

UNIVERSIDADE FEDERAL DO RIO GRANDE
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

IoT Tunnel - Uma proposta para mitigar vulnerabilidades na comunicação de elementos IoT

Matheus Müller dos Santos

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Paulo Francisco Butzen
Coorientador: Prof. Dr. Vagner Santos da Rosa

Rio Grande, 2020

Ficha catalográfica

S237i Santos, Matheus Müller.

IoT Tunnel: uma proposta para mitigar vulnerabilidades na comunicação de elementos IoT / Matheus Müller Santos. – 2019. 100 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2019.

Orientador: Dr. Paulo Francisco Butzen.

Coorientador: Dr. Vagner Santos da Rosa.

1. Internet das Coisas 2. Honeypot 3. Segurança
4. Vulnerabilidade I. Butzen, Paulo Francisco II. Rosa, Vagner Santos da III. Título.

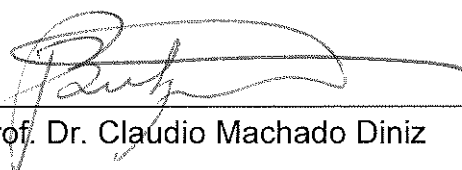
CDU 004.738.5

DISSERTAÇÃO DE MESTRADO

**IoT Tunnel – Uma proposta para mitigar a vulnerabilidade na
comunicação de elementos IoT**

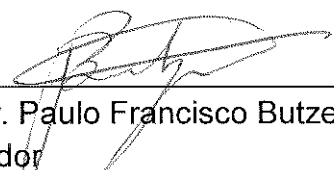
Matheus Müller Santos

Banca examinadora:

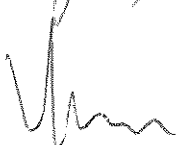

P/ _____
Prof. Dr. Claudio Machado Diniz



Prof. Dr. Eduardo Nunes Borges



Prof. Dr. Paulo Francisco Butzen
Orientador



Prof. Dr. Vagner Santos da Rosa
Coorientador

Dedico esse trabalho a minha filha Amora que nasceu junto com ele.

AGRADECIMENTOS

Este trabalho não seria possível sem a ajuda direta ou indireta de diversas pessoas que estiveram e/ou estão presentes em minha vida. Inicialmente gostaria de agradecer meus pais por todo apoio e incentivo que sempre me deram desde sempre até os dias atuais. A dedicação em prover tudo que eu sempre precisei bem como o amor incondicional e os valores que carrego sempre comigo foram fundamentais para me manter nos trilhos e me fazer chegar onde cheguei. Outra pessoa que não poderia deixar de agradecer é a Érika Sanchez minha parceira e amiga que dedicou meses de cuidados a nossa filha Amora suportando minhas ausências para dedicação a esse trabalho. Agradeço ao meu eterno orientador Vagner Rosa por toda sua dedicação e amizade nos últimos 5 anos. Seus conselhos, visão aguçada e preocupação comigo foram imprescindíveis para mais essa conquista. Gostaria de agradecer aos meus sócios Fernando Toledo, Ismael Altenhofen, Jonas Casarin e Marcos Fagundes pela parceria, pelo apoio, compreensão e por terem segurado as pontas sempre que não pude estar presente. Agradeço ao Paulo Butzen pela dedicação, tempo, conselhos e conversas valiosas. Por último gostaria de agradecer a todos meus amigos que sempre me ajudaram nos momentos difíceis de estresse, angústia, preocupação e quando me sentia perdido. Obrigado por estarem perto de mim.

“Love is wise; hatred is foolish. In this world, which is getting more and more closely interconnected, we have to learn to tolerate each other, we have to learn to put up with the fact that some people say things that we don’t like. We can only live together in that way. But if we are to live together, and not die together, we must learn a kind of charity and a kind of tolerance, which is absolutely vital to the continuation of human life on this planet.”

— BERTRAND RUSSELL

RESUMO

MÜLLER DOS SANTOS, Matheus. **IoT Tunnel - Uma proposta para mitigar vulnerabilidades na comunicação de elementos IoT**. 2020. 103 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

A miniaturização da tecnologia e um mundo interconectado deu origem a um novo paradigma onde sistemas computacionais circundam quem os utiliza de forma a estarem sempre presentes em suas vidas e por toda parte, interagindo entre si e com o homem, muitas vezes de forma imperceptível. Esse paradigma foi possível devido a capacidade que itens do nosso cotidiano, integrando-se com elementos computacionais, obtiveram em interagir mutuamente e com o meio físico através de redes computacionais a fim de cooperar uns com os outros para alcançar objetivos comuns. Este modelo de tecnologia popularizou-se com o nome de internet das coisas (IoT) e possibilitou diversas novas abordagens e conquistas, nos mantendo mais conectados e facilitando nosso dia-a-dia. No entanto, a constante comunicação que é necessária para o funcionamento desses elementos os expõe a riscos os quais vem sendo muito explorados para atividades maliciosas das mais diversas por apresentarem vulnerabilidades em sua comunicação que podem ser utilizadas para esses fins. Tratando-se de elementos que estão presentes constantemente em nossas vidas privadas e por muitas vezes nem mesmo serem percebidos, garantir a segurança desses elementos se torna essencial. Desta forma, este trabalho tem como objetivo propor um sistema que seja capaz de mitigar as vulnerabilidades dessa comunicação reduzindo os riscos de se utilizar essa tecnologia. Para isso, inicialmente utilizou-se um *honeypot*, um sistema capaz capturar tentativas de acesso para a identificação do comportamento de atacantes. Por um período de 11 meses foi possível coletar dados que demonstraram quais serviços sofrem mais tentativas de exploração. Baseando-se nesses resultados foi então proposto o IoT Tunnel, um sistema capaz de mitigar os riscos em se comunicar com elementos IoT que estejam expostos a internet. Através do tunelamento dessa comunicação utilizando protocolos seguros, foi possível além de reduzir os vetores de ataque aos mesmos, aprender com os ataques sofridos. O modelo proposto apresenta diversos módulos que possibilitam a mitigação de algumas formas de exploração a serviços de acesso remoto presentes e muito comuns nesses elementos expostos na internet, fornecendo acesso a clientes externos de forma transparente e mais segura. O modelo experimental desenvolvido possui alguns módulos do modelo proposto e utilizou um estudo de caso real como base. Com isso comprovou-se a efetividade do modelo em gerar uma comunicação segura e transparente ao usuário e ainda capaz de identificar um comportamento malicioso além de capturar esse comportamento para análise posterior.

Palavras-chave: Internet das Coisas, Honeypot, Segurança, Vulnerabilidade.

ABSTRACT

MÜLLER DOS SANTOS, Matheus. **IoT Tunnel - A proposal to mitigate vulnerabilities in the communication of IoT elements**. 2020. 103 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

The miniaturization of technology and an interconnected world has given rise to a new paradigm where computer systems surround those who use them in a way that they are always present in their lives and everywhere, interacting with each other and with the man, often imperceptible. This paradigm was possible due to the ability that items of our daily life, integrating with computational elements, obtained in interacting with the physical environment and with each other through computer networks, cooperating to reach common goals. Internet of things (IoT), as we named it, is a technology model that has enabled us with many new approaches and achievements to keep us more connected and making our daily lives easier. However, the constant communication that is necessary for the functioning of these elements exposes them to risks that have been widely exploited for various malicious activities, as the present vulnerabilities in their communication that can be exploited for these purposes. Dealing with elements that are constantly present in our lives and privacy and often not even perceived, ensuring the security of these elements becomes essential. Thus this work aims to propose a system that is able to mitigate the vulnerabilities of this communication reducing the risks of using this technology. For this, initially a honeypot was used, a system capable of capturing all access attempts in order to identify the behavior of attackers. For a period of 11 months it was possible to collect data that demonstrated the most explored services. Based on these results, IoT Tunnel was proposed, a system capable of mitigating the risks of communicating with IoT elements that are exposed to the Internet by tunneling this communication using secure protocols, reducing attack vectors and learning from attacks. The proposed model presents several modules that allow the mitigation of some forms of exploitation to remote access services present in these elements exposed to the Internet, providing access to external clients in a transparent and more secure way. The experimental model developed has some modules of the initial proposed model and used a real case study as a basis. This proved the effectiveness of the model in generating safe and transparent communication to the user and is still able to identify malicious behavior and capture that behavior for further analysis.

Keywords: Internet of Things, Honeypot, Security, Vulnerability.

LISTA DE FIGURAS

Figura 1	Interação do domínio físico com domínio virtual. Fonte:(LABORATORY, 2017)	18
Figura 2	A queda do custo do processamento ao longo dos anos. Fonte: (SINGULARITY, 2017)	20
Figura 3	O custo médio anual do cibercrime por país. Fonte: (PONEMON, 2015)	22
Figura 4	O custo médio anual do cibercrime por país . Fonte: (PONEMON, 2019)	23
Figura 5	Previsão do crescimento de elementos IoT conectados no mundo. Fonte: (ANALYTICS, 2018)	23
Figura 6	Camadas de um elemento IoT. Fonte: (CHEN et al., 2018)	30
Figura 7	Categorização das vulnerabilidades da IoT.	32
Figura 8	Representação de máquinas virtuais rodando sobre um sistema hospedeiro. Fonte: (DOCKER, 2019)	38
Figura 9	Representação de containers rodando sobre um sistema hospedeiro. Fonte: (DOCKER, 2019)	39
Figura 10	Representação de um <i>firewall</i> . Fonte: (IPWITHEASE, 2019)	41
Figura 11	Representação de um IDS. Fonte: (IPWITHEASE, 2019)	41
Figura 12	Representação de um IPS. Fonte: (IPWITHEASE, 2019)	42
Figura 13	Demonstração de uma sessão Telnet. Fonte: (BURANDE et al., 2014)	43
Figura 14	Demonstração de uma sessão SSH. Fonte: (BURANDE et al., 2014)	44
Figura 15	Topologia de uma VPN. Fonte: (HACKOHACKU, 2019)	47
Figura 16	Exemplo de assinatura da ferramenta de varredura de vulnerabilidades Nessus, em um log de sistema.	49
Figura 17	Visão geral do IoT POT. Fonte:(PA et al., 2015)	53
Figura 18	Topologia interna do sistema IoT POT. Fonte:(PA et al., 2015)	54
Figura 19	Visão geral do SIPHON. Fonte: (GUARNIZO et al., 2017)	55
Figura 20	Locais onde foram colocados wormholes. Fonte: (GUARNIZO et al., 2017)	56
Figura 21	Países de origem dos ataques. Fonte: (GUARNIZO et al., 2017)	57
Figura 22	Portas mais atacadas. Fonte: (GUARNIZO et al., 2017)	57
Figura 23	Principais portas atacadas no dia 15/11/2017. Fonte: (PROJECT, 2019)	59
Figura 24	Principais sistemas operacionais os quais se originaram os ataques no dia 15/11/2017. Fonte: (PROJECT, 2019)	59
Figura 25	Visão geral do HIoTPOT. Fonte: (GANDHI et al., 2018)	61

Figura 26	Comparação do HIoT POT com sistemas existentes. Fonte: (GANDHI et al., 2018)	61
Figura 27	Configuração da rede usada para avaliação inicial do sistema proposto. Fonte: (FOROUSHANI; ADIBNIA; HOJATI, 2008)	63
Figura 28	Arquitetura interna do frontend do Securebox. Fonte: (HAFEEZ et al., 2016)	65
Figura 29	Esquemático do ambiente utilizado no <i>honeypot</i>	69
Figura 30	Esquemático comunicação cliente-servidor para identificação de IPs.	71
Figura 31	Ataques distribuídos ao longo dos meses do ano de 2018	73
Figura 32	Ataques distribuídos ao longo dos meses do ano de 2018 (Corrigido)	74
Figura 33	Variação de incidência de ataques durante os dias de execução	74
Figura 34	Distribuição de ataques pelos dias da semana	75
Figura 35	Incidência de tipos de pacote.	75
Figura 36	As 15 principais portas atacadas	76
Figura 37	Posição geográfica de atacantes	76
Figura 38	Visão geral do sistema proposto	79
Figura 39	Módulos do sistema proposto	81
Figura 40	Sistema intermediando comunicação de um elemento IoT com uma aplicação cliente.	83
Figura 41	Sistema bloqueando serviços desnecessários e inseguros ao meio externo e expondo um serviço necessário e seguro.	84
Figura 42	Sistema intermediando comunicação com interface web.	84
Figura 43	Wireless IP Camera 360. Fonte: (GARCIA, 2018)	86
Figura 44	Escaneamento de portas Wireless IP Camera 360. Fonte: (GARCIA, 2018)	87
Figura 45	Acesso Telnet na porta 9527. Fonte: (GARCIA, 2018)	88
Figura 46	Lista de comandos disponíveis. Fonte: (GARCIA, 2018)	88
Figura 47	Vazamento do <i>token</i> de acesso RTSP. Fonte: (GARCIA, 2018)	89
Figura 48	Topologia do experimento.	90
Figura 49	Visão geral do sistema experimental.	90
Figura 50	Captura de tráfego com conexão Telnet.	92
Figura 51	Captura de tráfego com conexão SSH.	93
Figura 52	Captura de comandos em tempo de execução.	94
Figura 53	Log de comandos gerado pelo honeypot.	94

LISTA DE TABELAS

Tabela 1	Comparação entre Telnet e SSH	45
Tabela 2	Porcentagem de acessos às portas capturados aleatoriamente no ano de 2018	60
Tabela 3	Comparativo de funcionalidades entre trabalhos	66
Tabela 4	Dados Coletados no Honeypot	70
Tabela 5	Dados de retorno de verificação de IP na API	72
Tabela 6	Comparativo de funcionalidades: trabalhos relacionados e IoT Tunnel	85

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
ARM	Advanced RISC Machine
ARP	Address Resolution Protocol
ARPANET	Advanced Research Projects Agency Network
ARPD	Address Resolution Protocol Daemon
CERT	Coordination Center Of The Computer Emergency Response Team
CIA-Triad	Confidentiality, Integrity and Availability Triad
CPS	Cyber Physical System
CPU	Central Processing Unit
CVE	Common Vulnerabilities and Exposures
DDoS	Distributed Denial of Service
DoS	Denial Of Service
EUA	Estados Unidos da América
IAS-Octave	Information Assurance and Security Octave
ICMP	Internet Control Message Protocol
IDC	International Data Corporation
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
LAN	Local Area Network
M2M	Machine to Machine
MIPS	Microprocessor Without Interlocked Pipeline Stages
NAT	Network Address Translation
OWASP	Open Web Application Security Project
PC	Personal Computer

PPC	Performance Optimization With Enhanced RISC – Performance Computing
REXEC	Remote Execution
RFID	Radio Frequency Identification
RSH	Remote Shell
RTSP	Real Time Streaming Protocol
SF	Securebox Frontend
SHA	Secure Hash Algorithm
SMS	Security and Management Service
SSH	Secure Shell
SSID	Service Set Identifier
TCP	Transmission Control Protocol
Telnet	Teletype Network
TI	Tecnologia da Informação
TV	Televisor
UDP	User Datagram Protocol
VM	Virtual Machine
VPN	Virtual Private Network

SUMÁRIO

1	INTRODUÇÃO	17
1.1	Cibercrime	19
1.2	O Futuro da Internet das Coisas e a Cibersegurança	22
1.3	Motivação	25
1.4	Objetivos	26
1.4.1	Objetivo Geral	27
1.4.2	Objetivos Específicos	27
1.5	Organização do trabalho	27
2	FUNDAMENTAÇÃO TEÓRICA	29
2.1	Taxonomia da IoT	29
2.1.1	Vulnerabilidades da IoT	30
2.2	Atributos de um sistema seguro e suas ameaças	35
2.3	Virtualização	37
2.3.1	Máquinas Virtuais	38
2.3.2	Containers	39
2.3.3	Sandboxing	40
2.4	Controle de Acesso	40
2.4.1	Firewall	40
2.4.2	IDS	41
2.4.3	IPS	42
2.5	Protocolos de Acesso Remoto	42
2.5.1	Telnet	42
2.5.2	SSH	43
2.5.3	Comparando Telnet e SSH	45
2.6	Comunicação Segura	45
2.6.1	Criptografia	45
2.6.2	VPN	46
2.7	Honeypots	46
2.7.1	Honeypots de baixa interatividade	48
2.7.2	Honeypots de alta interatividade	49
2.7.3	Honeypots físicos	50
2.7.4	Honeypots virtuais	50
2.7.5	Comparando honeypots físicos e virtuais	50
2.7.6	Honeynets	51

3	TRABALHOS RELACIONADOS	52
3.1	IoTPOT	52
3.2	SIPHON	55
3.3	Projeto honeyTARG Honeynet	58
3.4	HIoTPOT	60
3.5	Intrusion Detection in Encrypted Accesses with SSH Protocol to Network Public Servers	62
3.6	Securebox: Toward Safer and Smarter IoT Networks	64
3.7	Relação de funcionalidades	66
4	HONEYPOT UTILIZADO PARA OBTENÇÃO DE DADOS DE ATAQUES LOCAIS	68
4.1	Honeyd	68
4.2	Montagem do ambiente	69
4.3	Implementação de um sistema para coleta e análise de dados de ataques	70
4.4	Tratamento dos dados	71
4.5	Resultados obtidos	73
4.6	Conclusão	76
5	IOT TUNNEL	78
5.1	Arquitetura	79
5.2	Aplicações do modelo proposto	82
5.3	Relação de funcionalidades com trabalhos relacionados	85
5.4	Estudo de caso	86
5.5	Implementação	89
5.6	Resultados experimentais com dados sintéticos	91
5.7	Conclusão do estudo de caso	94
5.8	Limitações	95
6	CONCLUSÃO	96
6.1	Trabalhos Futuros	97
	REFERÊNCIAS	98

1 INTRODUÇÃO

Nos anos 70 com o surgimento do microprocessador e as melhorias nas tecnologias de circuitos integrados atingimos um crescimento de desempenho de processamento de cerca de 35% ao ano (HENNESSY; PATTERSON, 2011). Com a miniaturização do transistor além do aumento de desempenho, foi possível construir circuitos de tamanho gradativamente mais reduzidos. A possibilidade da inserção de grandes quantidades de componentes eletrônicos em pequenas áreas de silício viabilizou a criação de processadores minúsculos e com baixíssimo consumo de energia (NASCIMENTO; PIETZ; WATANABE, 2015). Paralelamente à evolução dos computadores, a necessidade da transmissão de dados levou a criação do que conhecemos hoje por Internet. A Internet ou Advanced Research Projects Agency Network (ARPANET) como antes era conhecida e que inicialmente interligava apenas computadores militares nos Estados Unidos da América (EUA), desde os anos 90 vem tendo um crescimento exponencial. Nesta época a internet começou a ser utilizada comercialmente a nível mundial, chegando hoje a atingir mais de 3,5 bilhões de usuários. Esses fatores geraram um mundo onde computadores, sensores, atuadores e sistemas embarcados podem se conectar e trabalhar em conjunto através de uma rede a fim de executar tarefas específicas. Recentemente, os termos ubíquo e pervasivo vem sendo empregados para definir características da tecnologia que representa um novo paradigma computacional, onde sistemas computacionais circundam quem os utiliza de forma a estarem sempre presentes em suas vidas e por toda parte, interagindo entre si e com o homem, muitas vezes de forma imperceptível (ATZORI; IERA; MORABITO, 2010). Este novo padrão, onde a presença destes elementos que interligam o mundo real e o virtual, passa a fazer parte do nosso cotidiano dando origem ao que chamamos de computação ubíqua e pervasiva; ubíqua de disponível em todos os lugares e pervasiva de tão integrado com as atividades diárias que dificilmente é notado (KUKULSKA-HULME, 2007). Seguindo essa tendência, onde objetos do cotidiano passaram a integrar a rede de computadores, podemos observar o surgimento do que conhecemos hoje por Internet das Coisas (Internet of Things ou IoT) ou seja, objetos físicos que se conectam à rede para compartilhar informações sobre si mesmos e seus arredores (GIUSTO et al., 2010). Internet das Coisas é um paradigma tecnológico que vem se popularizando cada vez mais

desde o início dos anos 90, quando ela dava ainda seus primeiros passos (WEISER, 1991) (GALEGALE et al., 2016). A ideia básica deste conceito é de proporcionar para elementos do nosso cotidiano, a capacidade de interagir com o meio físico e também pela capacidade de interagirem entre si através de redes de computadores, cooperando uns com os outros para alcançar objetivos comuns. Esta interação ocorre com base em dois domínios: domínio humano e físico e domínio virtual.

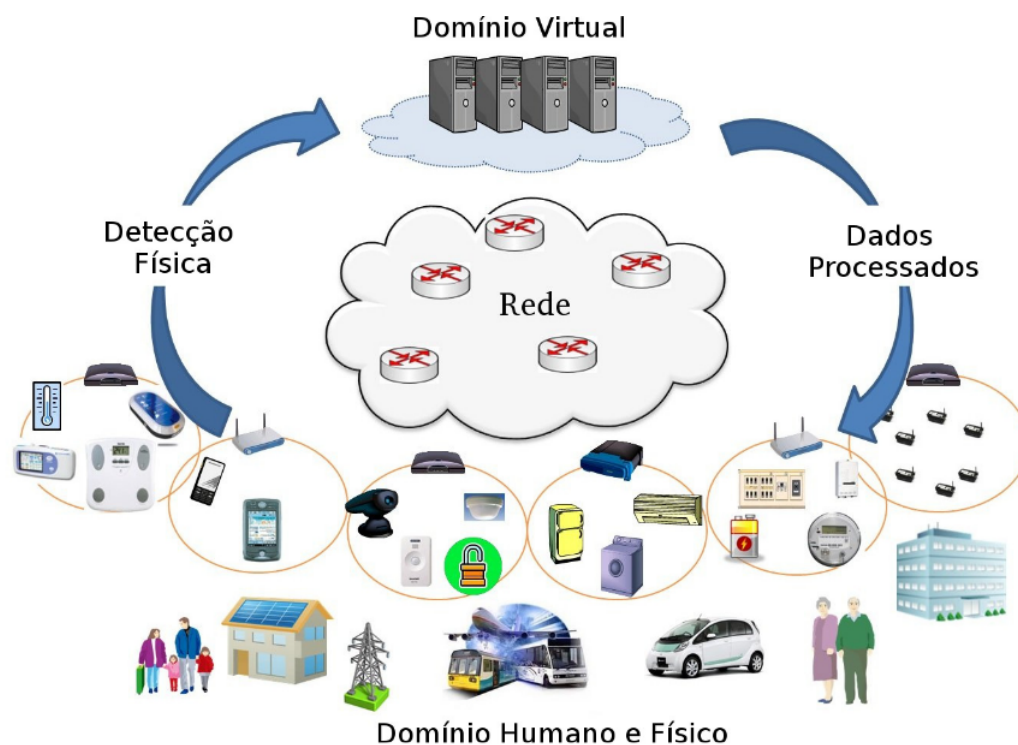


Figura 1: Interação do domínio físico com domínio virtual. Fonte:(LABORATORY, 2017)

A Figura 1 demonstra a interação do domínio físico, onde estamos nós humanos e o ambiente onde encontram-se os objetos que nos rodeiam no mundo real. O domínio virtual é representado pela nuvem, onde as informações extraídas do domínio físico são processadas. Entre os dois domínios estão os sensores, atuadores, sistemas embarcados e similares. Eles são os responsáveis pelo processo de comunicação entre ambas as partes no processo, ou seja, trocando informações obtidas no domínio físico para serem transportadas através da rede, processadas pelo domínio virtual a fim de realizar um trabalho para então serem devolvidas ao usuário de maneira a atingir um objetivo específico. Este diagrama apresenta algumas das características básicas da IoT, um sistema que apresenta uma rede altamente distribuída de dispositivos que se comunicam com seres humanos, bem como com outros dispositivos (XIA et al., 2012). Essa visão é importante para deixar claro que a quantidade de protocolos envolvidos na sua comunicação varia de acordo com a diversidade de tecnologias envolvidas e características da implementação do sistema. A “internet das coisas” é composta de máquinas que estão não apenas interagindo com

peessoas como também conversando umas com as outras em um modelo chamado M2M (Machine to machine - ou de máquina para máquina) e vão desde eletrodomésticos comuns como TVs e geladeiras até carros, medidores inteligentes, monitores de saúde e dispositivos vestíveis (*wearables*). A IoT promete uma comodidade incomparável. Porém, para que a IoT atinja seu potencial, é essencial conquistar e manter a confiança do consumidor sobre questões como privacidade e segurança. Os dados transmitidos pela IoT formarão uma representação gráfica de cada um de nós. O grande desafio será proteger essas informações. As ameaças na comunicação desses sistemas são em essência ameaças de segurança em TI, como ataques direcionados ao tráfego da rede, ataques infectando nodos com malware e acesso indevido. Com isso, sua defesa segue a mesma linha da aplicada em TI, como firewalls, detecção de intrusão, autenticação, controle de acesso, criptografia, etc (GOLLMANN; KROTOFIL, 2016). Considerando então estes sistemas, em parte uma rede de elementos computacionais (LEE, 2008), para preservarmos sua segurança, os atributos relacionados a segurança da informação devem ser garantidos. Este modelo de tecnologia nos possibilitou diversas novas abordagens e conquistas nos mantendo mais conectados e facilitando nosso dia-a-dia. No entanto, a necessidade de comunicação constante e a quantidade de informação extraída e compartilhada nestes sistemas acarreta em riscos. Constantemente presenciamos o crescimento do cibercrime e exploração de vulnerabilidades se valendo de brechas em sistemas computacionais. A ideia de estarmos gradativamente mais conectados e com sistemas heterogêneos, cada vez mais nos expõem a possibilidade de ataques, os quais vêm se tornando a tônica em sistemas interconectados (CARDENAS et al., 2009).

