.4). Para cada bloco de dados, foi aplicada uma modificação do tipo *Degrau* com diferentes percentuais de alteração em relação aos valores originais, são eles: 1%, 2%, 5%, 10%. A modificação afetou apenas 60 pontos – igual ao tamanho da *janela* – dos 7.530 pontos totais do bloco.

Dos resultados, podemos notar que a correlação média entre fp e fp' perma-

Tabela 5.2: Simulação de ataque de injeção de dados do tipo *Degrau* (1.000 rodadas/*fingerprints*).

Percentual de alteração	Correlação média $\pm\sigma$	Número de correlações > 0.99	Dentro do intervalo min/max
1%	0,6359 ± 0,167	16	53
2%	0,4153 ± 0,205	0	14
5%	0,2033 ± 0,178	0	0
10%	0,1115 ± 0,141	0	0

nece significativamente abaixo da correlação limite, $corr_{lim} = 0,99$, para validação do *fingerprint*. Podemos observar, também, a ocorrência de 16 Falsos Negativos, uma vez que esses 16 casos sofreram uma modificação maliciosa que não desrespeita o intervalo $[val^{min}, val^{max}]$ e indica que $corr(fp, fp^{'}) \geq 0,99$. De toda forma, os Falsos Negativos representam apenas 1,6% dos casos testados no cenário sob ataque do tipo *Degrau*, indicando uma percentual pequeno de não detecção. Foi possível invalidar, com sucesso, 98,4% dos *fingerprints* que sofreram o ataque.

Cenário sob ataque do tipo Alternado. A Tabela 5.3 mostra os resultados após a geração de 1.000 *fingerprints* submetidos a um ataque do tipo *Alternado* (ver Figura 5.5). O ataque do tipo *Alternado*, como explicado na Seção 5.4.5, propõe um ataque mais sofisticado, que tem como objetivo alterar maliciosamente os valores, porém tentando manter as médias das janelas bem perto das médias originais. Aos nossos conhecimentos, não se trata de um ataque específico aplicável a um cenário de uma aplicação WAMS. No entanto, é importante avaliar a nossa proposta contra esse tipo de ataque por causa do filtro utilizado na geração do *fingerprint* – o Filtro de Médias Móveis.

A partir dos resultados, podemos observar que as correlações médias entre fp e $fp^{'}$ mostram-se altas e relativamente perto da correlação limite para validação, $corr_{lim} = 0,99$. Para alterações de 1% e 2% nos dados, obtivemos respectivamente

Tabela 5.3: Simulação de ataque de injeção de dados do tipo *Alternado* (1.000 rodadas/*fingerprints*).

Percentual de alteração	Correlação média $\pm\sigma$	Número de correlações > 0.99	> 0.999	> 0.9999	Contido em $[val^{min}, val^{max}]$
1%	0,9993 $\pm 6 \times 10^{-4}$	1.000	783	156	7
2%	0,9975 $\pm 2,6 \times 10^{-3}$	982	379	64	0
5%	0,9861 ± 0,014	550	95	16	0
10%	0,9494 ± 0,048	183	33	0	0

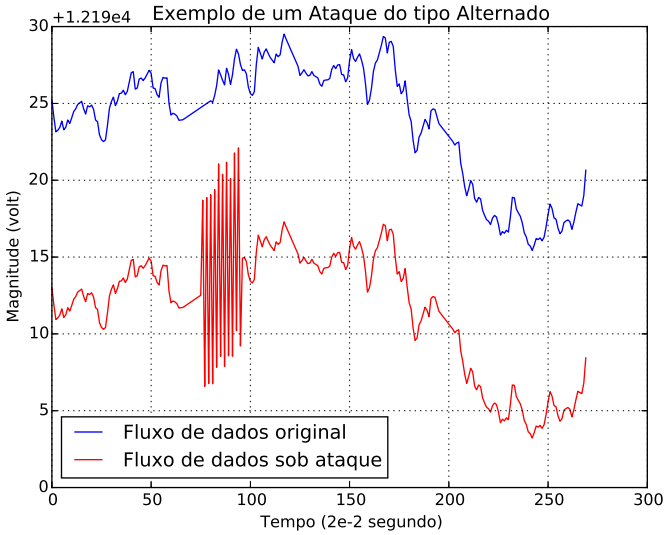

Figura 5.5: Exemplo de um ataque do tipo *Alternado*.