1.1 Cibercrime

Muitos dos problemas de segurança da informação com que lidamos hoje em dia retratam um mundo globalizado, computadorizado e conectado. A evolução dos computadores, e características como, por exemplo, a sua miniaturização, aumento de desempenho, barateamento e eficiência energética (BLAAUW; BROOKS JR, 1997), nos levaram a tecnologias pervasivas e ubíquas. Além disso a evolução possibilitou que cada vez mais usuários tenham acesso a estas tecnologias, visto que cada vez mais, os dispositivos eletrônicos, como *smartphones*, tablets e gadgets no geral, vem se tornando baratos e indispensáveis na vida do ser humano. É importante salientar que o crescimento de usuários na internet ao longo dos anos possibilitou a grande maioria dos ataques de exploração de vulnerabilidades atuais pela possibilidade de efetuar um ataque partindo de qualquer lugar e direcionado a qualquer lugar no mundo, e às vezes de uma forma muito difícil de detectar o ataque e identificar o atacante.

A vasta quantidade de dispositivos presentes nas redes, seguindo o modelo de “sempre conectado”, bem como a grande capacidade de processamento destes sistemas, municiam

agentes maliciosos que consigam explorar falhas tanto no software como no hardware dos mesmos.

De acordo com o *Georgia Tech's Institute for Information Security and Privacy* existe uma relação no aumento do cibercrime diretamente ao crescimento do uso de dispositivos IoT. O aumento no número de elementos IoT conectados à internet vem se multiplicando exponencialmente e em paralelo cresce também o interesse de atacantes em encontrar novas vulnerabilidades neste ecossistema rico em informações e poder tecnológico. Apesar da grande demanda deste tipo de dispositivo, a segurança ainda não tem sido motivo de preocupação por parte dos fabricantes, que além de não possuírem um processo focado em segurança, devido ao aumento no custo de implementação, utilizam desenvolvedores que não possuem o treinamento adequado para lidar com essa questão (INSTITUTE, 2016).

Conforme podemos ver na Figura 2 a evolução dos processadores nos levou a um barateamento dos componentes eletrônicos, fato este que alavancou e possibilitou a popularização dos dispositivos IoT. No entanto a intenção dos fabricantes em reduzir os custos de projeto e desenvolvimento ao máximo, sem avaliar os riscos geram consequências. E neste caso a consequência é: vulnerabilidade.

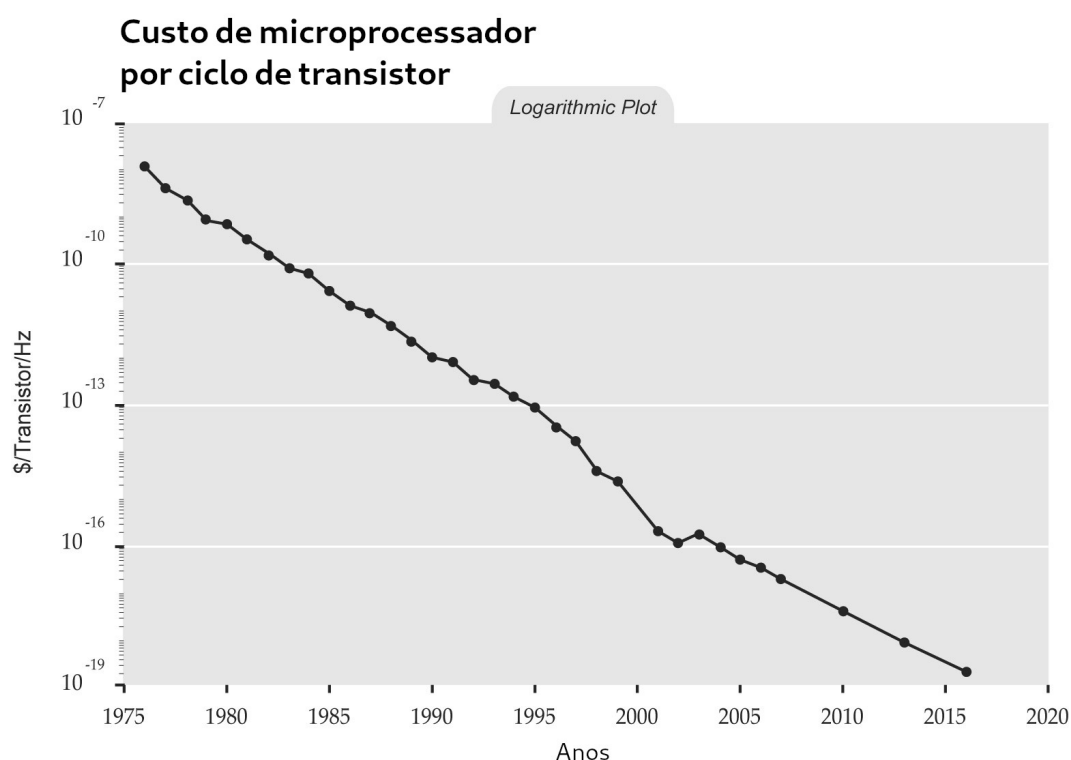


Figura 2: A queda do custo do processamento ao longo dos anos. Fonte: (SINGULARITY, 2017)

Um estudo realizado pela Hewlett-Packard (RICHARDS, 2016), descobriu-se que 7 de 10 dispositivos testados, incluindo uma Smart TV, termostato doméstico e uma tranca

de porta com conexão de rede, apresentaram vulnerabilidades graves e puderam ser exploradas. Outro estudo realizado pela Symantec em 2014 (BARCENA; WUEEST; LAU, 2014) descobriu que um scanner de US\$ 75 poderia capturar informações confidenciais ou sensíveis de monitores de exercícios e outros dispositivos portáteis. Outra característica relacionada ao contexto de IoT é a capacidade que dispositivos comprometidos tem em agregar força em ataques de negação de serviço e outros crimes cibernéticos. Ataques dessa natureza já se valeram de elementos IoT, como vimos em 2016 com o *malware Mirai* (NEWS, 2016) e seus variantes e mais recentemente com o *IoT_reaper* (NEWS, 2017). Esses *malwares* em conjunto com ferramentas automatizadas tem conseguido explorar milhares de dispositivos pela internet, comprovando como a IoT tem grande potencial em amplificar a robustez e sofisticação de ataques de negação de serviço.

Dos dispositivos móveis aos roteadores, os cibercriminosos estão sempre à procura de plataformas que sejam menos protegidas para que possam explorar. Além dos CPSs e IoT os dispositivos móveis, serviços móveis e usuários móveis tiveram um crescimento expressivo ao longo dos anos. Em 2015, a empresa RSA Data Security (SECURITY, 2016) verificou que 45% de todas as transações eram originárias de sistemas móveis, enquanto que 61% das tentativas de fraude eram feitas a partir de um dispositivo móvel. À medida que as organizações continuam a lançar mais serviços móveis aos clientes e os funcionários começam a depender deles para fazer negócios em seu nome, o canal de comunicação móvel tornou-se abundante em crimes cibernéticos. O celular é cada vez mais popular no mundo e se tornou o canal dominante para a comunicação instantânea e a via expressa para o setor bancário e comercial. Podemos ainda citar que a possibilidade da comunicação direta entre dispositivos de IoT com dispositivos móveis agrava ainda mais na quantidade de vetores de ataque. De fato, até 2020, estima-se que 80% dos adultos na Terra terão um *smartphone* (ECONOMIST, 2019). Os dispositivos móveis ajudaram a influenciar mais de US\$ 1 trilhão no total de compras em 2015 entre as transações on-line e off-line e as receitas das vendas de comércio eletrônico móvel deverão atingir US\$ 523 bilhões em 2020 (LINDNER, 2019). As organizações transformaram a maneira como interagem com os clientes e isso não passou despercebido pelos cibercriminosos. Isso fica evidente pelo aumento das tentativas de fraude provenientes do canal móvel, aumentando 173% entre 2013 e 2015 em comparação a apenas 1% no canal da Web (SECURITY, 2016). No entanto o Mobile não é a única plataforma em risco. Outras plataformas baseadas em PC foram segmentadas, incluindo roteadores que também estão sujeitos a *malware* e permitiram que atacantes obtivessem controle remoto quase indetectável para lançar ataques contra outros roteadores e sistemas na mesma rede.

A segurança, no entanto, não é um problema a ser enfrentado apenas pelo cidadão comum. Atualmente a segurança da informação vem sendo tratada como uma área indispensável no meio corporativo. Um estudo da Gartner (GARTNER, 2016), afirma que só de agosto até final de dezembro de 2016 os gastos mundiais com produtos e serviços de

segurança chegaram a mais 81,6 bilhões de dólares, o que representa um crescimento de 7,9% em relação a 2015.

Anualmente a quantidade de ataques cibernéticos tem crescido significativamente. Em uma publicação de 2015 no blog da ESET é citado que, nos Estados Unidos, vem ocorrendo um crescimento de 80 a 90 milhões de ataques anualmente, com aproximadamente 400 novas ameaças por minuto e cerca de 70% dos ataques não são identificados (ESET, 2015). De acordo com o Cybermap Kaspersky no presente momento o Brasil ocupa o 7º lugar no ranking de países que mais sofrem ciberataques (KASPERSKY, 2017). Outro grande problema ao lidar com ataques cibernéticos são as despesas e prejuízos que eles podem acarretar. Em um documento publicado pela HP em um estudo realizado por (PONEMON, 2015) com 252 empresas de sete países, incluindo o Brasil, é possível observar na Figura 3, os altos custos envolvendo crimes cibernéticos nos anos de 2013 a 2015 e seu contínuo crescimento baseado em um estudo mais recente também realizado pelo Instituto Ponemon (PONEMON, 2019), como vemos na Figura 4.

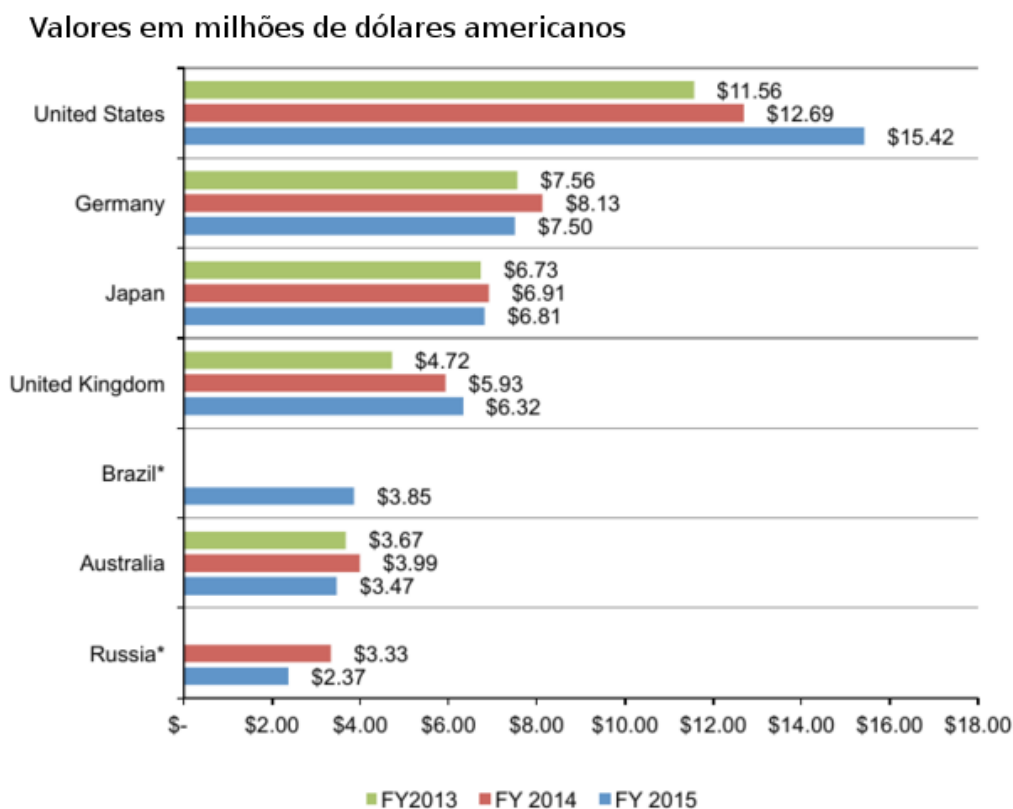


Figura 3: O custo médio anual do cibercrime por país. Fonte: (PONEMON, 2015)

1.2 O Futuro da Internet das Coisas e a Cibersegurança

De acordo com as pesquisas, o cenário futuro demonstra que o contínuo crescimento tanto de dispositivos IoT quanto não IoT. A Figura 5 traz uma previsão que demonstra

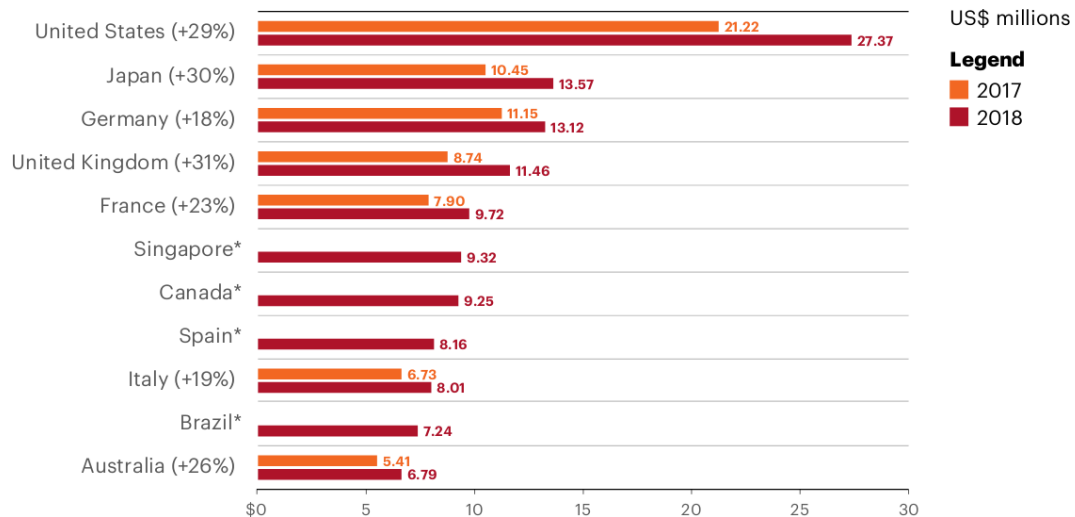


Figura 4: O custo médio anual do cibercrime por país . Fonte: (PONEMON, 2019)

que dispositivos IoT tendem a crescer muito mais em relação à outras tecnologias. Isso delineia um cenário onde num futuro próximo estaremos muito mais imersos em uma tecnologia ubíqua e pervasiva do que estamos hoje.

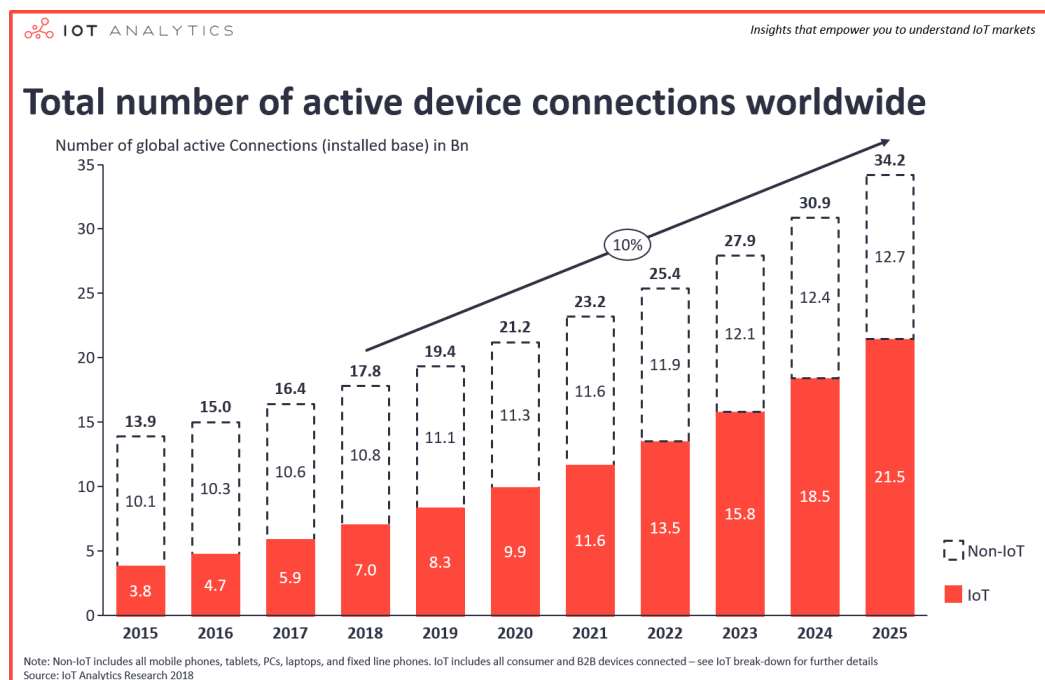


Figura 5: Previsão do crescimento de elementos IoT conectados no mundo. Fonte: (ANALYTICS, 2018)

Apesar da crescente atenção e do rápido desenvolvimento, a Internet das Coisas não atingiu seu ápice quando se trata de automação residencial (SYMANTEC, 2017). Talvez um dos obstáculos mais difíceis de serem ultrapassados pela IoT sejam os protocolos de comunicação padronizados. Até agora, vimos muito crescimento com dispositivos IoT interconectados usando protocolos bem estabelecidos, como Wi-Fi e Bluetooth. Os

dispositivos que utilizam protocolos sem fio 802.11b / g / n / ac, incluindo TVs inteligentes, termostatos inteligentes, câmeras IP e outros dispositivos, estão aparecendo em todos os lugares. Os dispositivos que empregam Bluetooth 4.0, como rastreadores de fitness, relógios inteligentes e outros *wearables*, também ajudaram a IoT a obter uma força significativa nesse mercado. No entanto, esses protocolos de comunicação não se comportam como esperado em muitos casos de automação residencial. Além disso não foi possível fazer isso de forma barata o suficiente para levar a tecnologia à onipresença. Mas novamente, ao introduzir qualquer nova tecnologia, a superfície de ataque se expande, o que apresenta uma variedade de novos problemas do ponto de vista da segurança. As redes proprietárias de IoT já foram encontradas com múltiplas vulnerabilidades de segurança, algumas triviais e algumas graves. A questão fundamental sobre IoT na automação residencial é: como aplicar a tecnologia em um lugar tão importante como o lar de maneira segura? Com a adoção de padrões comuns, é provável que os protocolos proprietários mais antigos falhem no caminho, preparando o caminho para uma consolidação potencialmente maior no mercado (SYMANTEC, 2017). Sabemos que a segurança cibernética deve estar no cerne para que a adoção desta nova geração de tecnologia tenha sucesso. A medida que mais casas se tornam conectadas, mais difícil será para os consumidores ignorarem os benefícios que essa nova tecnologia promete. É sempre importante pesar a conveniência de controle remoto, automação, facilidade de uso e os benefícios que eles podem trazer, contra os potenciais riscos introduzidos que poderiam levar a exploração de vulnerabilidade que comprometeriam a privacidade dos usuários.

Em decorrência desse crescimento de elementos conectados e junto ao crescimento dos usuários de internet, a Cybersecurity Ventures (VENTURES, 2017) prevê que os custos globais com o cibercrime anual crescerão de US\$ 3 trilhões em 2015 para US\$ 6 trilhões até 2021, o que inclui danos e destruição de dados, dinheiro roubado, perda de produtividade, roubo de propriedade intelectual, roubo de dados pessoais e financeiros, desfalque, fraude, destruição ou interferência no curso normal dos negócios, investigação forense, restauração e exclusão de dados e sistemas e perda de reputação.

Ainda acrescentando que estamos na era da informação, de acordo com um estudo realizado pela Juniper em 2017, é esperado que as brechas de dados não só se intensifiquem em termos de seu impacto, mas também se tornem mais prolíficas. Ao redor do mundo, em 2015 houveram mais de 6.000 vazamentos de dados. Espera-se que esse valor atinja 16 mil até 2020. Esse aumento deve-se, em parte, à crescente superfície de ataque causada pelo crescimento dos dados empresariais. De acordo com a IDC, a capacidade total de armazenamento em todo o mundo será de 521 mil petabytes em 2020 (RESEARCH, 2017).

1.3 Motivação

A área da segurança de informação focada em IoT vem ganhando foco recentemente devido os diversos ataques como:

- Ataque DDoS massivo originado de 100 mil dispositivos IoT hackeados (NEWS, 2016)
- Nova *botnet* IoT de crescimento rápido (NEWS, 2017)
- O novo *malware* VPN *Filter* tem como alvo pelo menos 500 mil dispositivos de rede em todo o mundo (CISCO, 2018)
- Variante do *malware* Mirai adiciona dezenas de novas possibilidades de exploração em dispositivos de IoT da empresariais (NEWS, 2019).

Grande parte desses ataques ainda são bem sucedidos utilizando ataques simples como tentativas de senha padrão ou ataques de força bruta, estando esses nas primeiras posições no ranking da Open Web Application Security Project (OWASP) de ameaças a IoT (OWASP, 2018). Reforçando essa afirmação podemos citar os casos apresentados na seção 1.1 envolvendo o *malware* Mirai e seus variantes que do ano de 2016 até a atualidade, vem infectando centenas de milhares de componentes IoT através de varreduras na internet em busca de elementos com senhas padrão ou suscetíveis a ataques de força bruta em serviços de acesso remoto como Telnet e SSH. A nível nacional, ainda podemos observar mais adiante nos resultados extraídos do projeto *honeyTARG Honeypots* (PROJECT, 2019) - seção 3.3 - onde quase a totalidade do tráfego gerado nos *honeypots* do projeto na amostra temporal selecionada foram direcionados as portas de serviços de acesso remoto (Telnet e SSH), resultado esse coeso ao obtido no experimento realizado no Capítulo 4, onde grande parte do tráfego capturado foi direcionado a esses serviços, caracterizando tentativas de acesso indevido. Um dos grande problemas relacionados a esse tipo de exploração envolvem o despreparo, desconhecimento ou desinteresse por parte do usuário em aumentar a segurança de suas redes e produtos, alterando senhas padrões e utilizando senhas fortes. Conforme o artigo *Practical Ways to Misuse a Router* publicado pela *Positive Technologies* (TECHNOLOGIES, 2017) 15% das pessoas que possuem dispositivos IoT não se preocupam em mudar a senha padrão. O artigo também revela que apenas cinco das combinações de nome de usuário e senha mais populares são suficientes para obter acesso administrativo a 1 em cada 10 dispositivos. Isso significa que milhões de dispositivos são extremamente vulneráveis a ataques dos mais diversos, como os que desejam criar *botnets*, podendo utilizar uma lista de senhas padrão para obter acesso a esses dispositivos e adicioná-los a uma *botnet* de IoT. No entanto é sabido que diversos componentes IoT já presentes no mercado e até mesmo online atualmente na internet possuem senhas “*hardcoded*” que são senhas que estão inseridas no código

fonte do equipamento, normalmente sem nenhuma criptografia e que independente da vontade do usuário não podem ser alteradas e ainda sendo fabricadas em série e tendo todos produtos compartilhando da mesma senha. Outro fator importante a se considerar são vulnerabilidades encontradas nos softwares/firmwares desses equipamentos. Diariamente diversas vulnerabilidades são reportadas ao sistema CVE (Common Vulnerabilities and Exposures), sistema responsável por catalogar vulnerabilidades nos mais diversos tipos de sistemas, incluindo elementos IoT. Essas vulnerabilidades constantemente independem da qualidade da senha ou se a senha foi alterada pelo usuário, pois elas exploram falhas em serviços ou protocolos que esses sistemas utilizam. Para corrigir um problema como esse é necessário que o fabricante disponibilize uma versão de correção do software/firmware para esses equipamentos, o que muitas vezes não ocorre ou as atualizações não ocorrem de forma automática e/ou prática, dificultando muito para usuários comuns aplicarem as correções, se elas existirem. Por último ainda devemos considerar vulnerabilidades desconhecidas, que são formas de explorar equipamentos e serviços com técnicas ainda não descobertas e documentadas ou vulnerabilidades desconhecidas por órgãos de controle como Mitre Corporation, responsável pelo sistema CVE, CERT Coordination Center e seus derivados de cada país, responsáveis por incidentes de segurança a nível global e nacional respectivamente ou empresas privadas como Kaspersky Lab e Symantec Corporation. Desta forma se torna coerente propor uma solução para mitigar esses problemas de forma a dificultar tentativas de ataque de força bruta e senhas padrões, disponibilizar serviços de acesso remoto atualizados independente de atualizações por parte dos desenvolvedores e analisar o comportamento dos elementos a fim de identificar uma exploração que ainda não tenha sido identificada, bloqueá-la e possivelmente aprender com ela a fim de melhorar a segurança.

1.4 Objetivos

Tendo em vista os riscos ao qual estamos expostos e considerando a natureza pervasiva e a ubiquidade desse tipo de sistema na vida do ser humano, uma questão muito relevante a se considerar é a segurança da informação. A possibilidade de explorar uma falha em um sistema que está diariamente presente na vida das pessoas oportuniza agentes maliciosos a comprometerem o sistema e a segurança do usuário de maneiras que antes não era possível. Desta forma é necessário analisar e buscar por possíveis falhas, quantificar a gravidade das mesmas, determinar os impactos no sistema e como elas expõem quem o usa para alertar e proteger o sistema e o usuário. Baseando-se nisso surgiu a proposta de um **sistema automatizado para mitigar problemas relacionados a segurança de elementos IoT** inerentes a esse tipo de tecnologia, visando reduzir a possibilidade de exploração de casos conhecidos e documentados por diversos órgãos de pesquisa, pesquisadores acadêmicos e empresas de segurança da informação e corroborado experimental-

mente neste trabalho. O sistema monitorará os elementos conectados a ele e identificará acessos indevidos e tentativas de ataque em tempo real. Além disso acessos identificados como maliciosos serão isolados e capturados para estudo posterior. O presente trabalho apresenta inicialmente um estudo utilizando um *honeypot* de baixa interatividade que foi executado para avaliar a aplicabilidade deste modelo de sistema. Os resultados obtidos na região de estudo foram tratados e comparados com dados disponibilizados no cert.br (CERT.BR, 2019) e se mostraram condizentes com os dados nacionais. Foi então proposto um sistema implementado em um computador de placa única, que funciona como um centralizador de conexão para elementos IoT, roteando a conexão dos elementos internamente com a internet. O sistema gerencia tentativas de acesso a elementos conectados a ele adicionando uma camada extra de comunicação nos protocolos de acesso remoto. O gerenciador de acessos ainda monitora as conexões em tempo real analisando o comportamento do usuário. Se for identificado que o comportamento é hostil, a conexão é redirecionada e isolada dos elementos reais em um ambiente virtualizado que funciona como um laboratório de estudos emulando um elemento IoT real, para captura de malwares e análise do comportamento do atacante ou ferramenta automatizada de ataque.

1.4.1 Objetivo Geral

Este trabalho tem como objetivo estudar problemas relacionados a segurança da informação com foco em IoT e desenvolver um sistema para mitigar a eficácia de ataques direcionados a comunicação de elementos desse tipo de sistema expostos na internet.

1.4.2 Objetivos Específicos

1. Estudar problemas relacionados a segurança da informação em elementos IoT.
2. Instalar e configurar um *honeypot* para obtenção de dados de ataques locais.
3. Desenvolver uma ferramenta para tratar os resultados obtidos pela utilização de um *honeypot*.
4. Obter um *dataset* da incidência de ataques a diferentes tipos de serviços na região.
5. Propor um modelo de sistema para aumentar a segurança, mitigando acessos indevidos e o comprometimento dos elementos conectados a ele.

1.5 Organização do trabalho

Este trabalho está organizado da seguinte forma: No Capítulo 2 são apresentados conceitos fundamentais para a compreensão do trabalho proposto distinguindo algumas técnicas e ferramentas, seus tipos, suas aplicações, vantagens e desvantagens. No Capítulo 3 são apresentados trabalhos relacionados ao tema principal deste trabalho. O Capítulo

4 apresenta a implementação do *honeypot* utilizado para obtenção de dados de ataques locais bem como os resultados obtidos. No Capítulo 5 é apresentada o *IoT Tunnel*, modelo proposto para mitigação de vulnerabilidades na comunicação de elementos IoT, proposta principal do trabalho. E finalmente o Capítulo 6 apresenta as conclusões e trabalhos futuros deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo serão apresentados alguns conceitos que julgou-se importante para a melhor compreensão da proposta do trabalho.

2.1 Taxonomia da IoT

Desde o surgimento da IoT diversas definições de padronização foram propostas no decorrer dos anos. A versão mais simplificada divide a IoT em três camadas bem definidas cada uma responsável por um tipo de característica desses elementos. Esse modelo se divide em camada de percepção, de rede e de aplicação. Ele separa de uma forma mais simplificada e direta as camadas dando a ideia básica das funcionalidades e objetivos da IoT (YUN; YUXIN, 2010). Mais recentemente se mostrou necessário um modelo mais completo que especificasse melhor os detalhes do funcionamento de uma rede IoT e melhorasse o entendimento e até abordagens para segurança. Para isso foi proposto o modelo de cinco camadas adicionando a camada *middleware* ou processamento e a camada *business* (KHAN et al., 2012). Nessa distribuição cada uma das camadas ficou responsável pelas seguintes funções:

1. **Camada de percepção:** Esta camada tem a função de identificar as coisas a seu redor e coletar as informações delas. Existem muitos tipos de sensores conectados aos objetos para coletar informações como RFID, código de barras 2D e sensores de temperatura. Os sensores são escolhidos de acordo com os requisitos das aplicações. As informações coletadas por esses sensores podem ser sobre localização, alterações no ar, ambiente, movimento, vibração etc.
2. **Camada de rede ou transporte:** A camada de rede, também conhecida como camada de transporte, atua como uma ponte entre a camada de percepção e a camada de aplicação e de processamento. Ela transmite as informações coletadas dos objetos físicos através de sensores utilizando uma rede que pode ser sem fio ou com fio. Também assume a responsabilidade de conectar elementos IoT, dispositivos e outras redes entre si.

3. **Camada middleware ou de processamento:** A camada de middleware, também conhecida como camada de processamento, é responsável por coletar as informações enviadas da camada de transporte. Ela executa o processamento das informações coletadas a fim de extrair informações úteis. Essas informações são enviadas, através da camada de rede, para servidores externos (nuvem) onde o poder de processamento e armazenamento é muito superior.
4. **Camada de aplicação:** A camada de aplicação é responsável por fornecer serviços específicos ao usuário. Ele define vários aplicativos nos quais a IoT pode ser implantada. Os serviços podem variar para cada aplicativo, pois dependem das informações coletadas pelos sensores e do objetivo da aplicação em si.
5. **Camada business:** A camada *business* se responsabiliza pelo comportamento desejado em um aplicativo e age como um gerente do sistema inteiro. Ele tem responsabilidade em gerenciar e controlar aplicativos, criando modelos e gráficos de negócios com base nos dados recebidos da camada de aplicação. A privacidade do usuário também é gerenciada por essa camada. Ela também tem a capacidade de determinar como as informações podem ser criadas, armazenadas e alteradas.

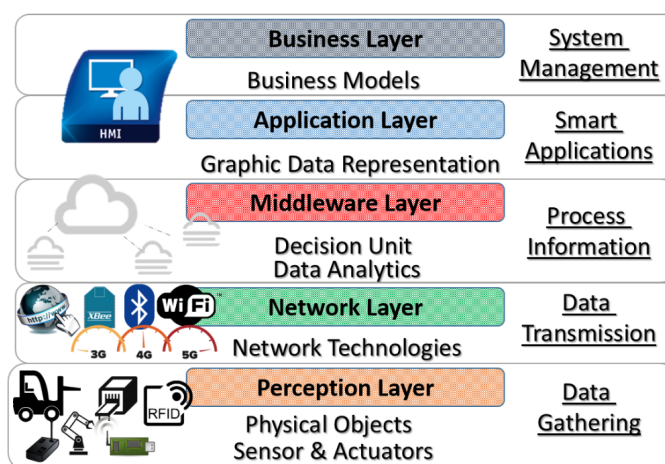


Figura 6: Camadas de um elemento IoT. Fonte: (CHEN et al., 2018)

2.1.1 Vulnerabilidades da IoT

A IoT é extremamente vulnerável a ataques por várias razões. Inicialmente seus componentes passam grande parte do tempo atuando de forma autônoma e sem assistência facilitando o ataque físico. Também, a maioria das comunicações desses elementos são sem fio, o que favorece a interceptação das mesmas. Finalmente, a maioria dos elementos IoT são caracterizados por possuir baixa capacidade de recursos de energia e computação (principalmente elementos passivos) e, portanto, não podem implementar esquemas complexos de segurança (ATZORI; IERA; MORABITO, 2010).

Na Figura 7 é apresentada uma visão geral categorizada das vulnerabilidades presentes em sistemas IoT. O gráfico apresentado é uma adaptação retirada da pesquisa Demystifying IoT Security (NESHENKO et al., 2019) onde os autores categorizaram as vulnerabilidades presentes na IoT baseando-se em uma vasta quantidade de trabalhos relacionados.

Como vemos, os autores classificaram as ameaças seguindo o modelo de três camadas sendo elas vulnerabilidades baseadas em dispositivo, em rede e em software. O impacto dessas vulnerabilidades foram categorizadas com base nos tipos de ameaças que eles representam a confidencialidade, integridade, disponibilidade e responsabilidade. Com relação aos tipos de ataques são identificadas falhas de segurança categorizadas pela abordagem na qual as vulnerabilidades de IoT inferidas podem ser exploradas. Esta classe é dividida em três subclasses, que abordam ataques contra confidencialidade e autenticação, ataques contra a integridade de dados e ataques contra a disponibilidade. Os autores exploram ainda abordagens de defesa que incluem contramedidas - técnicas de remediação disponíveis para mitigar as vulnerabilidades da IoT - contendo: Controle de acesso e autenticação, garantia de software e protocolos seguros. Também incluem consciência situacional, responsável por categorizar as técnicas disponíveis para capturar informações precisas e suficientes sobre atividades maliciosas geradas no contexto da IoT. São elas: Análise de vulnerabilidades, *honeypots*, descoberta de rede e detecção de intrusão.

- **Camadas:** De um modo geral as vulnerabilidades na IoT são distribuídas em três camadas de sua estrutura. A seguir vemos como se caracterizam essas vulnerabilidades nas diferentes camadas.
1. **Baseadas em dispositivo:** Um grande número de dispositivos IoT opera de maneira autônoma e um atacante pode se valer de um acesso físico a um dispositivo para causar danos significativos, alterar seu funcionamento ou obter acesso ilimitado aos dados armazenados em sua memória fornecendo assim controle total do dispositivo.
 2. **Baseado em rede:** A necessidade de possuir comunicação em rede constante e principalmente via *Wi-Fi* nesse tipo de tecnologia expõe os elementos IoT a vulnerabilidades presentes em sistemas computacionais similares. Essa característica possibilita que atacantes explorem essa comunicação valendo-se de falhas comumente encontradas como; autenticação inadequada, criptografia imprópria e portas abertas desnecessariamente.
 3. **Baseado em software:** Os atacantes também podem obter acesso remoto a sistemas IoT explorando vulnerabilidades de software. Falhas nessa camada envolvem controle de acesso insuficiente, incapacidade de atualizações de segurança, más práticas de programação, ausência de mecanismo de auditoria.

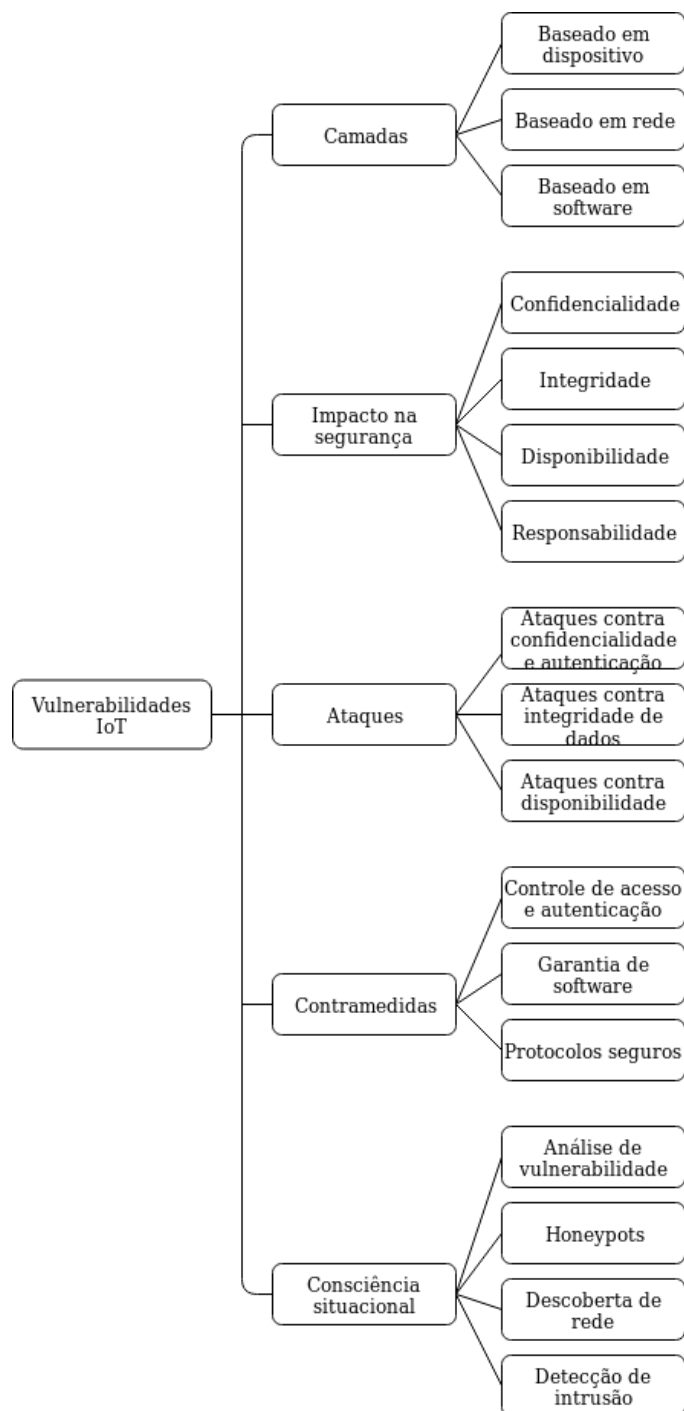


Figura 7: Categorização das vulnerabilidades da IoT.

- **Impacto na segurança:** Ataques que exploram vulnerabilidades em um sistema causam impactos negativos em seu funcionamento. Esses impactos agem diretamente em pilares importantes na segurança da informação como descritos abaixo.
1. **Confidencialidade:** A confidencialidade tem como objetivo proteger os ativos de acesso não autorizado garantindo que somente quem deve estar participando de um troca de mensagens específica realmente é quem ali está. Para isso normalmente é aplicado um controle de acesso rigoroso, procedimentos de autenticação e criptografia adequada.
 2. **Integridade:** Garantir a integridade significa que não vão haver modificações não autorizadas durante a comunicação de elementos. Desta forma deve-se possuir controle de acesso, criptografia, restrições de interface, validações de entrada e métodos de detecção de invasão.
 3. **Disponibilidade:** A disponibilidade garante que um elemento vai estar acessível quando necessário. Para isso é necessário monitorar e adaptar os sistemas implementando mecanismos de redundância, mantendo sistemas de *backup* e aplicando políticas de segurança e gerando correções e atualizações de *software* (ou *firmware*).
 4. **Responsabilidade:** O objetivo da responsabilidade é garantir a viabilidade de rastrear ações e eventos para determinado usuário ou sistema atacante, com o intuito de responsabilizá-lo pelas ações. Para isso se faz necessário a utilização de mecanismos de auditoria e capacidade de armazenamento de *logs* e metadados que sirvam como evidências e permitam recriar os passos realizados por um atacante ao sistema.
- **Ataques:** Diversos tipos de ataques são possíveis em sistemas computacionais. De modo geral diferentes tipos de ataques podem impactar em diferentes áreas do sistema como descrito a seguir.
1. **Ataques contra confidencialidade e autenticação:** O principal objetivo desse tipo de ataque é obter acesso não autorizado a recursos e dados de elementos IoT para então realizar outras ações maliciosas. Esse tipo de ataque normalmente é executado através de ataques de força bruta, extração de dados valendo-se de acesso físico a elementos IoT ou pela falsificação de identidade em mecanismos de autenticação de dispositivos.
 2. **Ataques contra integridade de dados:** Ataques contra integridade são caracterizados pela injeção de dados falsos ou modificação do *firmware* do dispositivo. Ataques dessa natureza envolvem a alteração ou inserção de pacotes durante a comunicação de elementos em uma rede ou a engenharia reversa de binários do

firmware para então modificá-los e instalá-los nos dispositivos alterando seu comportamento original.

3. **Ataques contra a disponibilidade:** Ataques contra a disponibilidade são comumente conhecidos como ataques de negação de serviço ou *denial-of-service* DOS. Esse tipo de ataque tem como objetivo tornar os serviços dos elementos IoT inacessíveis impedindo o acesso de usuários legítimos ao sistema. Sua estratégia envolve o esgotamento dos recursos computacionais ou de rede dos elementos através de modificação de topologia da rede ou exploração de alguma vulnerabilidade conhecida.
- **Contramedidas:** As contramedidas são formas de um sistema se proteger de ataques ou mitigar o impacto dos mesmos. Apesar das dificuldades de implementação de sistemas de contramedidas em elementos IoT devido as limitações de hardware, sua utilização se mostra cada vez mais indispensável.
1. **Controle de acesso e autenticação:** O controle de acesso busca limitar o acesso a elementos para usuários ou sistemas que tenham obtido a permissão para tal, através do uso de autenticação e autorização. Usuários que não estejam autenticados e/ou autorizados a efetuar o acesso ao sistema devem ser impedidos por um sistema de controle. O controle de acesso é importante para garantir que um elemento IoT não seja explorado por um ente malicioso que encontre o elemento em uma rede limitando esse acesso.
 2. **Garantia de software:** A garantia de software visa reduzir as vulnerabilidades do código fonte e de binários do sistema a fim de garantir robustez no funcionamento dos elementos. Considerando o potencial impacto da exploração direta do software de elementos IoT, a garantia de software adequada deve ser parte integrante do processo de desenvolvimento efetuando testes no código fonte e aplicando técnicas de verificação para assim minimizar a efetividade de ataques.
 3. **Protocolos seguros:** O uso de protocolos seguros garantem uma nível maior de segurança na comunicação de elementos IoT. Protocolos que utilizam criptografia para transferência de informação, por exemplo, dificultam ou impossibilitam ataques que comprometam a confidencialidade ou integridade das mensagens. No entanto a baixa capacidade computacional de elementos da IoT tornam a utilização de algoritmos criptográficos custosa devido a complexidade matemática na execução de criptografar e descriptografar pacotes em tempo real durante a comunicação.
- **Consciência situacional:** Para garantir um certo nível de segurança e resiliência da IoT, a eficácia de qualquer mecanismo de segurança necessita de uma capacidade de inteligência acerca de novas técnicas e novos tipos de ameaças.

1. **Análise de vulnerabilidades:** A análise de vulnerabilidades é um estudo abrangente de todo um sistema de informação, buscando potenciais pontos fracos de segurança. A execução de avaliações de segurança ajuda a descobrir vulnerabilidades da IoT antes que elas sejam exploradas. Uma abordagem muito comum nessa área foca no desenvolvimento de novas plataformas de testes ou na adoção de métodos existentes para realizar a avaliação de elementos da IoT a fim de encontrar possíveis vulnerabilidades que estejam presentes.
2. **Honeypots:** Os *honeypots* interceptam e analisam um ataque expondo intencionalmente vulnerabilidades em um sistema. Esse tipo de abordagem captura atividades maliciosas para investigação de vetores de ataque ou para gerar padrões de ataque, que podem ser usados para implementações de segurança. *Honeypots* imitam um tipo específico de dispositivo em um ambiente específico, introduzindo um grande problema de escalabilidade no contexto do ecossistema de IoT.
3. **Descoberta de rede:** A descoberta de rede é a capacidade que um sistema possui de encontrar elementos conectados a rede. Considerando a escalabilidade de elementos IoT em redes de computador é essencial ter um sistema com capacidade escalável para identificar dispositivos vulneráveis ou comprometidos para uma tomada de decisão imediata.
4. **Deteção de intrusão:** Sistemas de detecção de intrusão são uma abordagem eficaz na neutralização de ações maliciosas. A detecção de intrusão tem como objetivo identificar uma atividade maliciosa e então fazer uma tomada de decisão. Dados os recursos limitados dos dispositivos da IoT, a maioria das técnicas de detecção de intrusão implantadas são baseadas em rede com um sistema de resposta ativo, que opera interrompendo as comunicações dos nós comprometidos.

2.2 Atributos de um sistema seguro e suas ameaças

Em segurança é esperado que um sistema seja tolerante a falhas. De acordo com Tanenbaum (TANENBAUM; VAN STEEN, 2007) um sistema tolerante a falhas deve ser confiável, ou seja, podemos ter confiança de que o sistema estará prestando o serviço que se espera. Para isso este sistema deve possuir quatro características; disponibilidade, confiabilidade, segurança (*safety*) e capacidade de manutenção. Já os atributos básicos desejados em um sistema que visa segurança deve garantir o que é chamado na literatura de *CIA-Triad*, confidencialidade, integridade e disponibilidade (SALTZER; SCHROEDER, 1975). Todavia, com o passar do tempo e com a evolução na área de segurança, ficou comprovado que estes atributos não cobriam todas as ameaças as quais o sistema podia estar exposto (WHITMAN; MATTORD, 2011). Assim foi proposto o *IAS-octave* como

uma extensão ao *CIA-triad* adicionando: Responsabilidade, auditabilidade, autenticidade, não-repúdio e privacidade (CHERDANTSEVA; HILTON, 2013).

Apesar da divergência de alguns autores em quais são os atributos básicos para garantir segurança da informação, todos eles, aqui citados, convergem em três atributos sendo eles: confidencialidade, integridade e disponibilidade. Na norma ISO 27001 (ABNT, 2013) são apresentados as definições dos objetivos destes atributos:

1. **Integridade:** Preservar exatidão e completude, ou seja assegurar que a informação está íntegra e que não seja alterada (em trânsito) para que o sistema funcione corretamente.
2. **Disponibilidade:** Garantir que o sistema estará acessível e utilizável sob a demanda de uma entidade autorizada, ou seja assegurar que o sistema esteja funcionando quando deveria estar funcionando.
3. **Confidencialidade:** Garantir que a informação não é disponibilizada ou divulgada a indivíduos, entidades ou processos não autorizados.

Uma outra maneira de ver a segurança em sistemas de computação é na situação em que tentamos proteger os serviços e dados contra ameaças a segurança. Pode-se citar quatro tipos de ameaças a segurança:

1. **Interceptação:** O conceito de interceptação refere-se à situação em que se obtém acesso a um serviço ou dados de forma não autorizada. Um usuário que analisa uma rede e visualiza e/ou captura uma troca de mensagens privada entre um cliente e servidor sem autorização de ambas as partes está praticando uma interceptação.
2. **Interrupção:** A interrupção se refere ao fato de um ente malicioso intencionalmente impedir que um serviço ou dados fiquem disponíveis para uso. Um exemplo muito comum é um ataque de negação de serviço ou *denial-of-service* DOS onde um ou múltiplos usuários atacam um serviço ou servidor até o mesmo não conseguir disponibilizar o serviço em questão.
3. **Modificação:** O ato de modificar implica em alteração de dados com o objetivo de mudar o comportamento do sistema ou ludibriar um usuário. A modificação - no contexto de redes de computadores - é utilizada em conjunto com a interceptação onde os dados que são capturados são também modificados para então ser encaminhados a um destino a fim de realizar um ato malicioso.
4. **Invenção:** Consiste na criação de informação que não existia previamente. Quando dados são inseridos em um banco ou pacotes são gerados e transmitidos fingindo ser outra máquina, serviço ou usuário.