100% e 98,2% dos casos testados acima da $corr_{lim}$. No entanto, para esses mesmos percentuais de alteração, obtivemos apenas 7 e nenhum caso, respectivamente, de alterações que respeitaram o intervalo $[val^{min}, val^{max}]$. Dessa forma, podemos concluir que tivemos apenas 7 casos de Falsos Negativos no cenário sob ataque do tipo *Alternado*, ou seja, 0,7% do total. Foi possível invalidar, com sucesso, 99,3% dos *fingerprints* que sofreram o ataque.

Nas aplicações WAMS, a acurácia nas medições de um típico PMU podem variar em 0,5% [61]. O padrão IEEE Standard C37.118.1-2011 [43] define requisitos para a acurácia das medições dos PMUs na faixa de 1%. Dessa forma, permite-se um erro de até 1% nas medições da magnitude da fase. Em relação aos nossos resultados, a informação sobre o erro permitido nas medições dos PMUs é bastante conveniente, uma vez que todos os casos de Falsos Negativos foram observados quando o percentual de alteração é igual a 1%. Isso indica que uma alteração dessa proporção não afeta significativamente uma aplicação WAMS, sendo inclusive permitida pelo padrão.

Conclusão da avaliação. Podemos concluir nossa avaliação experimental ressaltando os resultados satisfatórios obtidos na validação e na detecção de ataques de injeção de dados durante a transmissão de dados nos três cenários cogitados. Para o primeiro cenário, considerando apenas as perdas da rede, obtivemos entre 99,7% e 100% de validação dos *fingerprints*. No segundo cenário, considerando um ataque do tipo *Degrau*, obtivemos 98,4% de invalidação com sucesso dos *fingerprints*, e 16 casos de Falsos Negativos. No terceiro cenário, considerando um ataque do tipo *Alternado*, foi obtido 99,3% de invalidação com sucesso dos *fingerprints*, e 7 casos de Falsos Negativos. No entanto, para uma aplicação WAMS, os casos de Falsos Negativos observados poderiam ser negligenciados por estarem dentro da faixa do percentual permitido de erro de medição do PMU.

Por fim, destacamos que o esquema de autenticação aqui avaliado gera uma assinatura digital a cada 2,5 minutos, considerando a frequência de 50 pacotes/segundo e os demais parâmetros da simulação, o que equivale a 24 assinaturas digitais geradas por hora. Considerando os sistemas de recursos limitados, a utilização de forma não intensiva da assinatura digital ao longo da transmissão é apropriada, pois reduz o esforço computacional dos dispositivos e não sobrecarrega a rede de comunicação.

5.6 Conclusão

Neste capítulo, apresentamos a proposta de uma esquema de autenticação baseada em características para dados de monitoramento, permitindo a validação externa (ou pública) de evidências digitais. Esse esquema considera as possíveis limitações de recursos computacionais dos dispositivos, utilizando uma única assinatura digital para vários pontos de dados, e também considera um canal de comunicação suscetível a ataques e perda de pacotes.

Nossa abordagem descreve uma *tag* de autenticação que é gerada para cada bloco de dados que se queira autenticar. A *tag* carrega informações sobre o bloco de dados, incluindo um *fingerprint*, juntamente com a sua assinatura digital. Descrevemos os três modos de operação do esquema de autenticação, assim como os campos de dados da *tag*, e a geração e a verificação do *fingerprint*. Discutimos, também, alguns aspectos de segurança do esquema proposto, apontando fraquezas e possíveis mitigações.

Na seção de avaliação experimental, testamos nossa proposta para uma aplicação de monitoramento nas Redes Elétricas Inteligentes. Realizamos a simulação dos três seguintes cenários: (*i*) considerando apenas a perda de pacotes na rede, (*ii*) sob ataque do tipo *Degrau*, e (*iii*) sob ataque do tipo *Alternado*. Como resultado, obtivemos um alto grau de validação e de detecção de alteração maliciosa nos dados. Para o cenário que considera apenas as perdas da rede, obtivemos entre 99,7% e 100% de validação dos *fingerprints*. Enquanto que nos cenários sob os ataques do tipo *Degrau* e *Alternado*, obtivemos 98,4% e 99,3%, respectivamente, de invalidação com sucesso dos *fingerprints*.

Como trabalhos futuros, consideramos buscar melhores formas de representação das características de um conjunto de dados, utilizando outros filtros e técnicas de extração de características. Consideramos, também, a avaliação do esquema para outros tipos de ataques e, também, para dados de outras aplicações nos Sistemas Ciberfísicos, como os dados de dispositivos médicos em aplicações de assistência à vida.

6 CONCLUSÃO

Nesta tese, apresentamos métodos de segurança da informação para a autenticação e validação de evidências digitais durante as etapas de armazenamento e transmissão em Sistemas Ciberfísicos que possuem dispositivos de recursos limitados. Como contribuições, descrevemos três propostas em relação à autenticação de evidências digitais, cada uma com foco específico em uma parte do sistema, são elas:

1. *Autenticação de evidências digitais armazenadas em dispositivos suscetíveis a ataques.* Apresentamos o esquema de estruturação SbC-EC para log seguro, apropriado para dispositivos com restrições de armazenamento e de comunicação em rede, incluindo duas novas propriedades: Separação por Categoria e Compactação de Entradas. Descrevemos, também, o esquema de log seguro SbC-EC MAC, que implementa a estruturação proposta.

2. *Autenticação de evidências digitais em trânsito.* Apresentamos o protocolo de autenticação atrasada inf-TESLA para comunicação *multicast*, apropriado para dar suporte às altas taxas de envio de dados em fluxo contínuo por um longo período de tempo. O mecanismo de geração e gerenciamento de chaves *Dual Offset Key Chains* que apoia o protocolo proposto também é desenvolvido.

3. *Validação externa de evidências digitais.* Apresentamos um esquema de uma *tag* de autenticação baseado em características para dados de monitoramento. Essa *tag* oferece não-repúdio e garantia de integridade, com um alto grau de correlação estatística, de um fluxo de dados transmitidos por um canal de comunicação suscetível a ataques e perdas de pacotes.

Nossos métodos foram avaliados de forma comparativa à outros métodos existentes na literatura, demostrando redução na sobrecarga de segurança necessária para implementação do método, além da redução do custo computacional, armazenamento interno e utilização da rede de comunicação. Além disso, eles foram avaliados, também, para aplicações de recursos limitados, em relação ao armazenamento interno, desempenho computacional e à rede de comunicação, no domínio dos Sistemas Ciberfísicos. Em específico, os sistemas de monitoramento das Redes Elétricas Inteligentes foram escolhidos como o cenário principal de avaliação.

Como trabalhos futuros, pretendemos o aperfeiçoamento dos métodos apresentados, adicionando as características indicadas em cada conclusão de capítulo. Além disso, dado o caráter genérico dos métodos aqui apresentados e considerando que eles podem ser adaptados a cenários similares que possuem restrições em relação aos recursos computacionais, pretendemos a avaliação desses métodos para outras aplicações no domínio dos Sistemas Ciberfísicos (p.ex., sistemas automotivos e de assistência à vida), ou até mesmo, em outros domínios, como o da Internet das Coisas.

REFERÊNCIAS

[1] ACCORSI, R. Safe-Keeping Digital Evidence with Secure Logging Protocols: state of the art and challenges. In: INTERNATIONAL CONFERENCE ON IT SECURITY INCIDENT MANAGEMENT AND IT FORENSICS, 5. **Anais...** [S.l.: s.n.], 2009. p.94–110.

[2] ACCORSI, R. BBox: a distributed secure log architecture. In: EUROPEAN CONFERENCE ON PUBLIC KEY INFRASTRUCTURES, SERVICES AND APPLICATIONS, 7., Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2011. p.109–124. (EuroPKI'10).

[3] ACCORSI, R. A secure log architecture to support remote auditing. **Mathematical and Computer Modelling**, [S.l.], v.57, n.7, p.1578–1591, 2013.

[4] ALI, S. T.; SIVARAMAN, V.; OSTRY, D. Authentication of lossy data in body-sensor networks for cloud-based healthcare monitoring. **Future Generation Computer Systems**, [S.l.], v.35, p.80–90, 2014.

[5] ANAND, M. et al. Security challenges in next generation cyber physical systems. In: BEYOND SCADA: CYBER PHYSICAL SYSTEMS MEETING (HCSS-NEC4CPS). **Anais...** [S.l.: s.n.], 2006. p.347–356.

[6] ANDERSON, R. et al. A new family of authentication protocols. **ACM SIGOPS Operating Systems Review**, [S.l.], v.32, p.9–20, 1998.

[7] Association Of Chief Police Officers' (ACPO). **Good Practice Guide for Computer-Based Electronic Evidence, Official release version 4.0**. [S.l.: s.n.], 2009.

[8] BARKER, E.; ROGINSKY, A. Recommendation for Transitioning the Use of

Cryptographic Algorithms and Key Lengths. **Special Publication 800-131A Transitions**, [S.l.], 2011.

[9] BASSO, S. et al. Estimating Packet Loss Rate in the Access Through Application-level Measurements. In: ACM SIGCOMM WORKSHOP ON MEASUREMENTS UP THE STACK, 2012., New York, NY, USA. **Proceedings...** ACM, 2012. p.7–12. (W-MUST '12).