Ainda de acordo com TANENBAUM; VAN STEEN 2007, afirmar que um sistema deve ser capaz de se proteger contra todas as possíveis ameaças de segurança não é a forma correta de se construir um sistema seguro. Para isso deve-se inicialmente obter uma descrição de uma política de segurança para então focar em mecanismos que possam garantir essa segurança. Para uma política de segurança devemos definir o que um ente tem permissão ou não para fazer em um sistema ou rede e então com uma política estabelecida pode-se utilizar mecanismos para impor essas políticas como:

1. **Criptografia:** A criptografia é um método de proteger informações e comunicações através do uso de códigos, de modo que apenas aqueles para os quais a informação é destinada possam lê-los e processá-los.
2. **Autenticação:** Autenticação é o processo de determinar se alguém ou algo é, de fato, quem ou o que se declara ser. A tecnologia de autenticação fornece controle de acesso para sistemas, verificando se as credenciais de um usuário correspondem às credenciais em um banco de dados de usuários autorizados ou em um servidor de autenticação de dados.
3. **Autorização:** Autorização é o processo de dar permissão a alguém para fazer ou ter algo. Em sistemas de computadores multiusuários, um administrador de sistema define para o sistema quais usuários têm permissão para acessar o sistema e quais privilégios de uso.
4. **Auditoria:** Ferramentas de auditoria são usadas para identificar acessos de clientes. A auditoria não é capaz de fornecer proteção contra ameaças de segurança no entanto as informações geradas podem ser de grande valia para uma análise de violação de segurança e para então posteriormente uma tomada de decisão contra intrusões e vulnerabilidades.

2.3 Virtualização

Em computação a virtualização representa a abstração de recursos computacionais da camada de hardware físico (SHACKLEFORD, 2012). Através da virtualização é possível a criação de uma versão virtual de componentes de computação física e ambos, componentes de hardware e software, podem ser virtualizados. A virtualização fornece um uso cada vez mais eficiente de recursos existentes que proporcionam enormes economias desses recursos em um curto período de tempo (PORTNOY, 2012). Existem diversos tipos de virtualização como emulação ou virtualização completa, paravirtualização e virtualização em nível de sistema operacional (KOLYSHKIN, 2006), entre outros. Cada um dos tipos de virtualização apresentam vantagens e desvantagens entre si, no entanto todos coincidem na característica de isolamento de ambiente. Com a virtualização é possível criar um

ambiente ideal para testar aplicativos. Múltiplas configurações do sistema operacional podem ser facilmente armazenadas e, em seguida, utilizadas para criar, verificar e validar aplicativos de software sendo também facilmente excluídas quando o teste é concluído (BARRETT; KIPPER, 2010). Graças ao seu isolamento do sistema hospedeiro, um ambiente virtualizado também pode ser usado para testar ações de *malware* de forma segura. Um *malware* executado em uma máquina virtual não pode prejudicar o sistema operacional hospedeiro. E se o *malware* danificar a máquina virtual, é possível simplesmente reinstalar o sistema operacional na máquina virtual ou retornar a máquina virtual para um estado onde não havia sofrido danos. (SIKORSKI; HONIG, 2012).

2.3.1 Máquinas Virtuais

As máquinas virtuais são como um computador dentro de um computador, conforme ilustrado na Figura 8. Um sistema operacional convidado é instalado no sistema operacional hospedeiro em uma máquina virtual, e o sistema operacional em execução na máquina virtual é mantido isolado do sistema operacional hospedeiro. Cada máquina virtual pode executar um sistema operacional em um ambiente independente e apresentar ao usuário, por meio de abstração, um substituto para uma máquina física. O *hypervisor* emula o hardware de CPUs, memória, armazenamento e recursos de rede, que podem ser compartilhados inúmeras vezes por várias instâncias das máquinas virtuais.

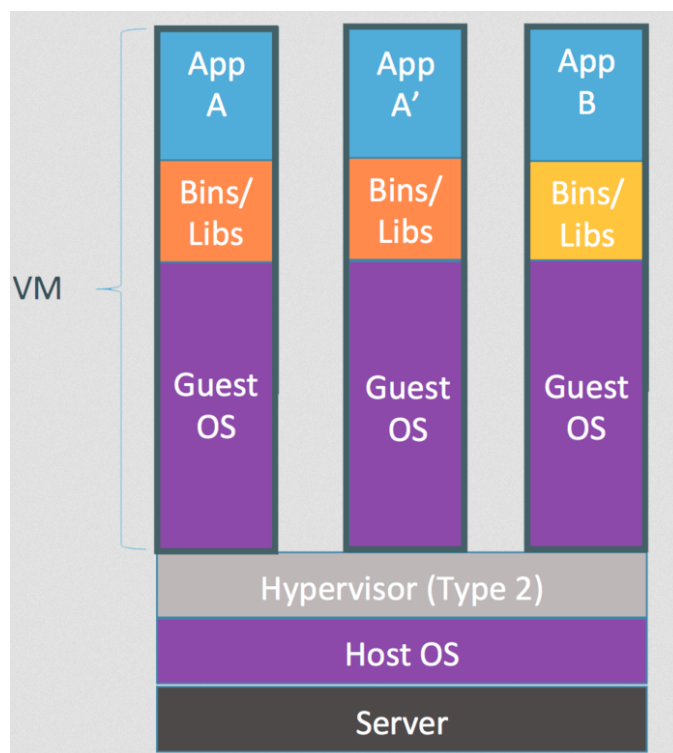


Figura 8: Representação de máquinas virtuais rodando sobre um sistema hospedeiro. Fonte: (DOCKER, 2019)

Sua principal vantagem, do ponto de vista da segurança, é que possuem um nível de isolamento maior pois cada máquina virtual executa um sistema operacional completo e independente, dificultando por exemplo que um *malware* comprometa múltiplas máquinas ou o sistema hospedeiro. No entanto, a execução de sistemas operacionais completos podem exigir uma sobrecarga substancial de recursos como memória, disco, entrada/saída e de rede, tornando-as custosas computacionalmente.

2.3.2 Containers

Containers são um tipo de virtualização em nível de sistema operacional que encapsulam os arquivos, as dependências e as bibliotecas de um aplicativo para serem executados em um sistema operacional. Aplicações de container permitem que o usuário crie e execute um contêiner separado para vários aplicativos independentes ou que vários serviços constituam um único aplicativo. Eles contêm os componentes necessários para executar o software desejado e esses componentes incluem arquivos, variáveis de ambiente, dependências e bibliotecas. O sistema operacional hospedeiro restringe o acesso do contêiner a recursos físicos, como CPU, armazenamento e memória garantindo dessa forma também o isolamento de processos executados no container.

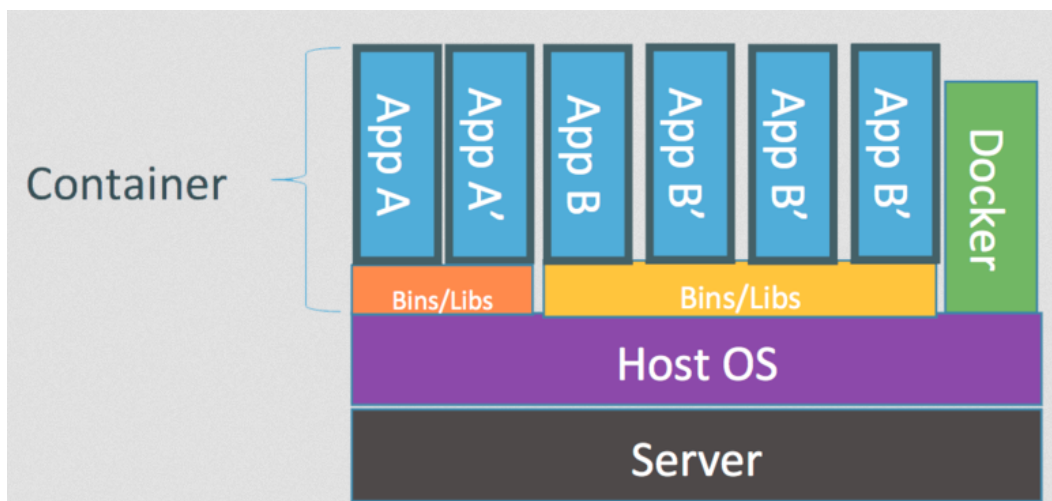


Figura 9: Representação de containers rodando sobre um sistema hospedeiro. Fonte: (DOCKER, 2019)

Conforme vemos na Figura 9 os contêineres executam diretamente sobre o sistema operacional hospedeiro compartilhando o *kernel* do mesmo. Desta forma apenas uma instância de um sistema operacional pode executar muitos contêineres isolados com um custo computacional muito menor em relação a máquinas virtuais oferecendo uma maior eficiência em memória, CPU e armazenamento como principais benefícios em comparação com a virtualização tradicional pois compartilhar o *kernel* permite que os contêineres sejam executados mais rapidamente possibilitando o uso de recursos de gerenciamento difíceis de realizar com a virtualização tradicional. Como os containers não

possuem a sobrecarga exigida pelas máquinas virtuais - como instâncias separadas de um sistema operacional - é possível executar muitos outros contêineres na mesma infraestrutura. Por outro lado o isolamento fornecido pelos containers é menos robusto do que o das máquinas virtuais reais, que têm *kernels* independentes e são executados sobre um *hypervisor*.

2.3.3 Sandboxing

Em segurança da computação, um *sandbox* é um sistema que serve para separar os programas em execução com o intuito de reduzir as falhas do sistema ou as vulnerabilidades do software. É frequentemente usado para executar programas ou códigos não testados ou não confiáveis sem o risco de prejudicar a máquina hospedeira ou o sistema operacional. O *Sandboxing* se vale de virtualização para capturar o tráfego quando ele entra ou sai da rede e é usado um ambiente para detecção de atividade mal-intencionada. Um *sandbox* normalmente fornece um conjunto de recursos rigidamente controlado para execução dos programas a serem executados, como espaço temporário no disco e na memória. O acesso à rede, a capacidade de inspecionar o sistema hospedeiro ou a capacidade de ler os dispositivos de entrada geralmente são proibidos ou altamente restritos. No entanto os *sandboxes* compreendem ambientes virtualizados que muitas vezes simulam os serviços de rede de alguma forma para garantir que o software ou *malware* que está sendo testado funcione normalmente para um estudo mais detalhado e sua compreensão (SIKORSKI; HONIG, 2012).

2.4 Controle de Acesso

Controlar o acesso a um objeto tem a ver com a proteção do objeto contra invocações por sujeitos que não podem ter métodos específicos (ou mesmo nenhum) executados (TANENBAUM; VAN STEEN, 2007). No contexto de redes de computador o controle de acesso é responsável por regular a troca de informações (mensagens/pacotes) entre elementos pertencentes a uma rede local e uma rede externa, podendo atuar tanto nas trocas internas quanto externas e também em ambas simultaneamente. Esse tipo de controle pode ser passivo ou ativo - dependendo da abordagem utilizada - podendo ele ser tanto responsável por uma tomada de decisão "*on the fly*", baseado em regras ou simplesmente uma captura de dados para análise posterior. A seguir são apresentados três abordagens para controle de acesso em redes de computador.

2.4.1 Firewall

Firewall é um dispositivo ou aplicativo de segurança que filtra o tráfego de entrada e saída da rede com base em regras predeterminadas. Ele atua analisando os cabeçalhos de pacotes e impõe a política com base no tipo de protocolo, endereço de origem, endereço

de destino, porta de origem e/ou porta de destino. Os pacotes que não correspondem à essas políticas são rejeitados e pacotes em conformidade são liberados para trafegar normalmente. Como vemos na Figura 10 (nesta topologia) ele é um intermediário entre a rede externa (internet) e a rede interna LAN filtrando de forma ativa as trocas de mensagens entre ambas redes. Existem também *firewalls* que atuam diretamente em um dispositivo em forma de aplicativo agindo similarmente, regulando as conexões que vem do meio externo (LAN ou Internet) com a rede interna (aplicações, redes virtuais).



Figura 10: Representação de um *firewall*. Fonte: (IPWITHEASE, 2019)

2.4.2 IDS

Sistemas de detecção de intrusão ou *Intrusion Detection Systems* (IDS) são sistemas que atuam analisando o tráfego de rede em busca de um comportamento suspeito na rede. Como vemos na Figura 11 um IDS se posiciona como um observador do tráfego de pacotes da rede atuando de forma passiva sem interferir nas trocas de mensagens. Seu objetivo é monitorar e identificar irregularidades nas trocas de mensagens e pacotes ou acessos a serviços e portas, analisando o fluxo de rede para então gerar relatórios e alertas ao sistema ou administradores da rede.

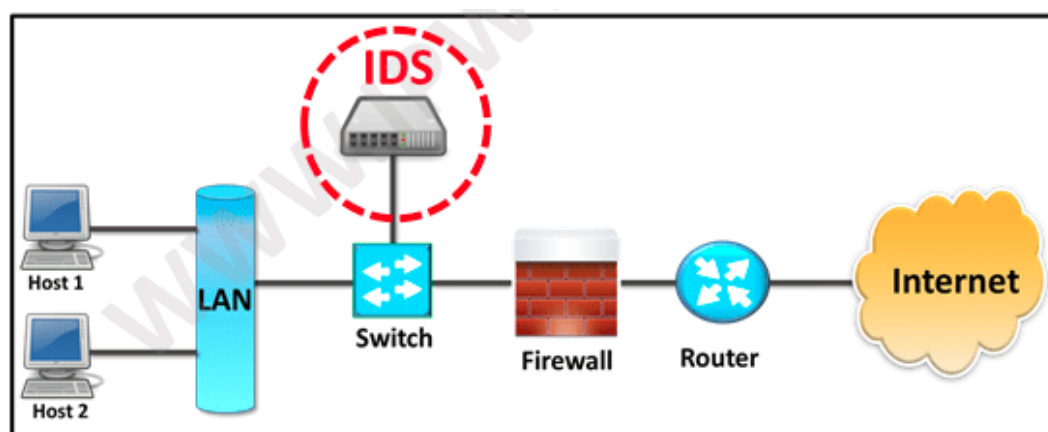


Figura 11: Representação de um IDS. Fonte: (IPWITHEASE, 2019)

2.4.3 IPS

Os sistemas de prevenção de intrusão ou Intrusion Prevention Systems (IPS), como vemos na Figura 12, atuam entre firewall e a rede interna. São utilizados para impedir que tráfego suspeito chegue ao resto da rede. O IPS monitora os pacotes de entrada e para o que eles realmente estão sendo usados antes de decidir deixar os pacotes entrarem na rede. Um IPS inspeciona de forma ativa o conteúdo da solicitação e toma decisão se deverá descartar, alertar ou até mesmo bloquear uma solicitação de rede mal-intencionada com base nesse conteúdo. A determinação do que é malicioso baseia-se na análise do comportamento ou no uso de assinaturas ¹.

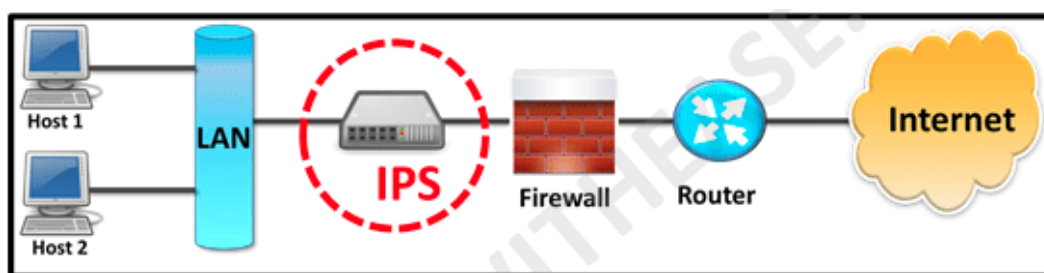


Figura 12: Representação de um IPS. Fonte: (IPWITHEASE, 2019)

2.5 Protocolos de Acesso Remoto

A necessidade de se acessar outros computadores remotamente sempre existiu. Para isso protocolos de comunicação foram criados possibilitando o envio e recebimento de comandos em interfaces de linha de comandos. Esses protocolos funcionam em um modelo cliente-servidor sobre TCP onde em um computador que se deseja obter acesso deixa-se serviço ativo que verifica em uma porta específica, requisições de acesso. Outros computadores que utilizam o serviço cliente podem estabelecer uma conexão direta para executar comandos no interpretador de linha de comandos ou *shell* do computador que executa o serviço servidor. Uma vez que a conexão foi estabelecida os computadores cliente adquirem a capacidade de executar comandos (limitados ao privilégio da conta de acesso) no computador servidor como se estivessem executando-os localmente. O primeiro protocolo padronizado e muito conhecido e utilizado até hoje é o Telnet e será abordado a seguir.

2.5.1 Telnet

O Telnet foi criado em 1969 antes mesmo da existência da internet. Além de antigo o Telnet é um protocolo muito simples que possui uma grande compatibilidade com muitos tipos de sistemas e que exige baixo consumo de banda. Graças a esses fatores o Telnet

¹ Assinaturas: Características catalogadas presentes em um dado capaz de identificá-lo

ainda é muito utilizado em sistemas embarcados e elementos IoT, pois sistemas desse tipo necessitam possuir acesso remoto e normalmente não possuem grande capacidade de processamento e energia. No entanto a simplicidade deste protocolo implica em problemas relacionados a segurança. Conforme podemos ver na Figura 13 um usuário através de um serviço cliente de Telnet pode fazer acesso a uma máquina que esteja rodando um serviço servidor de Telnet. O usuário faz uma requisição de acesso e o servidor envia uma resposta requisitando login e senha, o usuário então responde enviando os dados de login e senha para o servidor para então obter acesso ao shell. No entanto as informações trocadas entre o cliente e o servidor não possuem nenhum tipo de criptografia o que permite a um usuário ou programa, mal-intencionado, visualizar e capturar esses dados para possivelmente utilizá-los de forma indevida. O grande problema aqui é que não existe nenhum tipo de configuração que possa ser alterada no protocolo Telnet que evite esse tipo de ataque, visto que a forma como foi projetado não permite o uso de nenhum tipo de proteção para os dados que transitam.

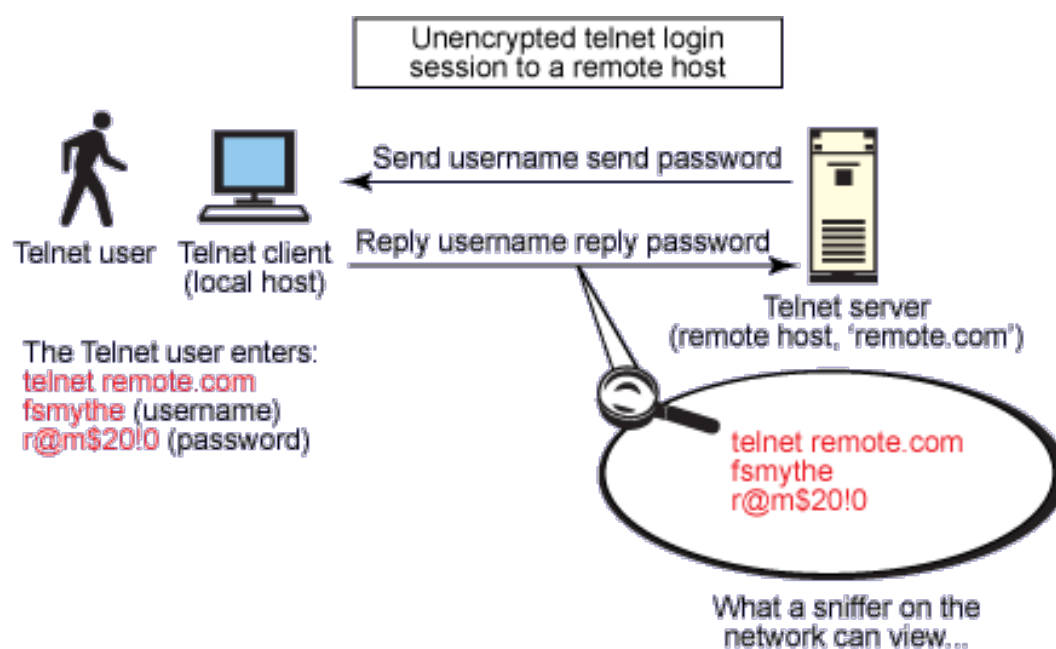


Figura 13: Demonstração de uma sessão Telnet. Fonte: (BURANDE et al., 2014)

2.5.2 SSH

O *Secure Shell* (SSH) é um protocolo de acesso remoto que veio para implementar uma comunicação segura e substituir o *Telnet* e outras implementações menos populares mas também inseguras como Berkeley *rlogin*, RSH e *rexec*. O grande diferencial do SSH é sua capacidade de garantir que a comunicação seja visível apenas para os participantes da comunicação e também que essa comunicação seja íntegra e inalterável por agentes externos. Para isso o protocolo faz uso de uma comunicação criptografada utilizando algoritmos conhecidos e seguros em um modelo chave privada e pública.

Assim como o *Telnet*, o SSH possui capacidade de executar comandos de forma remota e baseia-se no modelo cliente-servidor sobre TCP o que implica em um servidor ouvindo por novas conexões de clientes externos. Por utilizar criptografia o SSH exige um maior consumo de banda e processamento pois toda comunicação é embaralhada antes do envio e desembaralhada ao receber. Esse processo exige que dados adicionais sejam acrescentados nas mensagens e algoritmos de criptografia são executados constantemente na comunicação.

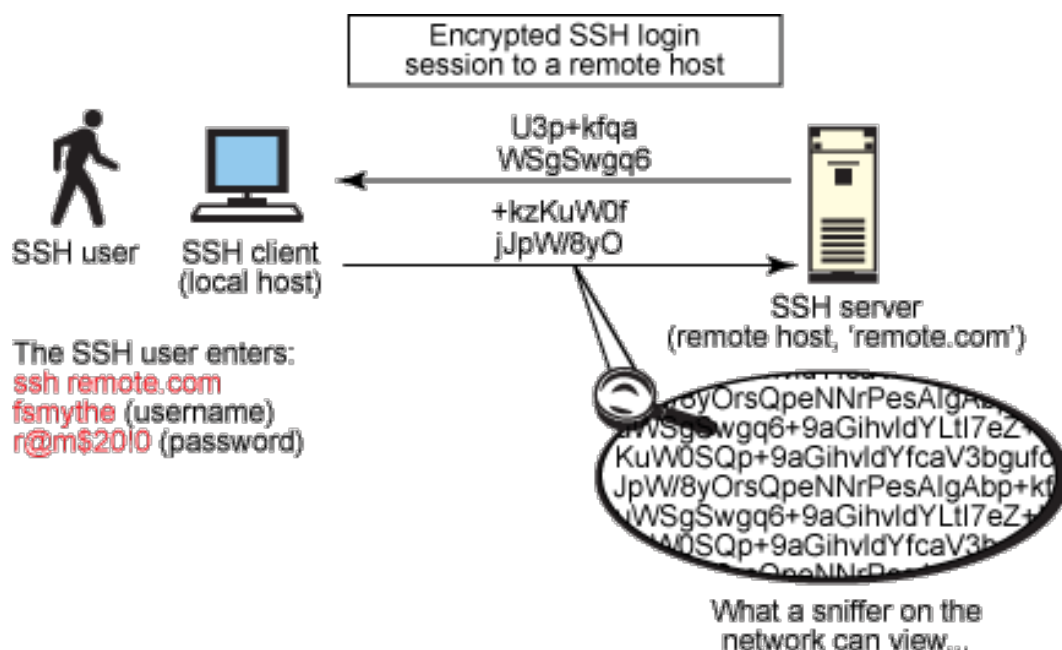


Figura 14: Demonstração de uma sessão SSH. Fonte: (BURANDE et al., 2014)

Como podemos ver na Figura 14 um usuário através de um serviço cliente de SSH pode fazer acesso a uma máquina que esteja com um serviço servidor de SSH em execução. O usuário então faz uma requisição de acesso e o servidor envia uma resposta requisitando *login* e senha, o usuário então responde enviando os dados de login e senha para o servidor para obter acesso ao *shell*. Nesse caso vemos que a criptografia na comunicação impede que um usuário ou programa mal-intencionado consiga visualizar os dados utilizando técnicas de *Sniff*² de rede. Ainda que as informações sejam capturadas elas são ilegíveis ao atacante que não possui a chave para desembaralhar a informação. Essa abordagem ainda impede que um atacante consiga alterar a informação "on the fly" visto que o algoritmo de criptografia identificaria qualquer alteração e descartaria o pacote. Desta forma o SSH garante uma comunicação segura, ainda que utilizando-a em redes inseguras temos a garantia de trocas de mensagens confidenciais e íntegras.

²Sniff: Captura de dados da rede

2.5.3 Comparando Telnet e SSH

Como apresentado, existem diferenças importantes entre os protocolos Telnet e SSH. A simplicidade do Telnet, o que em parte é uma de suas vantagens, nos traz o benefício de não termos um overhead de processamento e tráfego de dados para utilização do protocolo mas que acarreta em problemas de segurança relacionadas a confidencialidade e integridade. Já o SSH nos garante uma comunicação segura através do uso de criptografia em contraponto exige um maior processamento e tráfego de dados. A Tabela 1 traz um comparativo de ambos protocolos a respeito dos aspectos debatidos previamente.