[10] BELLARE, M.; CANETTI, R.; KRAWCZYK, H. Keying Hash Functions for Message Authentication. In: ANNUAL INTERNATIONAL CRYPTOLOGY CONFERENCE ON ADVANCES IN CRYPTOLOGY, 16., London, UK, UK. **Proceedings...** Springer-Verlag, 1996. p.1–15. (CRYPTO '96).

[11] BELLARE, M.; YEE, B. Forward-security in private-key cryptography. **Topics in Cryptology-CT-RSA 2003**, [S.l.], p.1–24, 2003.

[12] BELLARE, M.; YEE, B. S. **Forward Integrity For Secure Audit Logs**. [S.l.: s.n.], 1997.

[13] BYRES, E.; DR, P. E.; HOFFMAN, D. The myths and facts behind cyber security risks for industrial control systems. In: VDE KONGRESS. **Proceedings...** [S.l.: s.n.], 2004.

[14] CÂMARA, S. M.; CARMO, L. F. R. C.; PIRMEZ, L. Esquema de Estruturação SbC-EC para Log Seguro. In: SBSEG 2014 - ARTIGOS COMPLETOS / FULL PAPERS (), Belo Horizonte (MG). **Anais...** [S.l.: s.n.], 2014.

[15] CÂMARA, S. M. et al. Multicast Delayed Authentication for Streaming Synchrophasor Data in the Smart Grid. In: IFIP TC 11 INTERNATIONAL CONFERENCE, SEC 2016, 31., Ghent, Belgium. **Proceedings...** [S.l.: s.n.], 2016.

[16] CASEY, E. **Digital Evidence and Computer Crime**: forensic science, computers, and the internet. 3rd.ed. [S.l.]: Academic Press, 2011.

[17] CHAKRABORTTY, A.; KHARGONEKAR, P. P. Introduction to wide-area control of power systems. In: AMERICAN CONTROL CONFERENCE (ACC), 2013. **Anais...** [S.l.: s.n.], 2013. p.6758–6770.

[18] CHECKOWAY, S. et al. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In: USENIX CONFERENCE ON SECURITY, 20., Berkeley, CA, USA. **Proceedings...** USENIX Association, 2011. p.6–6. (SEC'11).

[19] CHISUM, J. W. **Crime reconstruction and evidence dynamics**. [S.l.]: Presented at the Academy of Behavioral Profiling Annual Meeting, 1999.

[20] Cisco ASA Series Syslog Messages. Online, acessado em 12/08/2016, `http://www.cisco.com/c/en/us/td/docs/security/asa/syslog-guide/syslogs/logmsgs1.html`.

[21] Common Criteria. **Common Criteria for Information Technology Security Evaluation Part 2**: security functional components. [S.l.: s.n.], 2009.

[22] CONTI, M. et al. Fast Track Article: looking ahead in pervasive computing: challenges and opportunities in the era of cyber-physical convergence. **Pervasive and Mobile Computing**, Amsterdam, The Netherlands, v.8, n.1, p.2–21, Feb. 2012.

[23] CROSBY, S. A.; WALLACH, D. S. Efficient Data Structures for Tamper-evident Logging. In: CONFERENCE ON USENIX SECURITY SYMPOSIUM, 18., Berkeley, CA, USA. **Proceedings...** USENIX Association, 2009. p.317–334. (SSYM'09).

[24] DAI, W. **Crypto++ 5.6.0 benchmarks**. 2009, `https://www.cryptopp.com/benchmarks.html`.

[25] DANIELS, D. J.; HART, S. V. **Forensic Examination of Digital Evidence : a guide for law enforcement**. [S.l.: s.n.], 2004. 634–111p. v.44, n.2.

[26] KEROMYTIS, A. D. (Ed.). **Efficient, Compromise Resilient and Append-Only Cryptographic Schemes for Secure Audit Logging**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p.148–163.

[27] ELLIOTT, E. O. Estimates of error rates for codes on burst-noise channels. **The Bell System Technical Journal**, [S.l.], v.42, n.5, p.1977–1997, Sept 1963.

[28] EROL-KANTARCI, M.; MOUFTAH, H. T. Smart grid forensic science: applications, challenges, and open issues. **IEEE Communications Magazine**, [S.l.], v.51, n.1, p.68–74, January 2013.

[29] Ext4 Disk Layout. Online, acessado em 12/08/2016, `https://ext4.wiki.kernel.org/index.php/Ext4_Disk_Layout`.

[30] FABRO, M.; CORNELIUS, E. **Recommended Practice: creating cyber forensics plans for control systems**. [S.l.]: U.S. Department of Homeland Security (DHS) National Cyber Security Division, 2008. (August).

[31] FALK, R.; FRIES, S. Security Considerations for Multicast Communication in Power Systems. **International Journal on Advances in Security**, [S.l.], v.6, n.3, p.111–121, 2013.

[32] FALLIERE, N.; MURCHU, L. O.; CHIEN, E. **W32.Stuxnet Dossier**. [S.l.: s.n.], 2011. (February 2011).

[33] FRANKEL, S.; KELLY, S. G. **Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec**. [S.l.]: RFC Editor, 2007. n.4868. (Request for Comments).

[34] GENNARO, R.; ROHATGI, P. How to Sign Digital Streams. In: ANNUAL INTERNATIONAL CRYPTOLOGY CONFERENCE ON ADVANCES IN CRYPTOLOGY, 17., London, UK. **Proceedings...** Springer-Verlag, 1997. p.180–197. (CRYPTO '97).

[35] GIESE, H. et al. Science and Engineering of Cyber-Physical Systems (Dagstuhl Seminar 11441). **Dagstuhl Reports**, Dagstuhl, Germany, v.1, n.11, p.1–22, 2012.

[36] GILBERT, E. N. Capacity of a burst-noise channel. **The Bell System Technical Journal**, [S.l.], v.39, n.5, p.1253–1265, Sept 1960.

[37] GOLLE, P.; MODADUGU, N. Authenticating Streamed Data in the Presence of Random Packet Loss (Extended Abstract). In: ISOC NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM. **Anais...** [S.l.: s.n.], 2001. p.13–22.

[38] GREER, C. et al. **NIST Framework and Roadmap for Smart Grid Interoperability Standards**. [S.l.]: National Institute of Standards and Technology, 2014.

[39] HABIB, A. et al. A Tree-Based Forward Digest Protocol to Verify Data Integrity in Distributed Media Streaming. **IEEE Transactions on Knowledge and Data Engineering**, Piscataway, NJ, USA, v.17, n.7, p.1010–1014, July 2005.

[40] HARTUNG, G. Secure Audit Logs with Verifiable Excerpts. In: RSA CONFERENCE ON TOPICS IN CRYPTOLOGY - CT-RSA 2016 - VOLUME 9610, New York, NY, USA. **Proceedings...** Springer-Verlag New York: Inc., 2016. p.183–199.

[41] HENSELER, J. **Computer crime and computer forensics**. [S.l.]: London: Academic Press, 2000.

[42] HOLT, J. E. Logcrypt: forward security and public verification for secure audit logs. In: AUSTRALASIAN WORKSHOPS ON GRID COMPUTING AND E-RESEARCH - VOLUME 54, 2006., Darlinghurst, Australia, Australia. **Proceedings...** Australian Computer Society: Inc., 2006. p.203–211. (ACSW Frontiers '06).

[43] IEEE Standard for Synchrophasor Measurements for Power Systems. **IEEE Std C37.118.1-2011 (Revision of IEEE Std C37.118-2005)**, [S.l.], p.1–61, Dec 2011.

[44] IEEE Standard for Synchrophasors for Power Systems. **IEEE Std C37.118-2005 (Revision of IEEE Std 1344-1995)**, [S.l.], p.1–57, 2006.

[45] International Electrotechnical Commission. **IEC TS 62351-6:2007 - power systems management and associated information exchange – data and communications security - part 6: security for iec 61850**. [S.l.: s.n.], 2007.

[46] International Electrotechnical Commission. **ISA - Security for Industrial Automation and Control Systems - Technical Security Requirements for IACS Components - Part 4**. Draft.

[47] International Electrotechnical Commission. **Communication networks and systems for power utility automation – Part 90-5: use of iec 61850 to transmit synchrophasor information according to ieee c37.118**. [S.l.: s.n.], 2012.

[48] International Telecommunication Union. **Information Technology — ASN.1 Encoding Rules — Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)**. 2002.

[49] JOHNSON, D.; MENEZES, A.; VANSTONE, S. The Elliptic Curve Digital Signature Algorithm (ECDSA). **International Journal of Information Security**, [S.l.], v.1, n.1, p.36–63, 2001.

[50] KANSAL, P.; BOSE, A. Bandwidth and latency requirements for smart transmission grid applications. **IEEE Transactions on Smart Grid**, [S.l.], v.3, n.3, p.1344–1352, 2012.