Característica	Telnet	SSH
Porta padrão do serviço	23	22
Formato de dados	Transfere os dados em texto simples.	O formato criptografado é usado para enviar dados e também usa um canal seguro.
Autenticação	Por padrão não fornece autenticação entre os hosts	Usa criptografia de chave pública para autenticação.
Adequação da rede	Redes privadas são recomendadas.	Adequado para redes privadas e públicas.
Quanto a segurança	Comprovadamente vulnerável a ataques direcionados a confidencialidade e integridade na transmissão.	Superou muitos problemas de segurança do Telnet.
Uso de largura de banda	Baixo	Alto
Uso de processamento	Baixo	Alto

Tabela 1: Comparação entre Telnet e SSH

2.6 Comunicação Segura

Quando vamos realizar uma comunicação entre computadores distintos em redes consideradas inseguras estamos sujeitos a ameaças como as descritas na sessão 2.2. Desta forma é necessário utilizarmos alguns mecanismos ou ferramentas que evitem essa ameaças e garantam a segurança na comunicação.

2.6.1 Criptografia

A criptografia refere-se a técnicas seguras de informação e comunicação derivadas de conceitos matemáticos e um conjunto de cálculos baseados em regras para transformar mensagens de maneiras difíceis de decifrar. A criptografia é responsável por garantir que

uma comunicação só poderá ser visualizada por quem possuir a chave criptográfica para tal. Além disso a criptografia deve garantir que nenhuma informação possa ser inventada e inserida em uma troca de mensagens. Em outras palavras a criptografia é capaz de converter um texto simples em texto cifrado (ato de criptografar), e depois de converter o texto de volta (ato de descriptografar ou decodificar). A criptografia moderna se preocupa com os quatro objetivos a seguir:

1. **Confidencialidade:** a informação não pode ser entendida por qualquer pessoa ou sistema para quem não foi intencionalmente enviada a informação.
2. **Integridade:** a informação não pode ser alterada no armazenamento ou trânsito entre o remetente e o destinatário pretendido sem que a alteração seja detectada.
3. **Não-repúdio:** o criador/remetente da informação não pode negar em um estágio posterior suas intenções na criação ou transmissão da informação.
4. **Autenticação:** o remetente e o destinatário podem confirmar a identidade um do outro e a origem/destino da informação.

2.6.2 VPN

Redes virtuais privadas ou Virtual Private Networks (VPN) é uma rede de comunicações privada construída sobre uma rede de comunicações pública. Uma VPN tem capacidade de criar um canal de comunicação seguro entre duas redes distintas através de um meio inseguro utilizando uma espécie de túnel criptográfico por onde essa informação passa. A Figura 15 apresenta esse conceito. Como se pode ver, um usuário comunica-se com um provedor de VPN através de um túnel criptografado. O túnel gerado, pelo qual a informação trafega da ponta do usuário até o provedor de VPN, aplica sobre os dados em trânsito uma camada de criptografia de ponta-a-ponta. Esse túnel protege o usuário de atacantes que possam obter acesso ou realizar uma alteração na informação contida nas trocas de mensagens com o destino final ao qual o VPN se comunica diretamente utilizando os protocolos disponíveis desse destino. O VPN além de fornecer esse túnel criptográfico também fornece acesso ao destino desejado, podendo remover limitações que o usuário venha a ter relacionado a restrições de IP ou região (considerando um VPN de outra região).

2.7 Honeypots

Em segurança da informação *honeypot* é uma ferramenta utilizada para, entre outras coisas, obter informações de ataques realizados a uma determinada rede ou sistema, tais como métodos e ferramentas utilizadas. PROVOS et al. 2004 descreve *honeypot* como uma armadilha de rede, monitorada, que pode executar qualquer sistema operacional,

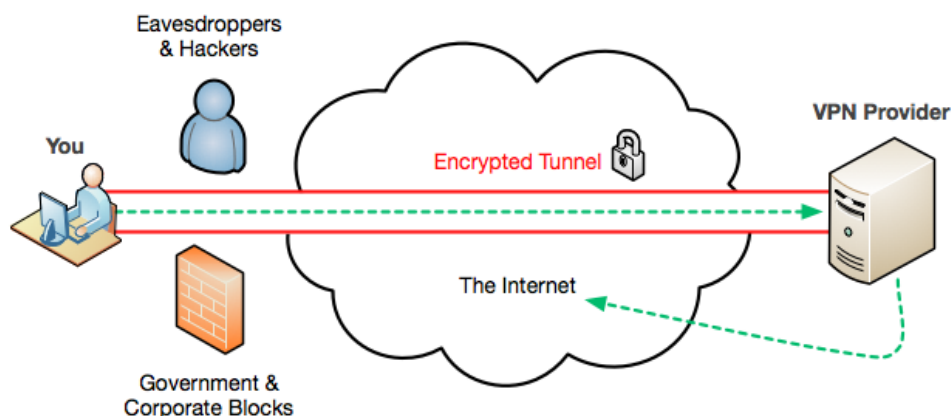


Figura 15: Topologia de uma VPN. Fonte: (HACKOHACKU, 2019)

qualquer tipo e número de serviços e que possui vários propósitos como; poder distrair os adversários de máquinas mais valiosas em uma rede, fornecer alertas precoces sobre novas tendências de ataque e exploração, ou permitir um exame aprofundado dos adversários durante e após a exploração de um *honeypot* para analisar suas atividades.

O seu funcionamento se baseia na premissa de que por serem recursos de segurança que não possuem valor de produção, nenhuma pessoa ou sistema deve estar se comunicando com eles, desta forma qualquer interação com o *honeypot* é suspeita por natureza. Qualquer tráfego enviado é provavelmente uma sondagem³, varredura de portas⁴ ou ataque. Qualquer tráfego iniciado pelo *honeypot* significa que o sistema provavelmente foi comprometido e o atacante está fazendo conexões de saída (SPITZNER, 2003a).

Do ponto de vista da aquisição de informação uma característica muito interessante de um *honeypot* é a capacidade que ele tem de monitorar os dados que entram e saem do mesmo, permitindo coletar informações relevantes a respeito do acesso a ele. Por exemplo, pode-se registrar as teclas de uma sessão interativa, mesmo que a criptografia seja usada para proteger o tráfego da rede. Também é comum capturar ferramentas e códigos utilizados e instalados no sistema durante sua exploração para posteriormente analisar o ataque. E por fim é possível detectar vulnerabilidades ainda não compreendidas, por exemplo, podemos detectar comportamento suspeito observando o tráfego de rede que deixa o *honeypot*, ainda que os meios da exploração nunca tenham sido vistos antes (PROVOS; HOLZ, 2007).

Uma das definições mais conhecidas de *honeypot* diz: "Um *honeypot* é um sistema de informação cujo valor reside no uso não autorizado ou ilícito desse recurso"(SPITZNER, 2003b), ou seja, para que se tenha resultados é desejável que um *honeypot* esteja conectado e acessível via internet e que seja encontrado e explorado de alguma forma, seja por ferramentas automatizadas que buscam vulnerabilidades previamente definidas ou manu-

³Sondagem: Verificação da existência de um sistema respondendo em determinado IP.

⁴Varredura de portas: Verificação de quais portas de um determinado sistema respondem a requisições.

almente por usuários maliciosos.

Pesquisas envolvendo *honeypots* possuem um grande valor no sentido de aprendizagem e são tipicamente utilizados por universidades, governos e departamentos militares para melhorar a prevenção, detecção e/ou resposta a ataques, através de sua principal característica; aquisição de inteligência, provendo assim uma plataforma de estudos de ameaça cibernética (MOKUBE; ADAMS, 2007).

Apesar das notáveis vantagens de se implantar o uso de um *honeypot* deve-se estar ciente da existência de suas desvantagens e o quão potencialmente perigosas são. Como dito, sendo um *honeypot* um sistema propositalmente vulnerável e acessível a qualquer usuário ou ferramenta automatizada na internet, imprevistos como, ataques muito sofisticados ou totalmente desconhecidos, podem ocorrer. Além disso é possível que um atacante detecte que seu ataque está sendo executado em um *honeypot* e mude seu comportamento, evitando assim efetuar o ataque da forma como faria. No melhor dos casos o atacante simplesmente percebe a existência de um *honeypot* e deixa o sistema, o mais rápido possível, tentando deixar o mínimo de rastros. No pior caso, consegue encontrar uma vulnerabilidade, compromete o sistema obtendo controle total e assim, pode efetuar os mais diversos tipos de ataques tanto a máquinas locais como externas podendo também evitar que o administrador⁵ detecte sua presença. Ambos os casos são totalmente indesejáveis e comprometem o experimento se fazendo então necessário tomar o máximo de cuidado possível para proteger a pesquisa, as máquinas locais e a própria internet.

2.7.1 Honeypots de baixa interatividade

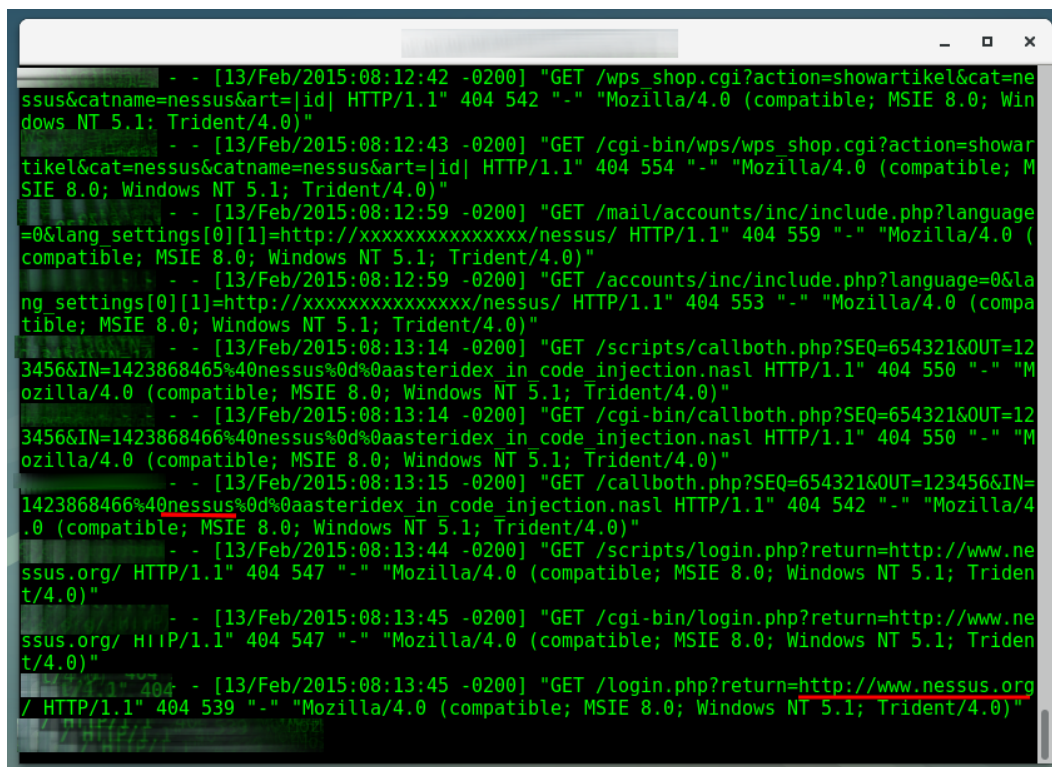
Um *honeypot* de baixa interatividade é um *honeypot* que simula um sistema operacional e/ou serviços que estejam rodando nele. Esse tipo de *honeypot* é o mais seguro por não fornecer ao atacante controle de um sistema real e sim apenas acesso a serviços emulados que não fornecem contato direto com o sistema hospedeiro. Esse tipo de *honeypot* geralmente simula um número limitado de serviços de rede e implementa apenas o suficiente dos protocolos da Internet, geralmente TCP e IP, para permitir a interação com o atacante e fazer com que ele acredite que está se conectando a um sistema real (PROVOS; HOLZ, 2007).

Utilizando essa abordagem não é possível capturar *malwares* ou ferramentas e nem seus códigos fontes ou analisar o comportamento de um usuário malicioso que tenha obtido controle da máquina, visto que o ambiente simulado não fornece um sistema operável onde seria possível explorar e efetivamente utilizar tais recursos. Este tipo de sistema também tem maior probabilidade de ser detectado como artificial pelo atacante, especialmente se o atacante for humano. O atacante pode achar que os serviços reais são emulados ou parcialmente implementados e, assim, perder o interesse em explorar.

No entanto com esse tipo de *honeypot* é possível capturar dados estatísticos a respeito

⁵ Administrador: Responsável por controlar e analisar resultados do *honeypot*

do número de ataques sofridos, as portas mais usuais para ataques, captura de tentativas de acesso por força bruta (podendo armazenar senhas mais utilizadas) e assinaturas⁶ de ferramentas automatizadas e *malwares*, evitando assim expor o sistema e a rede a riscos muito maiores como em um *honeypot* de alta interatividade. Na Figura 16 vemos um arquivo de *log* gerado pelo sistema operacional referente ao ruidoso processo de varredura executado pela ferramenta Nessus (TENABLE, 2019). Em vermelho foram destacadas informações que servem como assinaturas da ferramenta.



```

- - [13/Feb/2015:08:12:42 -0200] "GET /wps_shop.cgi?action=showartikel&cat=nessus&catname=nessus&art=id HTTP/1.1" 404 542 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:12:43 -0200] "GET /cgi-bin/wps/wps_shop.cgi?action=showartikel&cat=nessus&catname=nessus&art=id HTTP/1.1" 404 554 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:12:59 -0200] "GET /mail/accounts/inc/include.php?language=0&lang_settings[0][1]=http://xxxxxxxxxxxxxxx/nessus/ HTTP/1.1" 404 559 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:12:59 -0200] "GET /accounts/inc/include.php?language=0&lang_settings[0][1]=http://xxxxxxxxxxxxxxx/nessus/ HTTP/1.1" 404 553 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:14 -0200] "GET /scripts/callboth.php?SEQ=654321&OUT=123456&IN=1423868466%40nessus%0d%0aasteridex in code injection.nasl HTTP/1.1" 404 550 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:14 -0200] "GET /cgi-bin/callboth.php?SEQ=654321&OUT=123456&IN=1423868466%40nessus%0d%0aasteridex in code injection.nasl HTTP/1.1" 404 550 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:15 -0200] "GET /callboth.php?SEQ=654321&OUT=123456&IN=1423868466%40nessus%0d%0aasteridex in code injection.nasl HTTP/1.1" 404 542 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:44 -0200] "GET /scripts/login.php?return=http://www.nessus.org/ HTTP/1.1" 404 547 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:45 -0200] "GET /cgi-bin/login.php?return=http://www.nessus.org/ HTTP/1.1" 404 547 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"
- - [13/Feb/2015:08:13:45 -0200] "GET /login.php?return=http://www.nessus.org/ HTTP/1.1" 404 539 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"

```

Figura 16: Exemplo de assinatura da ferramenta de varredura de vulnerabilidades Nessus, em um log de sistema.

2.7.2 Honeypots de alta interatividade

Um *honeypot* de alta interatividade é um sistema o qual um atacante pode interagir como um sistema real. Isso significa que o *honeypot* não imita nenhum serviço, funcionalidade ou sistemas operacionais básicos, em vez disso, fornece sistemas e serviços reais. Na verdade esses sistemas são muito semelhantes aos sistemas reais, diferindo apenas no fato de que não tem valor de produção. Esses *honeypots* podem ser executados de forma simples, como um computador real que tenha um sistema operacional em execução, com serviços propositalmente vulneráveis e conectado a uma rede onde possa sofrer ataques, ou podem ser implementados como máquinas virtuais dentro de um sistema hospedeiro. Sua principal vantagem é a liberdade que ele dá para um atacante executar diversos ti-

⁶Assinatura: Uma característica que identifica um programa

pos de ataque, permitindo que o administrador do *honeypot* possa capturar comandos, binários, ferramentas. Por outro lado sua principal vantagem também é sua desvantagem. Fornecer a um atacante um sistema onde ele pode controlar é muito perigoso se cuidados não forem tomados. *Honeypots* de alta interatividade devem estar em redes isoladas de redes de produção, devem possuir um *firewall*, monitoramento de rede constante e preferencialmente estarem isolados em um sistema virtualizado para assim se ter mais controle. Com os devidos cuidados tomados, esse tipo de sistema pode fornecer um experimento muito mais rico em conhecimento e em possibilidade de enganar o atacante (SPITZNER, 2003a).

2.7.3 Honeypots físicos

Os *honeypots* físicos são aqueles que utilizam um sistema real como base para seu funcionamento. Por utilizarem um sistema real com todos seus recursos são tipicamente *honeypots* de alta interatividade e por definição são *honeypots* com alto custo computacional e oferecem maiores riscos.

2.7.4 Honeypots virtuais

Os *honeypots* virtuais são aqueles na qual o *honeypot* não executa diretamente sobre o hardware da máquina física. Dentre eles estão os *honeypots* de alta interatividade que executam sobre ferramentas de virtualização e os *honeypots* de baixa interatividade que apenas emulam determinadas partes de um sistema operacional real.

2.7.5 Comparando honeypots físicos e virtuais

Comparando os dois tipos de *honeypots*, é possível perceber que os virtuais possuem algumas vantagens sobre os *honeypots* físicos. Uma delas é o fato de que *honeypots* virtuais são menos custosos e mais fáceis de manter. A principal economia de recursos é o fato de que usando a abordagem de *honeypots* virtuais, várias máquinas podem ser executadas usando apenas uma máquina física. Isso também permite que múltiplos sistemas operacionais e suas aplicações possam rodar concorrentemente, aumentando as possibilidades de recebimento e estudo de ataques. Os *honeypots* virtuais também podem ser usados com maior vantagem em ambientes que utilizam redes privadas. Neste caso, determinados tráfegos de rede, vindos da Internet, podem ser redirecionados para *honeypots* que tratam distintos tipos de tráfego. Por outro lado, nem sempre é possível utilizar *honeypots* virtuais, principalmente quando eles são de alta interatividade. Isso porque dependendo das máquinas disponíveis para rodá-los, não é possível executar satisfatoriamente determinados tipos de ferramenta de virtualização pelo fato das máquinas não atenderem aos requisitos mínimos solicitados pela aplicação.

2.7.6 Honeynets

Assim como os *honeypots* de alta interatividade, as Honeynets também fornecem ao atacantes sistemas reais para atacar. Na verdade um *honeynet* é um conjunto de *honeypots* de alta interatividade, isolada de sistemas de produção, onde um atacante que compromete uma máquina pode tentar explorar outras máquinas, ou por assim dizer, outros *honeypots*. A principal vantagem de uma *honeynet* é que além de oferecer um ambiente muito mais realístico, como uma rede completa, é possível observar o comportamento de ataque a outras máquinas a partir de uma máquina já comprometida. Uma *honeynet* também pode ser virtualizada, utilizando um sistema hospedeiro com diversas *honeypots* virtuais ou até mesmo diversos sistemas hospedeiros com diversos *honeypots* virtuais. As desvantagens assim como um *honeypot* de alta interatividade, estão nos perigos de oferecer um sistema real para ser explorado por um atacante. Desta forma se faz necessário o monitoramento constante de máquinas e da rede. (SPITZNER, 2003c)

3 TRABALHOS RELACIONADOS

Neste capítulo serão descritos trabalhos relacionados e que foram considerados relevantes ao tema deste trabalho.

3.1 IoTPOT

Neste trabalho, com publicações em 2015 (PA et al., 2015) e 2016 (PA et al., 2016) é proposto o primeiro *honeypot* com foco em IoT, para analisar ameaças que utilizam ataques baseados em Telnet utilizando diversas arquiteturas de CPU.

Na sua descrição os autores utilizam o conceito de *honeypot* e *sandbox*. O *honeypot*, ao qual chamam de IoTPOT, é um *honeypot* de baixa interatividade que atua como o *frontend* do sistema cooperando com um *honeypot* de alta interatividade chamado de IoTBOX, operando com diversos ambientes virtuais com diferentes arquiteturas de CPU como ARM, MIPS, PPC, entre outras, atuando como um *sandbox* de análise de malware.

A Figura 17 ilustra uma visão geral de sua estrutura de funcionamento. O *Frontend Responder* é o responsável por gerenciar as conexões recebidas diretamente da internet. Então essas conexões são tratadas de acordo com seu tipo de requisição pelo *Profiles* é o ente responsável por armazenar:

1. Os *Banners Profiles*, que determinam as respostas do *honeypot* para interações de *banner*, são conhecidos como opções de Telnet, mensagem de boas-vindas ou *prompt* de *login*.
2. Os *Authentication profiles*, que determinam como responder as tentativas de autenticação recebidos.
3. Os *Command interaction profiles*, que determinam as respostas aos comandos recebidos, verificando em um conjunto de comandos, as respostas correspondentes.

Após esse tratamento as conexões que são autenticadas são então direcionadas a uma máquina virtual correspondente através do gerenciador Manager.

O “*Profiler*” analisa a interação entre o *Frontend Responder* e o IoTBOX para atualizar o *Profiles* sempre que um novo comando é enviado. Este ente também tem uma

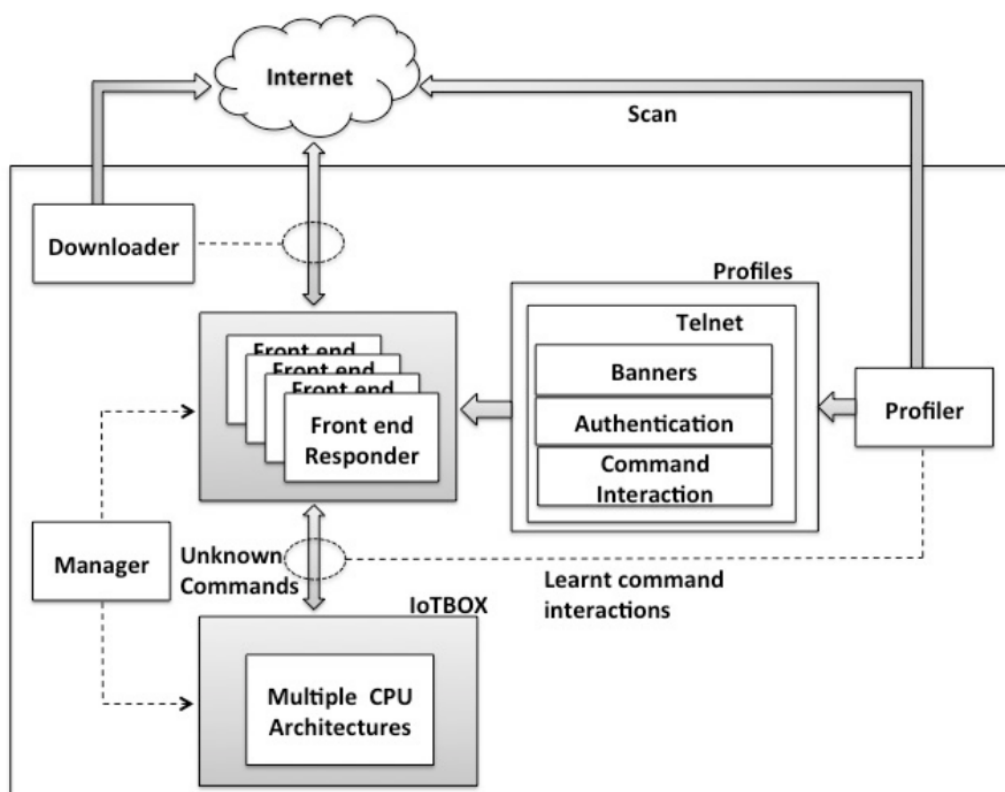


Figura 17: Visão geral do IoTPOT. Fonte:(PA et al., 2015)

funcionalidade peculiar, ele é o responsável por descobrir novos banners de dispositivos que existem na internet. Para isso ele utiliza dois modos de operação; o *Active Scan Mode* que coleta *banners* em diversas redes diferentes e o *Visitor Scan Mode*, que varre o *host* que visitou o *honeypot*. Finalmente o *Downloader* é o ente responsável por examinar as interações de *download* que ocorrem no *honeypot*, fazendo uma cópia de todos arquivos, como por exemplo, binários de *malwares*.

O funcionamento do IoTBOX, apresentado na Figura 18, demonstra a topologia do sistema e como as requisições previamente efetuadas no *Frontend Responder* são distribuídas pelas diferentes arquiteturas presentes em diferentes máquinas virtuais. O *Access Controller* é o responsável por distribuir os pacotes entre as diferentes máquinas conectadas ao *Virtual Switch* que é um *switch*¹ de rede virtualizado. O *Access Controller* também funciona como um limitador de acesso para garantir segurança; evitando ataques de negação de serviço a IPs externos, além de redirecionar varredura de portas que ocorrem na porta 23 para o *Dummy Server*, responsável por monitorar como se propagam os ataques via Telnet.