[51] KHAITAN, S. K.; MEMBER, S.; MCCALLEY, J. D. Design Techniques and Applications of Cyberphysical Systems : a survey. **IEEE Systems Journal**, [S.l.], v.9, n.2, p.350–365, 2015.

[52] KOSCHER, K. et al. Experimental Security Analysis of a Modern Automobile. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY (SP), 2010. **Anais...** [S.l.: s.n.], 2010. p.447–462.

[53] LAMPORT, L. Password authentication with insecure communication. **Communications of the ACM**, [S.l.], v.24, n.11, p.770–772, 1981.

[54] LAW, Y. W. et al. Comparative Study of Multicast Authentication Schemes with Application to Wide-area Measurement System. In: ACM SIGSAC SYMPOSIUM ON INFORMATION, COMPUTER AND COMMUNICATIONS SECURITY, 8., New York, NY, USA. **Proceedings...** ACM, 2013. p.287–298. (ASIA CCS '13).

[55] LEE, E. A. Cyber Physical Systems: design challenges. In: INTERNATIONAL SYMPOSIUM ON OBJECT/COMPONENT/SERVICE-ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC). **Anais...** [S.l.: s.n.], 2008. p.363–369.

[56] LI, Q.; CAO, G. Multicast authentication in the smart grid with one-time signature. **IEEE Transactions on Smart Grid**, [S.l.], v.2, p.686–696, 2011.

[57] LIU, D.; NING, P. Multilevel μTESLA: broadcast authentication for distributed sensor networks. **ACM Transactions on Embedded Computing Systems**, [S.l.], v.3, p.800–836, 2004.

[58] MA, D. I.; TSUDIK, G. A New Approach to Secure Logging. **ACM Transactions on Storage**, Berlin, Heidelberg, v.5, n.1, p.2:1—-2:21, 2009.

[59] MA, D.; TSUDIK, G. Extended Abstract: forward-secure sequential aggregate authentication. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 2007.,

Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2007. p.86–91. (SP '07).

[60] MA, D.; TSUDIK, G. A New Approach to Secure Logging. In: ANNUAL IFIP WG 11.3 WORKING CONFERENCE ON DATA AND APPLICATIONS SECURITY, 22., Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2008. p.48–63.

[61] MARTIN, K. et al. **Best Practice Recommendations for Synchrophasor Systems: administration, planning and implementation, and operation and maintenance**. [S.l.]: Eletric Power Group, 2013.

[62] MERKLE, R. C. A Certified Digital Signature. In: ADVANCES IN CRYPTOLOGY, New York, NY, USA. **Proceedings...** Springer-Verlag New York: Inc., 1989. p.218–238. (CRYPTO '89).

[63] MINER, S.; STADDON, J. Graph-based authentication of digital streams. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY. SP 2001, 2001. **Proceedings...** [S.l.: s.n.], 2001. p.232–246.

[64] Monitoring | EPFL Smart Grid. Online, acessado em 17/02/2017, `http://smartgrid.epfl.ch/?q=monitoring`.

[65] NASRALLA, M. M.; HEWAGE, C. T. E. R.; MARTINI, M. G. Subjective and objective evaluation and packet loss modeling for 3D video transmission over LTE networks. In: INTERNATIONAL CONFERENCE ON TELECOMMUNICATIONS AND MULTIMEDIA (TEMU), 2014. **Anais...** [S.l.: s.n.], 2014. p.254–259.

[66] National Institute of Standards and Technology (NIST). **FIPS PUB 197 – Announcing the Advanced Encryption Standard (AES)**. [S.l.: s.n.], 2001.

[67] National Institute of Standards and Technology (NIST). **FIPS PUB 180-4 – Secure Hash Standard (SHS)**. [S.l.: s.n.], 2015.

[68] National Science Foundation (NSF). **Cyber Physical Systems NSF 10-515**. Arlington, VA, USA: [s.n.], 2013.

[69] North American SynchroPhasor Initiative (NASPI). Online, acessado em 20/02/2016, `https://www.naspi.org/`.

[70] North American SynchroPhasor Initiative (NASPI). **Model Validation Using Phasor Measurement Unit Data**. [S.l.]: NASPI Technical Report, 2015.

[71] OPREA, A.; BOWERS, K. D. Authentic Time-stamps for Archival Storage. In: EUROPEAN CONFERENCE ON RESEARCH IN COMPUTER SECURITY, 14., Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2009. p.136–151. (ESORICS'09).

[72] PANNETRAT, A.; MOLVA, R. Efficient multicast packet authentication. In: NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM (NDSS), San Diego, California. **Proceedings...** [S.l.: s.n.], 2003.

[73] PARK, J. M.; CHONG, E. K. P.; SIEGEL, H. J. Efficient multicast stream authentication using erasure codes. **ACM Transactions on Information and System Security**, [S.l.], v.6, n.2, p.258–285, 2003.

[74] PATEL, M.; AIVALIOTIS, S.; ALLEN, E. **Real-time application of synchrophasors for improving reliability**. [S.l.]: North American Electric Reliability Corporation (NERC), 2010.

[75] Pearson Product-Moment Correlation. Online, acessado em 03/03/2017, `https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php`.

[76] PERRIG, A. The BiBa One-time Signature and Broadcast Authentication Protocol. In: ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY, 8., New York, NY, USA. **Proceedings...** ACM, 2001. p.28–37. (CCS '01).

[77] PERRIG, A. et al. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 2000., Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2000. p.56–. (SP '00).

[78] PERRIG, A. et al. Efficient and Secure Source Authentication for Multicast. In: NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM, NDSS '01. **Anais...** [S.l.: s.n.], 2001. p.35–46.

[79] PERRIG, A. et al. SPINS: security protocols for sensor networks. **Wireless networks**, [S.l.], v.8, n.5, p.521–534, 2002.

[80] PERRIG, A. et al. The TESLA broadcast authentication protocol. **CryptoBytes**, [S.l.], v.Summer/Fall, p.2–13, 2002.

[81] PERRIG, A. et al. **Timed efficient stream loss-tolerant authentication (TESLA)**: multicast source authentication transform introduction. [S.l.]: The Internet Society, 2005. RFC. (4082).

[82] PIGNATI, M. et al. Real-Time State Estimation of the EPFL-Campus Medium-Voltage Grid by Using PMUs. In: CONFERENCE ON INNOVATIVE SMART GRID TECHNOLOGIES (ISGT2015), 6. **Anais...** [S.l.: s.n.], 2015.

[83] POSTEL, J. **Internet Protocol - DARPA Internet Programm, Protocol Specification**. [S.l.: s.n.], 1981. RFC. (791).

[84] RAGGETT, D. The Web of Things: challenges and opportunities. **Computer**, [S.l.], v.48, n.5, p.26–32, May 2015.

[85] RAGHAVAN, S. Digital forensic research: current state of the art. **CSI Transactions on ICT**, [S.l.], v.1, n.1, p.91–114, 2012.

[86] RAJALAKSHMI, J. R.; RATHINRAJ, M.; BRAVEEN, M. Anonymizing log management process for secure logging in the cloud. In: INTERNATIONAL

CONFERENCE ON CIRCUITS, POWER AND COMPUTING TECHNOLOGIES, ICCPCT 2014, 2014. **Anais...** [S.l.: s.n.], 2015. p.1559–1564.

[87] REYZIN, L.; REYZIN, N. Better than BiBa: short one-time signatures with fast signing and verifying. **Information Security and Privacy**, [S.l.], v.2384, p.1–47, 2002.

[88] SCHNEIER, B.; KELSEY, J. Cryptographic Support for Secure Logs on Untrusted Machines. In: CONFERENCE ON USENIX SECURITY SYMPOSIUM - VOLUME 7, 7., Berkeley, CA, USA. **Proceedings...** USENIX Association, 1998. p.4. (SSYM'98).

[89] SCHNEIER, B.; KELSEY, J. Secure audit logs to support computer forensics. **ACM Transactions on Information and System Security**, [S.l.], v.2, n.2, p.159–176, 1999.

[90] SHA, L. et al. Cyber-Physical Systems: a new frontier. In: SENSOR NETWORKS, UBIQUITOUS AND TRUSTWORTHY COMPUTING, 2008. SUTC '08. IEEE INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2008. p.1–9.

[91] SLAY, J.; MILLER, M. Lessons Learned from the Maroochy Water Breach. In: CRITICAL INFRASTRUCTURE PROTECTION. **Anais...** Springer US, 2008. p.73–82. (IFIP International Federation for Information Processing, v.253).

[92] SMITH, S. W. **The Scientist and Engineer's Guide to Digital Signal Processing**. San Diego, CA, USA: California Technical Publishing, 1997.

[93] SOMMER, P. Downloads, logs and captures: evidence from cyberspace. **Journal of Financial Crime**, [S.l.], v.5JFC2, p.138–152, October 1997.

[94] SONG, D.; ZUCKERMAN, D.; TYGAR, J. D. Expander graphs for digital stream authentication and robust overlay networks. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 2002. **Proceedings...** [S.l.: s.n.], 2002. p.258–270.