Na execução da ferramenta os autores subdividiram o funcionamento em três etapas:

1. Intrusão: Nesse estágio o atacante tenta obter acesso ao sistema utilizando sequen-

¹É um dispositivo que conecta dispositivos em uma rede de computadores usando a troca de pacotes para receber, processar e encaminhar dados para o dispositivo de destino.

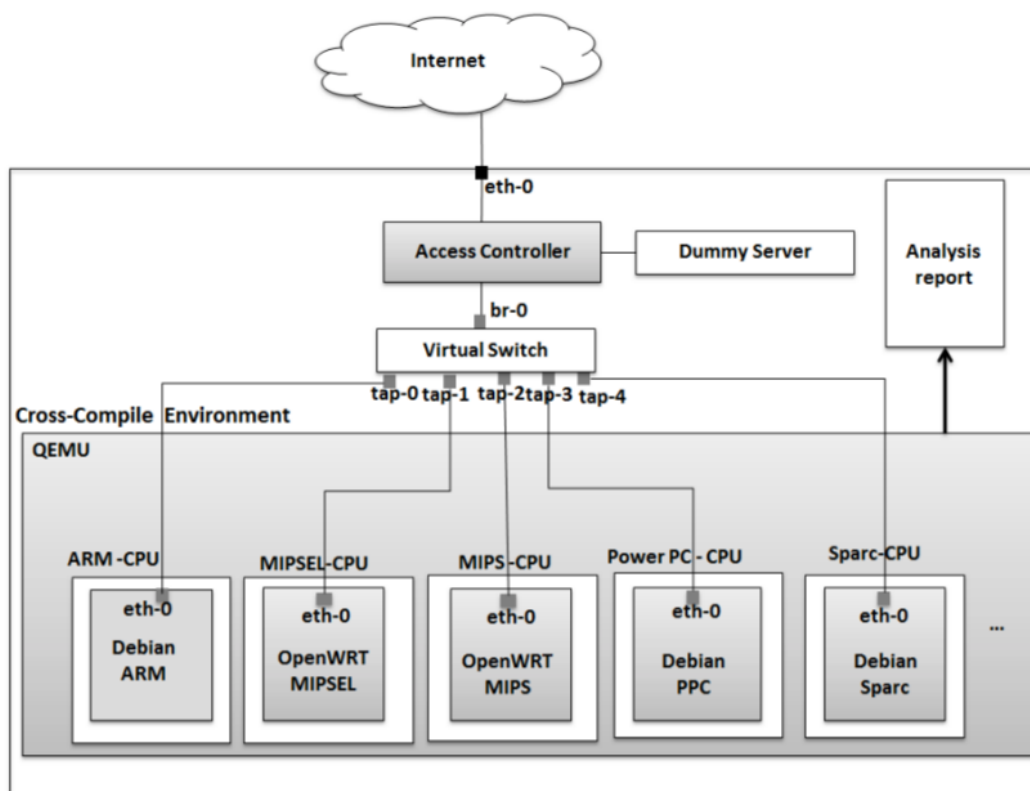


Figura 18: Topologia interna do sistema IoTPOT. Fonte:(PA et al., 2015)

cias de usuários e senhas.

2. Infecção: Após ter obtido acesso, o atacante tenta customizar o sistema utilizando diversos comandos de Telnet para fazer *download* dos binários dos *malwares*.
3. Monetização: Depois de todos os passos efetivados o atacante tenta agora executar suas ferramentas para obter o resultado desejado, por exemplo, ataques de negação de serviço, rastreamento de outras máquinas vulneráveis, etc.

Como resultados, os autores obtiveram uma série de dados interessantes. Durante os 81 dias de operação o sistema recebeu 79935 visitas de IPs diferentes, contabilizando 481521 tentativas de *download* de binários de *malware*. Foram capturados 106 *malwares* distintos capazes de rodar em 11 arquiteturas de CPU diferentes, dos quais 88 eram desconhecidos na base de dados do VirusTotal (VIRUSTOTAL, 2017), além de inúmeros padrões de usuário e senha para força bruta. A análise realizada pelo IoTBOX em 25 amostras revelou que as mesmas são utilizadas para executar 11 tipos diferentes de ataques DDoS, varreduras TCP na porta 23, 80, 8080, 5916 e varreduras UDP nas porta 123, 3143.

Com isso puderam concluir que:

1. Há pelo menos cinco famílias distintas de *malware* que se espalham via Telnet.

2. Seu comportamento comum é executar DDoS e a propagação da contaminação via Telnet .
3. Algumas famílias evoluem rapidamente, se atualizando frequentemente e distribuindo binários para uma variedade de arquiteturas de CPU.

3.2 SIPHON

Scalable high-Interaction honeypot platform for IoT devices (GUARNIZO et al., 2017) apresenta uma arquitetura de *honeypots* de alta interação para IoT que utiliza dispositivos que estão fisicamente em um local porém estão acessíveis via Internet através dos chamados *wormholes*. Esses *wormholes* ficam distribuídos por todo o mundo e tem o objetivo de expor um número reduzido de elementos físicos sobre um grande número de endereços IP geograficamente distantes.

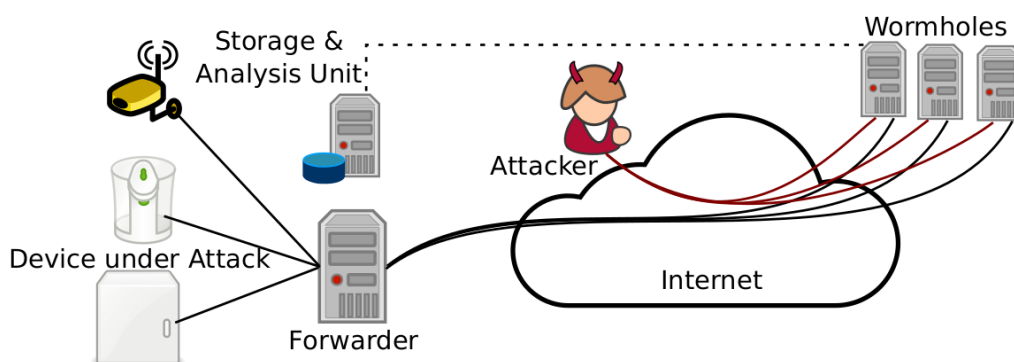


Figura 19: Visão geral do SIPHON. Fonte: (GUARNIZO et al., 2017)

A Figura 19 apresenta o esquemático de funcionamento do projeto. O atacante (Attacker) busca por IPs vulneráveis na internet e possivelmente encontra diversos IPs que constituem os Wormholes e que estão distribuídos ao redor do mundo redistribuindo o tráfego de rede de infra-estruturas de provedores de serviços na nuvem. Esse tráfego redirecionado aponta para o Forwarder que é o responsável por garantir que o tráfego entre o *wormhole* e o dispositivo IoT seja reescrito em tempo real para ocultar o fato de que os dispositivos estão fisicamente localizados em algum outro lugar. Se necessário, o Forwarder também pode atuar como um *man-in-the-middle*² para análise, porque terá acesso às chaves privadas dos certificados de comunicação criptografadas usados pelos dispositivos IoT. Unidade de Armazenamento e Análise (Storage & Analysis Unit) obtém registros do tráfego e logs gerais dos *wormholes* e agrega os dados para análises *off-line*. O dispositivo local sob ataque (Device under Attack), na arquitetura SIPHON, pode ser um dispositivo IoT comercial qualquer. Ele irá expor um ou mais serviços através da rede, provavelmente

²Man-in-the-middle: Técnica utilizada para visualizar a comunicação entre dois entes em uma rede.

por TCP. O dispositivo é capaz de manter uma série de conexões simultâneas em qualquer ponto no tempo. Uma das características mais interessantes do projeto é sua capacidade de escalabilidade. Para n dispositivos físicos IoT e m *wormholes*, o número de dispositivos apresentados ao atacante, sem expor o mesmo dispositivo duas vezes no mesmo *wormhole*, pode ser de até $n * m$. Por exemplo, se $n = 100$ e $m = 1000$ até 100000 serviços podem ser expostos à internet. A quantidade de serviços por *wormhole* é limitada a cerca de 65000 devido a restrições de numeração de portas. O número de *wormholes* por dispositivo físico é limitado pela quantidade de tráfego paralelo que pode ser gerenciado por cada dispositivo físico.

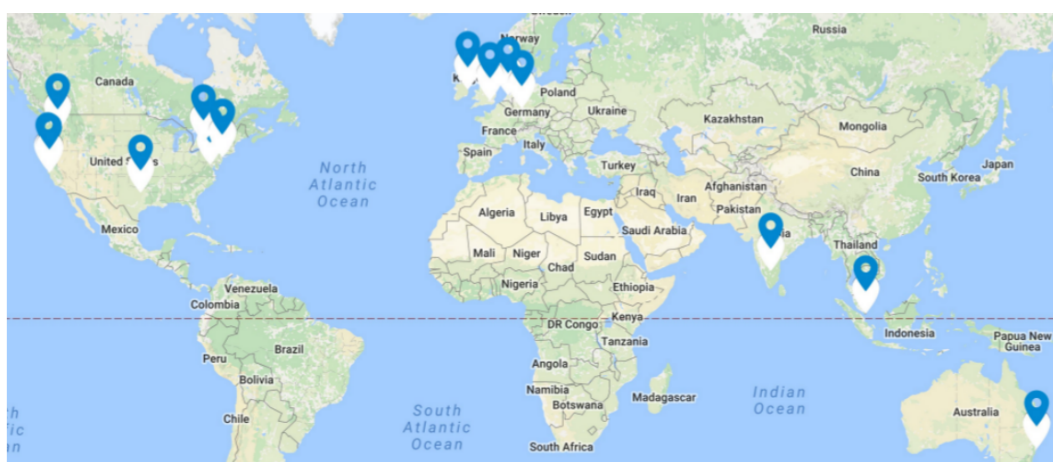


Figura 20: Locais onde foram colocados wormholes. Fonte: (GUARNIZO et al., 2017)

Conforme vemos na Figura 20 os *wormholes* foram distribuídos por 16 cidades em 9 países: EUA (San Jose, Boardman, Ashburn, Nova York, São Francisco, Dallas, Fremont, Newark), Canadá (Toronto), Europa (Frankfurt, Londres, Dublin, Amsterdã), Ásia (Singapura, Bangalore) e Austrália (Sydney). Dessa forma foi possível capturar o comportamento de ataques em diferentes regiões do mundo para análise.

Como resultado foi observado que determinadas cidades receberam mais ataques do que outras, por exemplo, Frankfurt recebeu a maioria das conexões (quase 600000), enquanto San José nos EUA recebeu o menor número de conexões (cerca de 50000) o que comprova que o comportamento de atacantes diferem de região para região. Outra característica observada foi a origem dos ataques. A Figura 21 mostra que a grande maioria dos ataques foram originados da China, cerca de 70.6% seguido de EUA e Holanda com 8% e 2.5% respectivamente.

Também relevantes e contrastante com os resultados do IoTPOT, citado na Sessão 3.1, foram os resultados obtidos com relação as portas mais buscadas pelos atacantes às câmeras. A Figura 22 mostra que quase todas as tentativas de ataque se deram na porta 22 referente ao serviço SSH nem ao menos citando a porta 23 referente ao serviço Telnet, muito explorado no IoTPOT. Essa divergência entre os dois trabalhos pode indicar uma mudança de comportamento dos atacantes, na data em que foram publicados re-

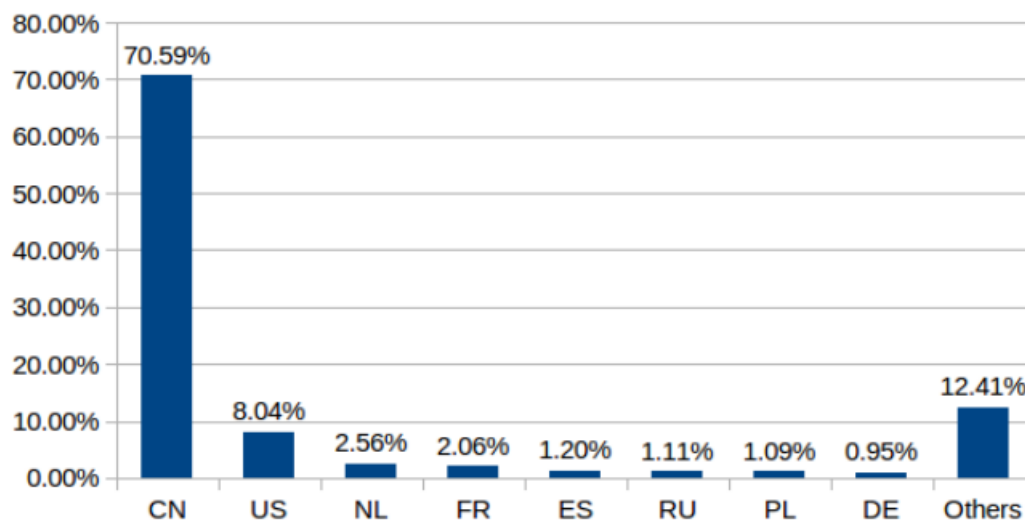


Figura 21: Países de origem dos ataques. Fonte: (GUARNIZO et al., 2017)

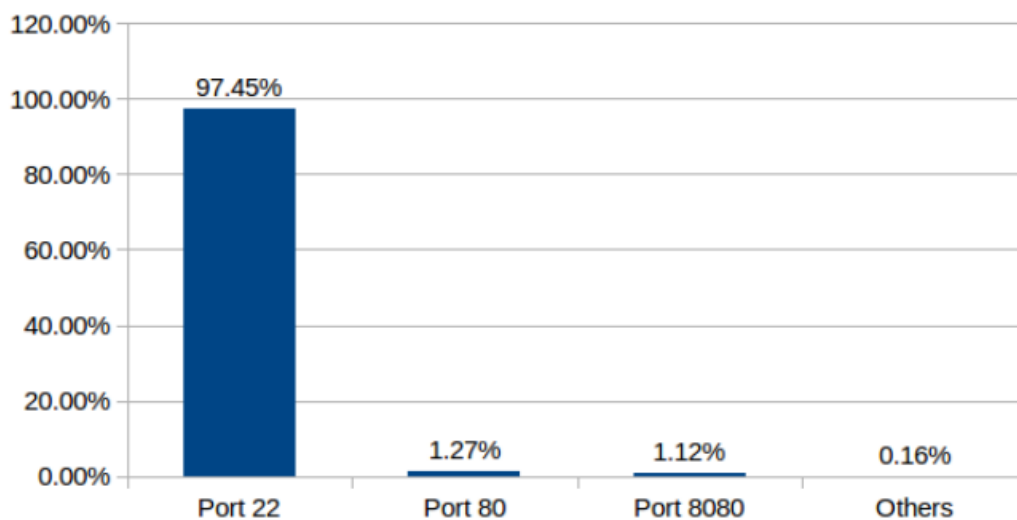


Figura 22: Portas mais atacadas. Fonte: (GUARNIZO et al., 2017)

sultados do IoT POT em relação ao SIPHON, ou uma modernização nos protocolos de comunicação dos elementos IoT, de Telnet para SSH. Como conclusão, foi observado que a arquitetura proposta foi bem sucedida ao atrair um alto número de conexões e coletar cerca de 20GB de dados em 2 meses de uso. Os autores demonstram, em suas conclusões, ter preocupações com sua metodologia, argumentando que um invasor inteligente pode detectar o *honeypot* de maneiras diferentes. Por exemplo, o atacante pode identificar que dois *wormholes* representam o mesmo dispositivo físico ao fazer *login* simultâneo nos mesmos dispositivos a partir de dois *wormholes* diferentes. Entretanto sugerem algumas modificações para mitigar esse problema, como utilizar um dispositivo IoT por *wormhole*. Por fim, apesar de utilizarem um *honeypot* de alta interatividade com grande quantidade de dados capturados, o artigo não apresentou resultados de estudo sobre ferramentas e códigos capturados em ataques.

3.3 Projeto honeyTARG Honeynet

O Projeto honeyTARG *Honeynet* (CERT.BR, 2019) é uma resposta brasileira ao Projeto Global *Honeynet* (PROJECT, 2019), focado em usar *honeypots* de baixa interação para coletar informações sobre o abuso da infraestrutura de Internet por atacantes e *spammers*. O projeto é administrado pelo CERT.br e se divide em:

1. *Spampots Project* é um sistema que usa *honeypots* de baixa interação para coletar dados relacionados ao abuso da infra-estrutura da Internet por *spammers*³ além de ajudar a desenvolver a pesquisa de caracterização de *spam*⁴ de email e identificação de *botnets*⁵, *phishing*⁶ e *malware*.
2. *Distributed Honeypots* é um consórcio de *honeypots*, distribuído por diversos estados brasileiros e tem como objetivo obter as tendências atualizadas de ataques bem como fazer sua análise e aumentar a capacidade de detecção de incidentes e correlação de eventos.

O CERT.br disponibiliza diariamente resultados compilados dos diversos *honeypots* espalhados pelo Brasil. A Figura 23 demonstra as principais portas que sofreram ataque no dia 15/11/2017.

Outro resultado interessante pode ser observado na Figura 24 onde é apresentado os sistemas operacionais dos quais se originam os ataques. Mais de 98% dos ataques partiram de um sistema operacional definido como “não identificado” (unknown) o que pode significar que os ataques estão partindo de dispositivos com um sistema operacional pouco

³Usuários ou ferramentas automatizadas que produzem *spam*

⁴Envio de mensagens em massa.

⁵Botnets: coleção de programas conectados à Internet que comunicam-se com outros programas similares, a fim de executar tarefas maliciosas.

⁶Phishing: tentativa de obtenção de informação pessoal, por criminosos, através de engenharia social.

Destination TCP Ports

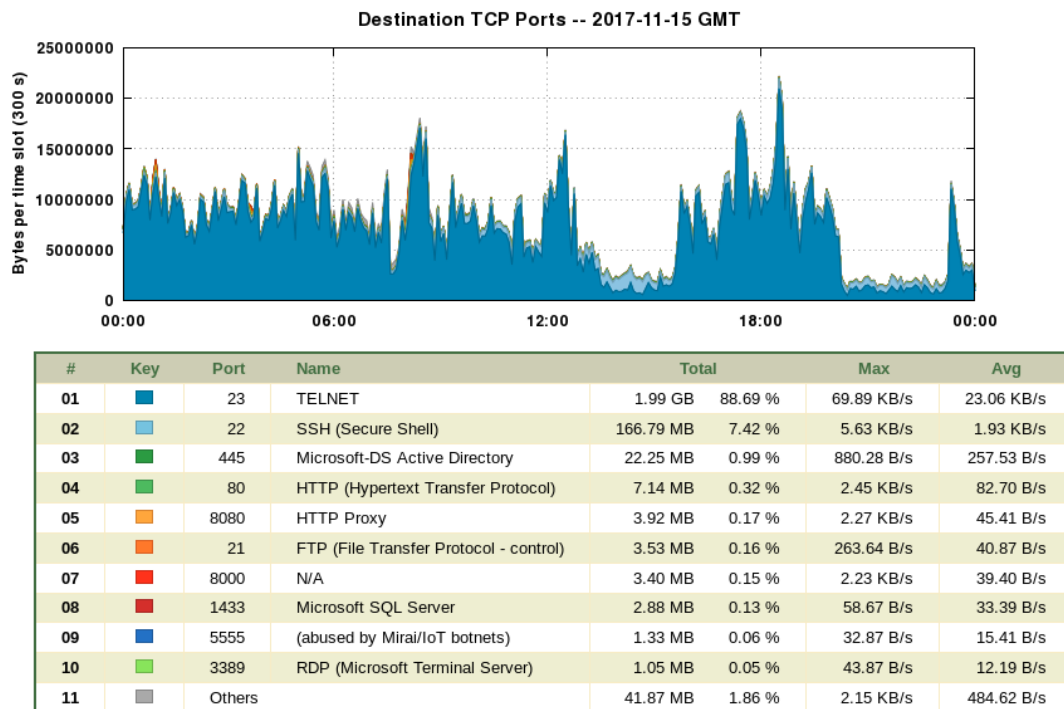


Figura 23: Principais portas atacadas no dia 15/11/2017. Fonte: (PROJECT, 2019)

Source Operating Systems (Other Platforms)

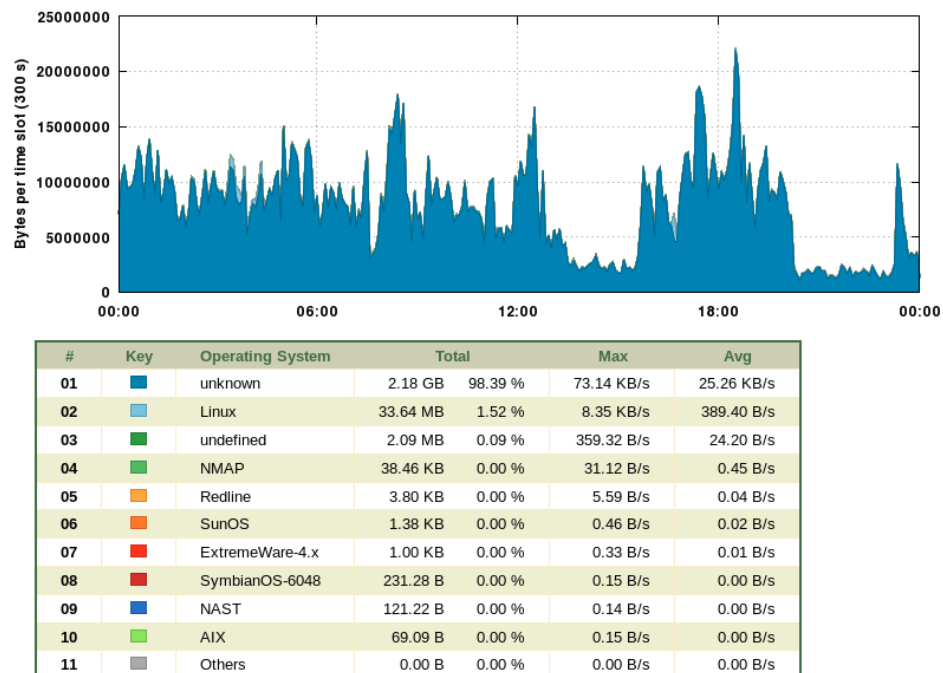


Figura 24: Principais sistemas operacionais os quais se originaram os ataques no dia 15/11/2017. Fonte: (PROJECT, 2019)

conhecido ou uma variação de um sistema opensource modificado que possuem fingerprints não identificáveis na base de dados.

Porta	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
23	82.76 %	68.86 %	85.26 %	93.99 %	93.98 %	90.56 %	86.47 %	92.80 %	86.02 %	86.32 %	83.67 %
22	8.45 %	19.19 %	4.98 %	2.97 %	2.78 %	5.82 %	8.43 %	4.49 %	7.47 %	9.17 %	11.24 %
445	1.14 %	1.86 %	1.26 %	0.59 %	0.34 %	0.45 %	0.56 %	0.62 %	0.95 %	0.66 %	0.64 %
80	0.56 %	0.66 %	0.23 %	0.30 %	0.32 %	0.37 %	0.31 %	0.39 %	0.86 %	0.95 %	0.54 %
8080	0.36 %	0.19 %	-	0.17 %	0.13 %	0.16 %	0.11 %	0.08 %	0.28 %	0.15 %	0.06 %

Tabela 2: Porcentagem de acessos às portas capturados aleatoriamente no ano de 2018

A Tabela 2 apresenta os resultados coletados em dias aleatórios durante os meses de fevereiro à dezembro no ano de 2018. Esse resultado demonstra que a porta 23 possui um tráfego muito superior que as outras portas, indicando que atacantes buscam, em sua maioria, por vulnerabilidades de Telnet seguidos do protocolo SSH (porta 22).

3.4 HIoT POT

Neste artigo (GANDHI et al., 2018), é apresentada a implementação de um *honeypot* de pesquisa que é usado para aprender táticas utilizadas por atacantes para atacar elementos IoT. O HIoT POT cria um ambiente fictício onde elementos IoT virtuais são usados para distrair atacantes evitando ataques ao ambiente com elementos IoT reais. Essa abordagem mantém os atacantes ocupados com esses elementos virtuais, evitando ataques aos elementos reais além de capturar informações a respeito do atacante e alertar usuários legítimos a respeito de ameaças.

A Figura 25 apresenta a visão geral do HIoT POT. Na figura o HIoT POT está implementado sobre um Raspberry Pi3 e contém diferentes elementos internos cada qual apresentando funcionalidades específicas. Seu funcionamento se baseia em avaliar tentativas de acesso. Ao receber uma tentativa de acesso, o sistema compara se esse usuário pertence ao banco de usuários - representado na figura por USER DB - se pertence o acesso é encaminhado ao ambiente IoT real (*Real IoT Environment*). Se o usuário que está acessando não estiver contido no banco de usuários o acesso é encaminhado o ambiente fictício (*Fake IoT Environment*), onde estão os elementos IoT virtuais. Esse usuário considerado malicioso terá suas ações capturadas e gravadas no banco de logs (*Logs e Chat Details*) e um alerta será encaminhado aos outros usuários do ambiente real informando para não se comunicar com esse usuário.