[95] SPYRIDOPOULOS, T.; TRYFONAS, T.; MAY, J. Incident analysis amp; digital forensics in SCADA and industrial control systems. In: SYSTEM SAFETY CONFERENCE INCORPORATING THE CYBER SECURITY CONFERENCE 2013, 8TH IET INTERNATIONAL. **Anais...** [S.l.: s.n.], 2013. p.1–6.

[96] STUDER, A. et al. Flexible, extensible, and efficient VANET authentication. **Journal of Communications and Networks**, [S.l.], v.11, p.574–588, 2009.

[97] Trending and Historian Software Information. Online, acessado em 17/02/2017, `http://www.globalspec.com/learnmore/industrial_engineering_software/industrial_controls_software/trending_historian_software`.

[98] TUFFNER, F. **Phasor Measurement Unit Application Data Requirements**. [S.l.]: Pacific Northwest National Laboratory, 2014.

[99] UGUS, O.; WESTHOFF, D.; BOHLI, J.-M. A ROM-friendly secure code update mechanism for WSNs using a stateful-verifier τ-time signature scheme. In: ACM CONFERENCE ON WIRELESS NETWORK SECURITY. **Proceedings...** [S.l.: s.n.], 2009. p.29–40.

[100] Union for the Coordination of the Transmission of Electricity (UCTE). **Final Report of the Investigation Committee on the 28 September 2003 Blackout in Italy**. [S.l.: s.n.], 2004. (April).

[101] U.S.-Canada Power System Outage Task Force. **Final report on the August 14, 2003 blackout in the United States and Canada**: causes and recommendations. [S.l.: s.n.].

[102] WANG, C. et al. Privacy-Preserving Public Auditing for Secure Cloud Storage. **IEEE Transactions on Computers**, [S.l.], v.62, n.2, p.362–375, Feb 2013.

[103] WANG, Q. et al. Time valid one-time signature for time-critical multicast data authentication. In: INFOCOM 2009, IEEE. **Anais...** [S.l.: s.n.], 2009. p.1233–1241.

[104] WATERS, B. et al. Building an Encrypted and Searchable Audit Log. In: ANNUAL NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM, 11. **Anais...** [S.l.: s.n.], 2004.

[105] WONG, C. K.; LAM, S. S. Digital signatures for flows and multicasts. **IEEE/ACM Transactions on Networking**, [S.l.], v.7, n.4, p.502–513, 1999.

[106] XU, W.; CHADWICK, D.; OTENKO, S. A PKI-Based Secure Audit Web Service. In: IASTED COMMUNICATIONS, NETWORK AND INFORMATION AND CNIS. **Anais...** [S.l.: s.n.], 2005.

[107] YAMPOLSKIY, M. et al. A language for describing attacks on cyber-physical systems. **International Journal of Critical Infrastructure Protection**, [S.l.], v.8, p.40–52, 2015.

[108] YAVUZ, A. A.; NING, P. BAF: an efficient publicly verifiable secure audit logging scheme for distributed systems. In: COMPUTER SECURITY APPLICATIONS CONFERENCE, 2009. ACSAC '09. ANNUAL. **Anais...** [S.l.: s.n.], 2009. n.ii, p.219–228.

[109] YAVUZ, A. a.; NING, P.; REITER, M. K. BAF and FI-BAF: efficient and publicly verifiable cryptographic schemes for secure logging in resource-constrained systems. **ACM Transactions on Information and System Security**, New York, NY, USA, v.15, n.2, p.1–28, July 2012.

[110] ZAWOAD, S.; DUTTA, A. K.; HASAN, R. SecLaaS: secure logging-as-a-service for cloud forensics. In: ACM SIGSAC SYMPOSIUM ON INFORMATION, COMPUTER AND COMMUNICATIONS SECURITY, 8., New York, NY, USA. **Proceedings...** ACM, 2013. p.219–230. (ASIA CCS '13).

[111] ZHANG, Z.; SUN, Q.; WONG, W. C. A proposal of butterfly-graph based stream authentication over lossy networks. In: IEEE INTERNATIONAL CON-

FERENCE ON MULTIMEDIA AND EXPO, ICME 2005. **Anais...** [S.l.: s.n.], 2005. v.2005, p.784–787.

[112] ZHU, K.; NORDSTRöM, L.; AL-HAMMOURI, A. T. Examination of data delay and packet loss for wide-area monitoring and control systems. In: IEEE INTERNATIONAL ENERGY CONFERENCE AND EXHIBITION (ENERGYCON), 2012. **Anais...** [S.l.: s.n.], 2012. p.927–934.