Na Figura 26 os autores comparam o HIoT POT com outros sistemas similares. De acordo com os mesmos, o nível de envolvimento do sistema é alto em comparação à outros, que são considerados baixo. Também no sistema proposto, existem alguns novos parâmetros como, mecanismo de alerta, manutenção do arquivo de log, dissuasão ao intruso e manutenção do histórico do sistema. Por causa de todas essas funcionalidades, não presentes em outros sistemas, a complexidade é dita alta.

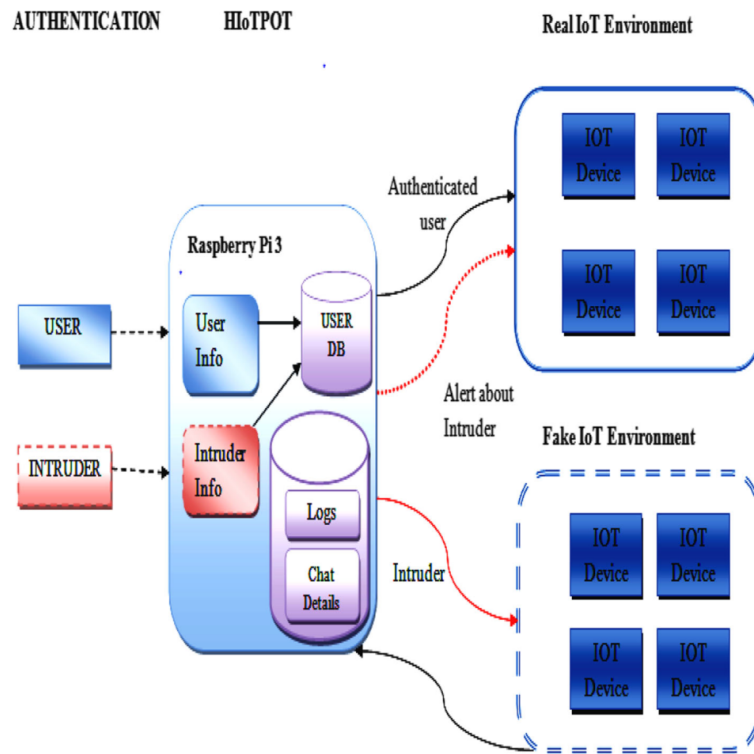


Figura 25: Visão geral do HIoT POT. Fonte: (GANDHI et al., 2018)

	Existing system	Proposed system
Level of involvement	Low	High
Alert mechanism	No	Yes
Maintain log file	No	Yes
Deterrence to intruder	No	Yes
Maintain chat history	No	Yes
Complexity	Low	High

Figura 26: Comparação do HIoT POT com sistemas existentes. Fonte: (GANDHI et al., 2018)

Como resultados os autores apresentam detalhes de *logs* gerados pelo sistema contendo detalhes de IP, data e hora de acessos recebidos em um único dia mostrando uma relação de 2 para 1 com 46 usuários legítimos e 23 intrusos. Eles destacam que por meio de seus arquivos de *log* é muito fácil aprender táticas e metodologias adotadas por atacantes e seus objetivos.

Os autores concluem o artigo dizendo que seus objetivos são de tornar o ambiente IoT seguro e ajudar a comunidade a aprender metodologias utilizadas por atacantes para evitar ataques futuros. Como trabalhos futuros planejam um método para garantir a segurança na comunicação de elementos IoT informando que suas limitação é o foco apenas na detecção de intrusão.

3.5 Intrusion Detection in Encrypted Accesses with SSH Protocol to Network Public Servers

Neste trabalho (FOROUSHANI; ADIBNIA; HOJATI, 2008) os autores apresentam uma nova abordagem para solucionar o problema da detecção da intrusão em conexões criptografadas. Conforme os autores, os sistemas de detecção de intrusão ou *intrusion detection system* (IDS) é uma solução para encontrar esses problemas relacionados a ataques direcionados a aplicações de rede. No entanto os IDSs são incapazes de reagir eficientemente a acessos seguros com protocolos de criptografia, porque eles não podem verificar o conteúdo de um pacote. Desta forma é apresentada uma nova abordagem para detectar comportamentos anômalos em acessos criptografados utilizando protocolo SSH em servidores públicos, como servidores HTTP, servidores FTP e servidores de banco de dados. Utilizando a abordagem proposta, primeiro o sistema extrai informações de cada cliente SSH, que consiste no tamanho dos dados transferidos e no intervalo de tempo entre as mensagens. Então várias ações são identificadas com base na similaridade das informações. Finalmente, os ataques são detectados de acordo com as assinaturas de intrusão, geradas a partir da frequência de acessos e especificações do tráfego TCP.

A Figura 27 apresenta a configuração utilizada e os passos que o sistema utiliza para a detecção. O tráfego gerado na rede é separado para identificar um usuário específico baseado em seu IP e em suas conexões o protocolo SSH é extraído das outras conexões. Uma tabela é atualizada com as informações desse cliente armazenando seu IP, um identificador de usuário e o tamanho do código de autenticação das mensagens do protocolo SSH. A atividade de um usuário é então separada das demais atividades da rede e uma matriz é gerada utilizando cada atividade, contentando o tamanho dos dados transferidos entre os processos do servidor e do cliente. Uma análise da frequência com o objetivo de reduzir o efeito do tamanho do bloco. Se uma atividade é realizada repetidamente, o tamanho dos dados é o mesmo, mas o tamanho do bloco é diferente, desta forma a chance de um falso positivo é reduzida. Os dados gerados na análise de frequência são então ar-

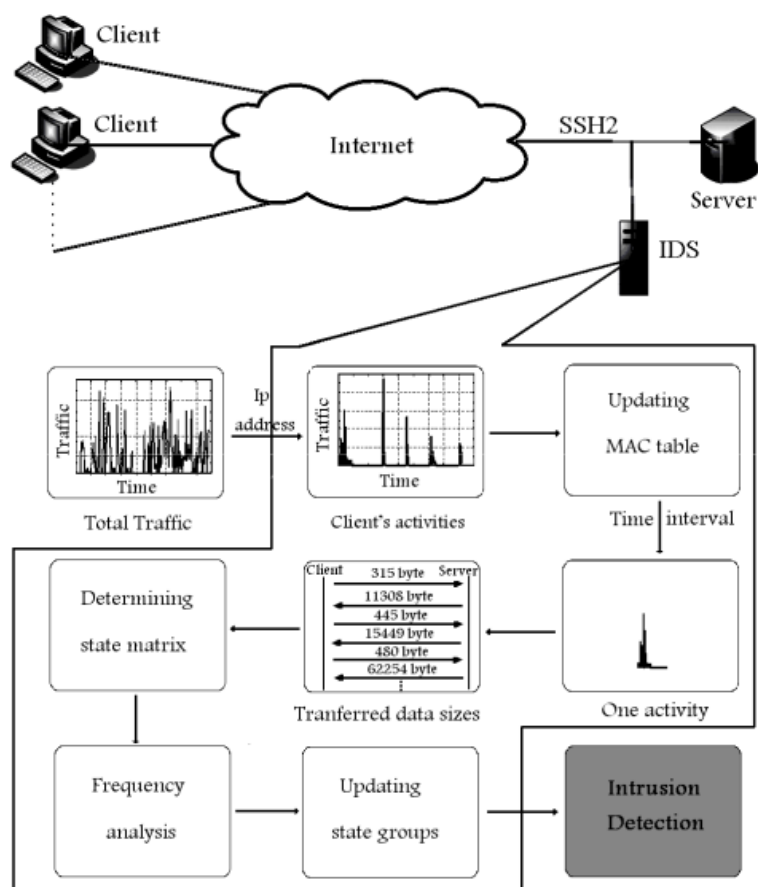


Figura 27: Configuração da rede usada para avaliação inicial do sistema proposto. Fonte: (FOROUSHANI; ADIBNIA; HOJATI, 2008)

mazenados no grupo de estados. A tabela de grupo mantém o ID, o número de membros e a matriz do grupo. Com a tabela de grupo, os tempos de ocorrência de uma atividade individual podem ser determinados. Com isso uma intrusão pode ser detectada.

Como conclusão os autores comprovaram a possibilidade de detectar intrusão em redes que utilizam protocolo criptografado SSH analisando tamanho e tempo dos dados transferidos sem descriptografá-los e os resultados revelaram que com o sistema proposto é possível identificar diferentes tipos de intrusões.

3.6 Securebox: Toward Safer and Smarter IoT Networks

Neste trabalho (HAFEEZ et al., 2016) os autores apresentam o Securebox, uma plataforma para proteger redes IoT. A plataforma apresenta vários recursos incluindo isolamento do dispositivo e verificação das comunicações entre elementos. A plataforma intercepta qualquer solicitação de conexão que um dispositivo IoT conectado venha a realizar a um destino remoto e verifica nas várias políticas de segurança se estas conexões correspondem à conexão solicitada. Quando uma atividade suspeita é detectada a plataforma coloca em quarentena essa tentativa de acesso e informa ao usuário para aumentar sua conscientização sobre segurança de dispositivos e dados. O Securebox é dividido em duas partes principais. O Securebox *Frontend* (SF) responsável pela interação direta com os elementos IoT e o *Security and Management Service* (SMS) que funciona como um *backend* para o sistema, fornecendo escalabilidades e funcionalidades em nuvem. O SF é descrito como uma implementação para ser executado sobre um hardware de baixo custo, desta forma a implementação é leve e simplificada. Sua principal função é controlar o tráfego dos elementos conectados a ele baseado em políticas de seguranças armazenadas. Para isso o SF pode solicitar análise do tráfego ao SMS em tempo real para que políticas de segurança recebidas, em resposta a essas solicitações de análise, sejam inseridas e armazenadas em cache localmente. Além disso o SF também pode criar isolamento de dispositivo, controle das comunicações dispositivo a dispositivo, descoberta e criação de perfil de dispositivo.

A Figura 28 demonstra a arquitetura interna e como o Securebox fica alocado na rede. O *Management console* é a interface entre o usuário e o sistema possibilitando ao usuário personalizar o sistema. Ele fornecido por meio de um aplicativo para *smartphone* para melhorar a experiência do usuário. Este aplicativo fornece classificação de segurança para dispositivos conectados com base na atividade do dispositivo. Ele também gera avisos quando uma atividade suspeita é detectada e colocada em quarentena, para aumentar a conscientização do usuário sobre o dispositivo e a segurança dos dados. O *SDN Controller* é responsável por gerenciar o tráfego dos elementos conectados a rede com e sem fio lidando com o monitoramento de tráfego e comunicação com o SMS. Já o *Policy Database*, como o próprio nome já diz, é o responsável por armazenar informações a respeito

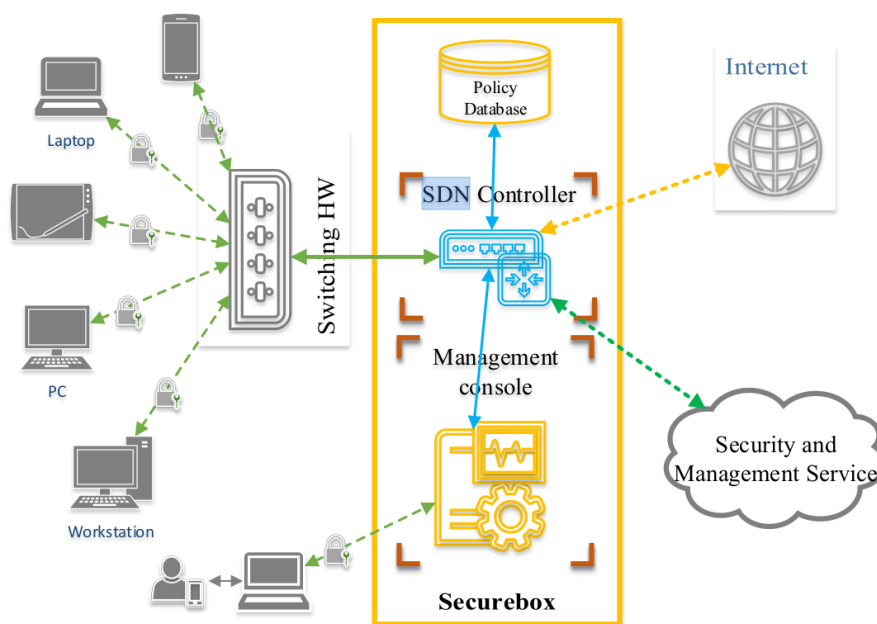


Figura 28: Arquitetura interna do frontend do Securebox. Fonte: (HAFEEZ et al., 2016)

das políticas de segurança obtidas do SMS e/ou configuradas pelo usuário.

O SMS é um serviço em nuvem que tem por objetivo agregar escalabilidade ao sistema processando as requisições do SF a respeito de requisições de conexões realizadas por elementos pertencentes a rede. Ao receber uma nova solicitação de análise, o gerenciador de nuvem valida a solicitação usando o servidor de certificação. O SMS recebe dados de tráfego de rede de todos os SFs conectados. Ele combina esses dados com dados coletados de fontes externas como por exemplo, bancos de dados de assinaturas de vírus, bancos de dados de malware para executar vários serviços de análise e detectar novas ameaças. Desta forma o SMS utiliza uma abordagem colaborativa para minimizar as chances de um invasor usar as mesmas técnicas em elementos distintos.

Como resultado os autores criaram um protótipo parcial e funcional do sistema. Eles verificaram que o Securebox é suscetível a um aumento na latência de acesso a links pelo usuário, pois as tarefas de análise de tráfego são transferidas para o serviço remoto. No entanto demonstram que os impactos na latência podem ser minimizados modificando algumas configurações e utilizando o banco de dados local para armazenar políticas do SMS. Os autores conseguiram comprovar também a possibilidade de escalar o sistema utilizando a nuvem, visto que a cada pico de consumo de processamento novas requisições de conexão foram redirecionadas para serem processadas no SMS. Finalmente os autores também conseguiram comprovar que o sistema foi capaz de prover tolerância a falhas pois se um ente principal ficava inativo nas operações dentro do SMS, uma réplica de *backup* o substituiu para análise de tráfego. Se o antigo ente principal voltasse a atividade ele começava a servir como réplica de *backup* para o novo ente principal.

3.7 Relação de funcionalidades

Nesta sessão foram identificadas características presentes e um ou mais trabalhos aqui apresentados. A Tabela 3 apresenta essas características relacionando os trabalhos que apresentam cada uma delas.

	IoTPOT	SIPHON	honeyTARG	HioTPOT	I.D.E.A.S.P.N.P.S.	Securebox
Captura de tentativas de acesso	x	x	x	x		
Deteção de intrusão	x	x		x	x	x
Captura de comportamento	x	x		x		
Múltiplas arquiteturas	x	x				
Ambiente virtualizado	x		x	x		x
Distração de atacantes				x		
Múltiplas frentes de ataque externas		x	x			
Transparente ao usuário	x	x		x		x
Escalável		x		x		x
Mecanismo de alerta			x	x	x	x
Aprendizagem de regras				x		x
Controle de acesso				x		x
Análise de tráfego em tempo real						x
Isolamento entre elementos						x

Tabela 3: Comparativo de funcionalidades entre trabalhos

1. **Captura de tentativas de acesso:** A presença de um *honeypot* garante a capacidade da captura de tentativas de acesso vindos do meio externo gerando log de requisições direcionadas a portas de serviços expostos por elementos da rede.
2. **Deteção de intrusão:** Um sistema capaz de detectar quando um usuário consegue obter acesso (de forma indevida) a recursos do sistema.
3. **Captura de comportamento:** *Honeypots* de alta interatividade possibilitam além da captura de tentativas de acesso também a captura de arquivos e ferramentas além de como um atacante se comporta utilizando o sistema comprometido.
4. **Múltiplas arquiteturas:** Elementos IoT podem ser implementados sobre diferentes arquiteturas de processador. Um sistema que emula múltiplas arquiteturas possui uma gama maior de possibilidade de binários que podem ser executados para serem capturados e estudados.
5. **Ambiente virtualizado:** Sistemas com ambientes virtualizados criam uma camada extra de proteção na execução de serviços e binários maliciosos, reduzindo muito a chance de comprometimento do sistema hospedeiro.
6. **Distração de atacantes:** *Honeypots*, se usados em paralelo com sistemas de produção podem servir para distrair atacantes de máquinas reais tendo em vista que o atacante irá despender de tempo e recursos computacionais para atacar uma máquina sem valor produtivo.

7. **Múltiplas frentes de ataque externas:** Sistema que ofertam múltiplas frentes de ataque possibilitam uma maior capacidade de coleta de dados, podendo ainda obter datasets de diferentes regiões valendo-se de servidores externos e IPs distintos.
8. **Transparente ao usuário:** A transparência ao usuário implica em o usuário utilizar um sistema ou serviço sem saber que determinados passos ou uma série de passos estão ocorrendo enquanto ele realiza um acesso ou executa uma ação.
9. **Escalável:** A escalabilidade é a capacidade que um sistema possui de se adaptar ao contexto, atendendo a uma demanda de acordo com a necessidade.
10. **Mecanismo de alerta:** Capacidade de informar administradores ou elementos da rede da existência de uma ameaça que foi detectada e que se deve tomar cuidado e/ou uma providência.
11. **Aprendizagem de regras:** Conforme o sistema é utilizado por múltiplos usuários, dados de uso são obtidos e novas técnicas de ataque ou elementos, IPS, ou *links* maliciosos são descobertos e alimentam uma base de dados que são replicados para sistemas similares atualizarem suas políticas e/ou os dados são utilizados para compreensão de novos ataques.
12. **Análise de tráfego em tempo real:** A análise de tráfego em tempo real permite verificar o fluxo de dados na comunicação de elementos na rede com entes externos. Dessa forma é possível identificar um comportamento malicioso durante o acesso e a partir daí fazer uma tomada de decisão antes de um comando ser executado no sistema local, por exemplo.
13. **Isolamento entre elementos:** O isolamento entre elementos é uma capacidade que o sistema tem de colocar os elementos conectados em redes distintas para que um elemento não consiga encontrar e se comunicar com outro, impossibilitando que um malware se propague na rede, por exemplo.
14. **Controle de acesso:** O controle de acesso além de limitar o acesso somente a usuários autorizados, também é responsável por identificar ataques de força bruta com dicionário e bloqueio por IP.

4 HONEYPOT UTILIZADO PARA OBTENÇÃO DE DADOS DE ATAQUES LOCAIS

Neste capítulo é apresentado o processo de instalação de configuração de um *honeypot* utilizado para obtenção de dados de ataques locais.

Os dados obtidos a partir do experimento realizado com este *honeypot* teve como objetivo avaliar a incidência de ataques direcionados a portas de serviços comumente utilizados por elementos IoT na região de execução do mesmo. Para isso foi utilizado um *honeypot* de baixa interatividade que ficou responsável por capturar tentativas de acesso a um IP externo exposto na internet e servido à uma máquina para este fim. O experimento esteve ativo por um período de 11 meses coletando IPs de acesso bem como as portas sondadas. Os resultados obtidos foram tratados eliminando ruídos da rede interna, agrupando tentativas de acesso por IP únicos e número de acessos a portas específicas. Desde seu início foram registradas 1.351.789 tentativas de acesso oriundas de 123.008 IPs distintos de 120 países ao redor do mundo e direcionados a diversas portas do sistema.

4.1 Honeyd

O Honeyd é um *framework* que permite a criação de múltiplos *honeypots* virtuais em um computador hospedeiro conectado a uma rede de computadores. Esses *honeypots* virtuais podem ser configurados para simular vários tipos diferentes de sistemas e serviços, permitindo que o usuário execute um número muito grande ¹ de configurações de rede de computadores.

No entanto, na realização deste experimento não foram expostos serviços simulados na rede, se valendo da ferramenta apenas para analisar as características de acesso malicioso a nível de rede e a nível de transporte.

¹Limitado apenas pelos recursos de hardware do sistema.

4.2 Montagem do ambiente

Para montagem do ambiente foi necessário um computador dedicado rodando 24hs por dia possuindo 2 IPs distintos exclusivos para o experimento. Na máquina *host* foi criada uma máquina virtual a fim de aumentar a segurança por isolamento e para separar os fluxos de rede da máquina física que foi utilizada como sistema de produção para acesso e gerenciamento externo.

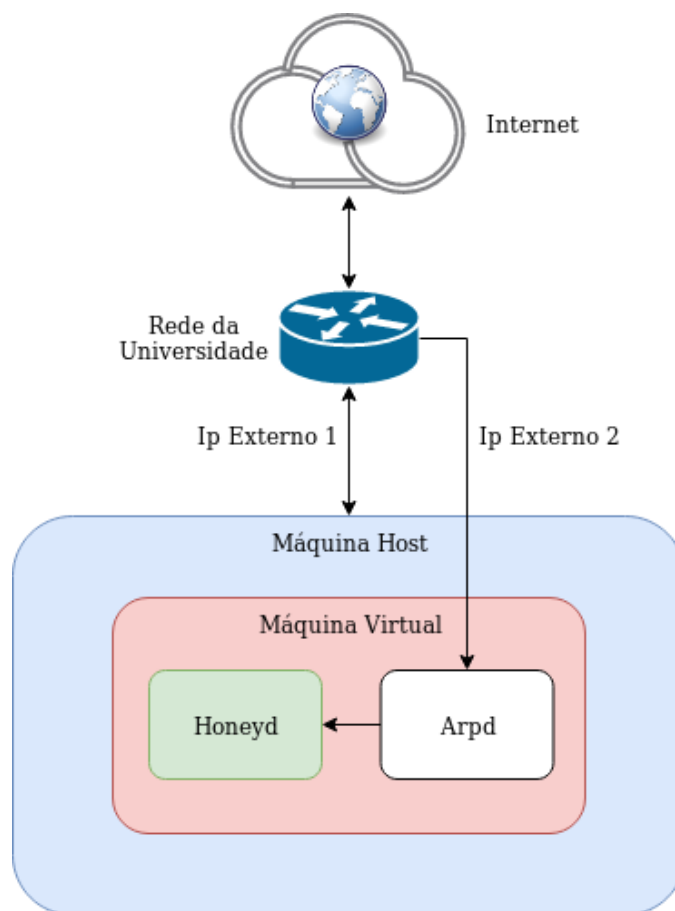


Figura 29: Esquemático do ambiente utilizado no *honeypot*.

A Figura 29 apresenta um esquemático do funcionamento do sistema. A máquina *host* é um sistema de produção e por isso possui seu próprio IP. Nela foi configurada uma máquina virtual para isolar do sistema principal os processos do *honeypot* bem como suas conexões, utilizando um sistema operacional *guest* completo capaz de rodar o Honeyd e os processos necessários para seu funcionamento. A máquina *guest* recebeu um IP distinto da máquina *host* através de um *switch*, responsável por direcionar o tráfego proveniente da internet e distribuído através da rede interna da Universidade Federal do Rio Grande (FURG). Essa organização garante que todos os acessos recebidos na máquina virtual serão potencialmente maliciosos visto que, como na definição de um *honeypot*, qualquer tentativa de acesso a um sistema que não é de produção, como é o caso da

Momento de Acesso	Protocolo	IP de Origem	Porta de Origem	IP de Destino	Porta de Destino	Tam. Pacote	Flag	ID de Ataque
2018-03-02-20:24:51.3605	udp(17)	134.119.213.127	5473	192.168.0.106	5060:	434		
2018-03-02-20:26:50.2325	tcp(6)	104.236.178.199	49135	192.168.0.106	8888:	40	S	
2018-03-02-20:28:25.6124	tcp(6)	77.72.82.92	47430	192.168.0.106	11758:	40	S	
2018-03-02-20:28:29.0684	tcp(6)	191.205.1.65	22361	192.168.0.106	23:	40	S	
2018-03-02-20:28:40.6243	tcp(6)	49.83.238.187	44255	192.168.0.106	22:	60	S	
2018-03-02-20:29:21.7365	tcp(6)	80.82.70.133	49727	192.168.0.106	1500:	40	S	
2018-03-02-20:31:23.3484	tcp(6)	14.18.243.247	51052	192.168.0.106	5901:	52	S	[Windows 2000 RFC1323]
2018-03-02-20:31:24.2485	tcp(6)	14.18.243.247	51052	192.168.0.106	5901:	52	S	[Windows 2000 RFC1323]
2018-03-02-20:31:25.1486	tcp(6)	14.18.243.247	51052	192.168.0.106	5901:	48	S	[Windows XP SP1]
2018-03-02-20:32:56.9605	tcp(6)	60.168.146.72	29066	192.168.0.106	2323:	40	S	
2018-03-02-20:33:53.0205	tcp(6)	185.143.223.201	40606	192.168.0.106	11789:	40	S	
2018-03-02-20:34:38.0205	tcp(6)	191.101.167.235	48358	192.168.0.106	8545:	40	S	
2018-03-02-20:34:45.4726	tcp(6)	118.81.195.15	49132	192.168.0.106	23:	40	S	
2018-03-02-20:35:38.5003	tcp(6)	88.249.91.162	24759	192.168.0.106	23:	40	S	
2018-03-02-20:35:48.1842	udp(17)	71.6.146.186	33457	192.168.0.106	27016:	53		
2018-03-02-20:36:33.6205	tcp(6)	189.18.4.229	52788	192.168.0.106	23:	40	S	

Tabela 4: Dados Coletados no Honeyd

máquina virtual, o mesmo será considerado malicioso. O ARPD é um processo *daemon*² que ouve solicitações e respostas do protocolo ARP usado para resolução de endereços, para endereços IP não alocados. Usando o ARPD em conjunto com o Honeyd, é possível preencher o espaço de endereço não alocado em uma rede de produção com *honeypots* virtuais. Esse processo é importante para o funcionamento do modelo pois, considerando que o Honeyd funciona como uma máquina virtual com um IP local na mesma faixa de IPs da rede local com NAT³ virtualizada da máquina *guest*, se faz necessário o uso do processo ARPD para responder as requisições ARP da rede informando que aquela máquina pode receber os pacotes direcionados ao IP configurado no Honeyd. Para a garantia do funcionamento do sistema em tempo integral foram implementados alguns *scripts* tanto na máquina *host* quanto na máquina *guest*. Desta forma sempre que houve um problema na máquina física, como uma queda de luz ou um desligamento acidental, assim que a energia foi recuperada a máquina se ligou automaticamente. Um *script* de início automático é inicializado junto com o sistema operacional da máquina *host*, sendo ele responsável por executar a máquina virtual. A máquina virtual ao inicializar seu sistema executa ao iniciar, um *script* responsável por chamar o processo ARPD e posteriormente o Honeyd devidamente configurados.

4.3 Implementação de um sistema para coleta e análise de dados de ataques

Durante a execução do experimento foram registradas 1.351.789 tentativas de acesso oriundas de 123.008 IPs distintos. Cada acesso direcionado ao Honeyd contém informações relacionadas a cada acesso por parte do atacante.

A Tabela 4 apresenta 16 instâncias de ataques recebidos, extraídas aleatoriamente para fins de ilustração de um log gerado pelo Honeyd. Como podemos ver o log é dividido em 9 campos sendo eles: Momento do acesso, protocolo utilizado para o acesso, IP de origem,

²Daemon é um programa de computador que executa como um processo em plano de fundo

³Rede NAT: Network Address Translation (NAT) é responsável por mapear IPs privados para um endereço IP exposto publicamente.

porta de origem, IP do *honeypot*, porta a que o ataque foi direcionado, tamanho do pacote, flag de início ou fim de conexão e identificação de sistema baseado em fingerprint.

4.4 Tratamento dos dados

Após o período de coleta de dados um grande volume de logs foi gerado. Para extrair informação útil os dados foram inicialmente tratados com comandos em shell script para bash. Nesse tratamento inicial buscou-se criar arquivos livres de ruídos, gerados por acessos de teste e pacotes provenientes da rede interna da universidade. Procurou-se organizar os dados em diferentes arquivos para facilitar as etapas seguintes, como por exemplo: arquivos contendo apenas IPs únicos de acesso, arquivos contendo o número de repetições de acesso de cada IP, arquivos contendo número de acessos baseado em data, etc. Com os dados previamente tratados foi necessário adquirir informações sobre os IPs obtidos. Para isso, conforme podemos ver na Figura 30, foi necessário implementar um cliente web em python para comunicar com uma API disponibilizada por um serviço web (ipdata.co) especializado em geolocalização e dados de IPs globais, além de um banco de dados para o armazenamento dos dados recebidos.

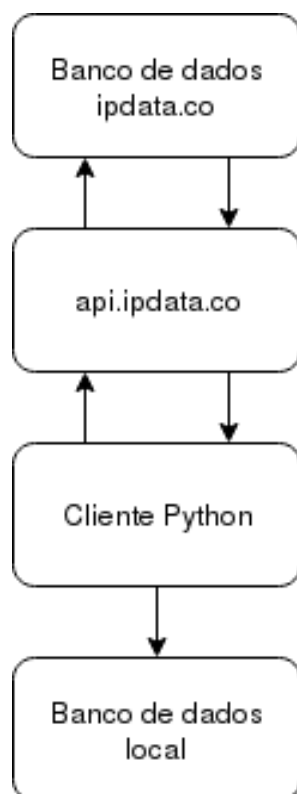


Figura 30: Esquemático comunicação cliente-servidor para identificação de IPs.

A API disponibilizada pelo ipdata.co possibilita inúmeras requisições que são feitas enviando sequencialmente IPs ao qual se deseja obter informação e gera um retorno específico para essa requisição. As informações retornadas especificam detalhes relaci-

ip	104236178199
is_eu	FALSE
city	San Francisco
region	California
region_code	CA
country_name	United States
country_code	US
continent_name	North America
continent_code	NA
latitude	37.7353
longitude	-122.3732
asn	AS14061
organisation	DigitalOcean, LLC
languages	
name	English
native	English
time_zone	
name	America/Los_Angeles
abbr	PST
offset	-800
is_dst	FALSE
current_time	2018-12-22T21:35:41.592452-08:00
threat	
is_tor	FALSE
is_proxy	FALSE
is_anonymous	FALSE
is_known_attacker	FALSE
is_known_abuser	FALSE
is_threat	FALSE
is_bogon	FALSE

Tabela 5: Dados de retorno de verificação de IP na API

onados ao IP que é solicitado. A Tabela 5 ilustra um retorno de uma requisição de IP enviada a api.data.co.

Conforme podemos ver uma simples requisição nos retorna dados de localização contendo país de origem do IP atacante, cidade, latitude e longitude, o nome e número referente a organização proprietária dos endereços IP, a língua nativa do local de origem, informações de fuso horário e finalmente informações de perfil de ameaça - extraídas de um banco de dados de 600 milhões de endereços IP maliciosos de listas autoritativas e ativamente mantidas - que determina se o IP de origem é um nodo TOR⁴ ou utiliza *proxy*, se é um atacante conhecido e previamente denunciado e se o endereço utilizado é ou não um endereço não atribuído ou não endereçável. Com os dados devidamente tratados foi possível analisá-los e plotá-los conforme veremos a seguir.

⁴The Onion Router (TOR) é uma rede de anonimato que redireciona o tráfego de internet através de uma rede de servidores voluntários, distribuídos pelo mundo. Essa abordagem dificulta muito a possibilidade de rastrear o acesso de origem a determinado recurso na internet.

4.5 Resultados obtidos

Em um período de 11 meses de execução do experimento foram coletados informações provenientes de 1.351.789 ataques direcionados a um único IP utilizado. A Figura 31 apresenta uma série temporal onde podemos ver a distribuição destes ataques durante os meses de sua execução.

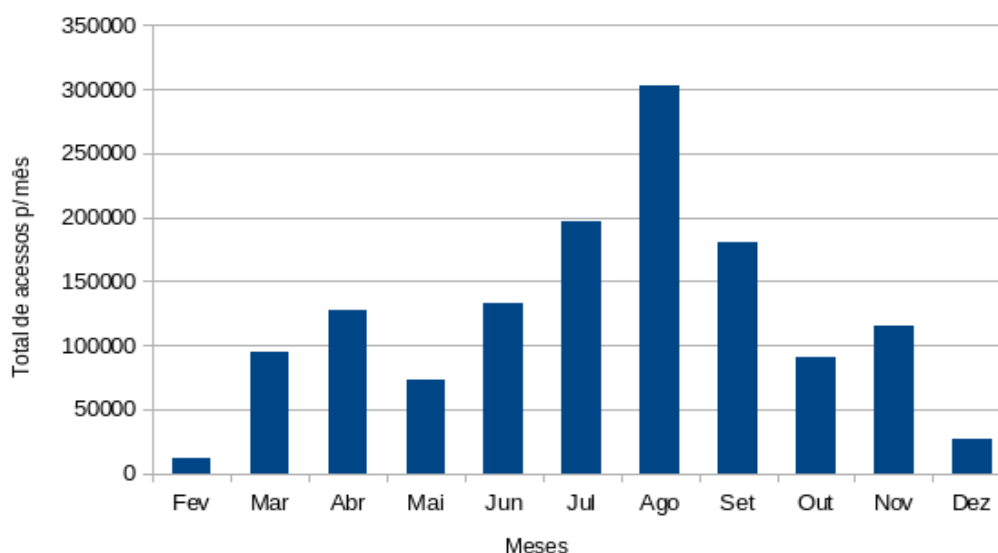


Figura 31: Ataques distribuídos ao longo dos meses do ano de 2018

No entanto, devido a problemas como quedas de luz, falta de internet e desligamentos acidentais do sistema foi necessário fazer uma média da quantidade de ataques de cada mês pela quantidade de dias que o sistema esteve ativo naquele determinado mês. Desta forma, conforme podemos ver na Figura 32, obteve-se um gráfico mais próximo da realidade reduzindo as discrepâncias por falhas do sistema.

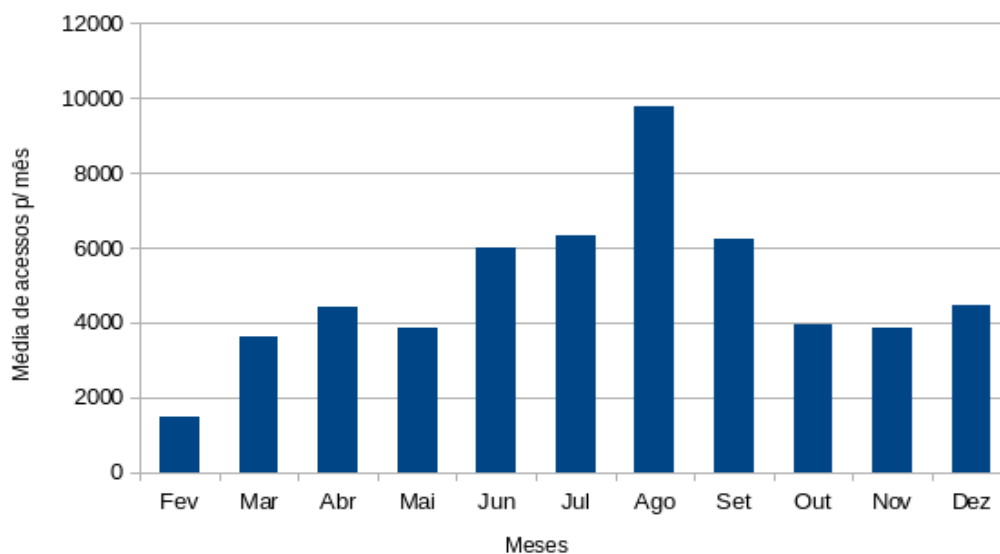


Figura 32: Ataques distribuídos ao longo dos meses do ano de 2018 (Corrigido)

Na sequência, como apresentado na Figura 33, foi analisado a distribuição dos ataques durante os dias do ano, gerando um gráfico temporal mais detalhado da variação da quantidade de ataques no decorrer do tempo.

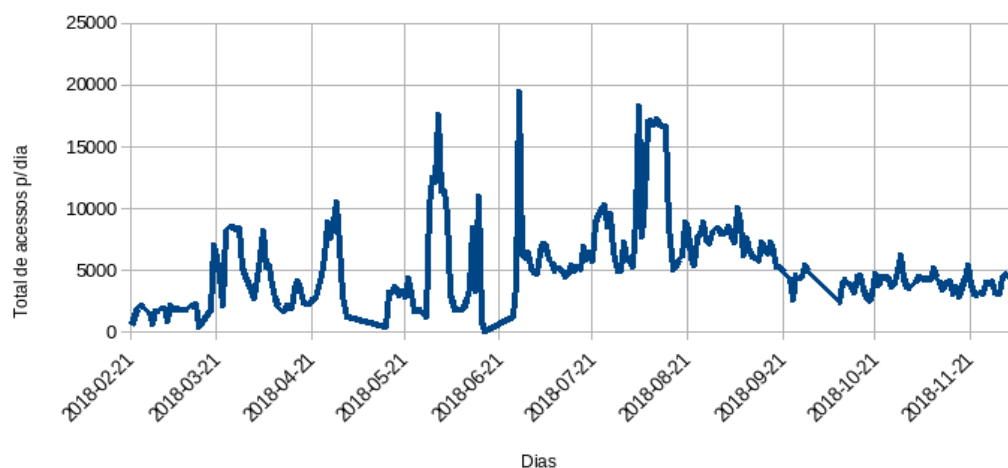


Figura 33: Variação de incidência de ataques durante os dias de execução

A partir dessa análise foi possível gerar um gráfico da distribuição dos ataques pelos dias da semana, conforme vemos na Figura 34.

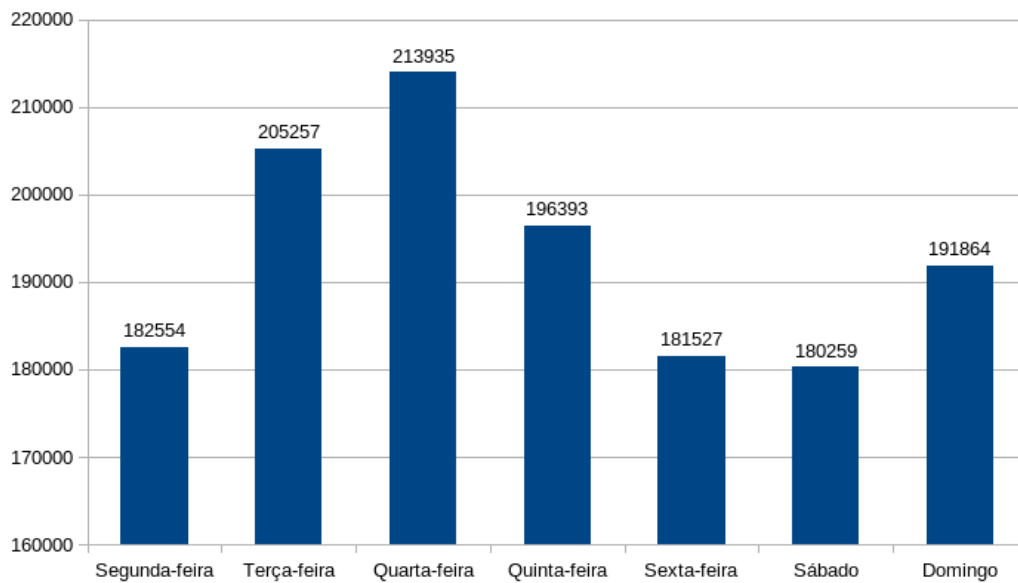


Figura 34: Distribuição de ataques pelos dias da semana

A Figura 35 apresenta a quantidade de cada tipo de pacote recebidos durante a execução do experimento. Conforme podemos ver a grande maioria dos pacotes recebidos foram do tipo TCP com poucos pacotes UDP e pouquíssimos pacotes ICMP.

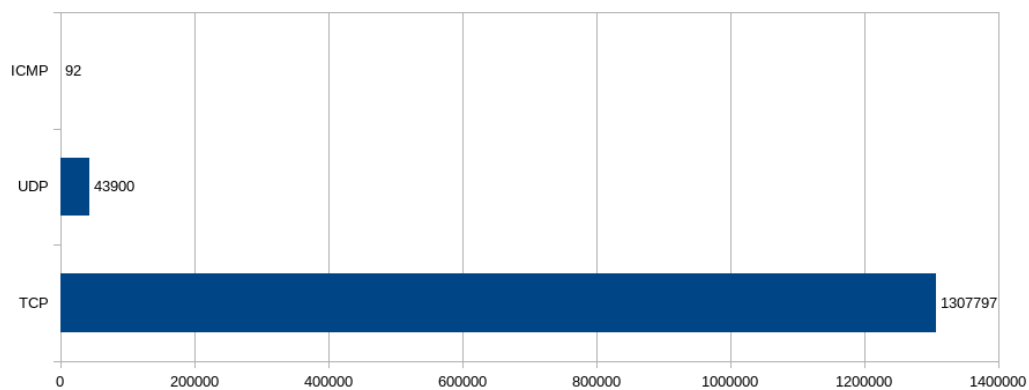


Figura 35: Incidência de tipos de pacote.

Esses pacotes capturados foram direcionados a diversas portas do sistema. Na Figura 36 podemos ver as 15 principais portas atacadas e destacando-se com mais de 100.000 acessos a porta 23, destinada ao serviço Telnet de acesso remoto, foi a porta que mais recebeu ataques.

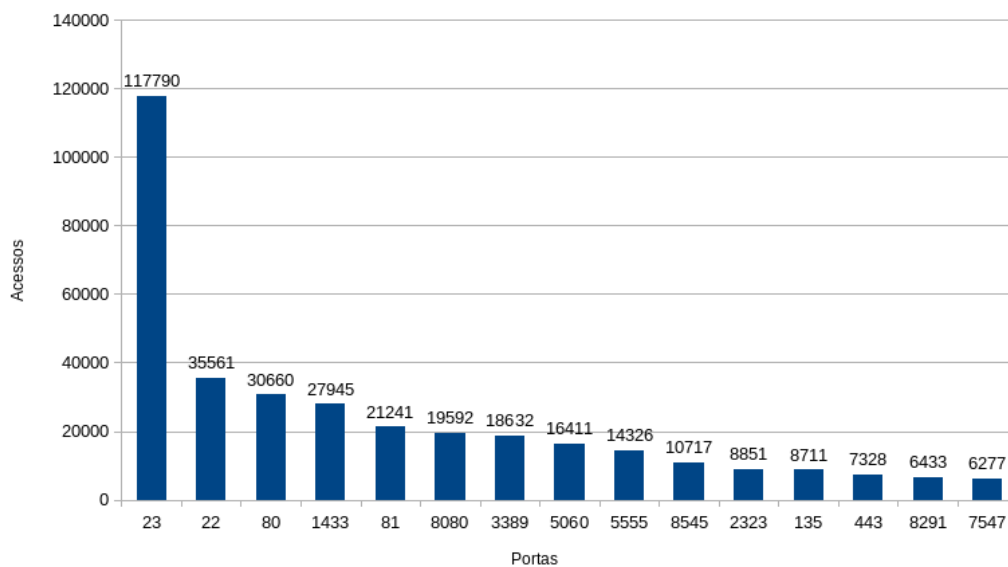


Figura 36: As 15 principais portas atacadas

Esses acessos após terem tido suas localizações identificadas através de seus IPs foi possível montar um mapa da origem dos ataques, como vemos na Figura 37. Como podemos ver, os ataques partiram dos mais diversos lugares do mundo contabilizando um total de 120 países incluindo o Brasil.



Figura 37: Posição geográfica de atacantes

4.6 Conclusão

Baseado nos resultados obtidos pode-se confirmar a grande incidência de ataques direcionados ao serviços Telnet (porta 23) e SSH(porta 22) bem como o tipo de pacote

TCP, predominante em sua conexão. Apesar de observarmos uma incidência de ataques maior em alguns meses e dias, os resultados nos mostram que ataques acontecem constantemente e em grande quantidade. Também podemos ver que os ataques partiram de diversas partes do mundo indicando que as ameaças podem partir de qualquer região e bloquear faixas de IP de países específicos não evitaria novos ataques. Considerando esses resultados na sequência foi proposto um sistema que com o intuito de mitigar esses problemas.

5 IOT TUNNEL

No Capítulo 4 foi concluído que os ataques direcionados a dispositivos IoT são frequentes, mesmo no local utilizado para os testes, a rede da universidade. Problemas relacionados a segurança da informação envolvendo elementos IoT, debatidos tanto em noticiários quanto estudos acadêmicos e de especialistas técnicos além de os resultados experimentais obtidos e analisados no experimento aqui apresentado previamente, despertaram a motivação de propor um sistema para mitigar a exploração desses elementos. Para isso, inicialmente avaliou-se relatórios de ataques direcionados a elementos IoT que visavam obter controle destes elementos e os meios empregados; confrontando os mesmos com os resultados obtidos. Desta forma verificou-se que uma das maiores fontes de exploração envolvem ataques direcionados a serviços de acesso remoto nesses equipamentos, mais especificamente os protocolos Telnet e SSH. A exploração a esses protocolos abrangem desde tentativas de acesso utilizando senhas padrões a exploração de *bugs* conhecidos em versões desatualizadas dos serviços.

Levando-se em consideração os problemas citados, o sistema aqui proposto visa criar uma camada extra de segurança na comunicação de elementos IoT que se comunicam com a internet ou que, por configuração padrão e/ou pouco conhecimento do usuário final, estabeleça essa conexão. Para garantir uma comunicação segura entre o usuário externo e os elementos, o sistema deverá garantir sempre uma comunicação criptografada independente de o elemento utilizar um serviço de acesso remoto que não possua essa capacidade. Além disso o sistema deve ser capaz de identificar o uso malicioso de elementos que estejam conectados através dele analisando o fluxo de dados identificando duas situações: (1) Elementos apresentando comportamento suspeito de forma autônoma (sem conexão externa ao elemento) e (2) Acesso externo enviando comandos suspeitos. Na situação (1) o sistema ao identificar comportamento suspeito em um elemento da rede que não está recebendo acesso deverá isolar esse elemento da rede, impossibilitando conexão com outros elementos e com a internet. Na situação (2) se ao analisar o tráfego de um acesso remoto a um elemento conectado for identificado comportamento suspeito, a conexão a esse elemento deverá ser roteada a um *honeypot* virtual no sistema para captura de comandos, binários e/ou *scripts* do atacante. Por fim, o sistema deve se comunicar

com um servidor externo dedicado para adquirir informações de novas ameaças e atualizar suas diretrizes de segurança no sistema local além de, enviar para esse servidor informações adquiridas de ataques recebidos com intuito de alimentar a base de dados com informações estatísticas e possivelmente ameaças e formas de ataques inéditas.

5.1 Arquitetura

Para cumprir seu objetivo, o sistema aqui proposto deve funcionar como um intermediário na comunicação dos elementos IoT, a que se desejam proteger, a rede local e a internet. A Figura 38 apresenta o esquemático de seu funcionamento.

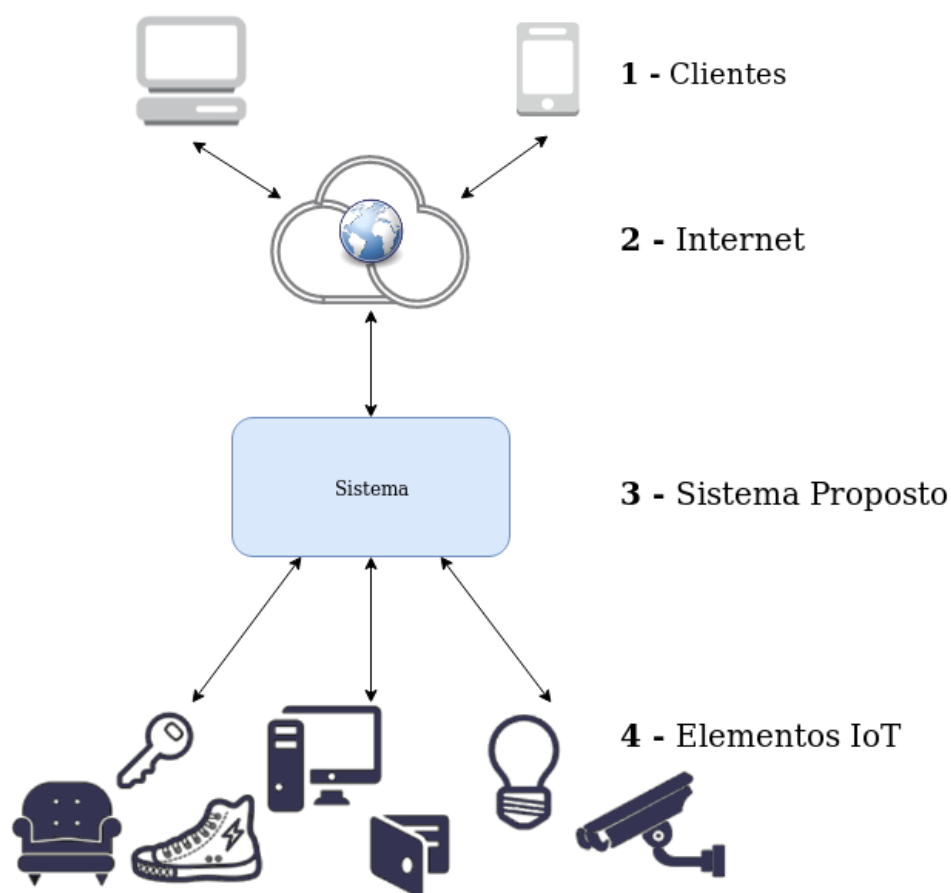


Figura 38: Visão geral do sistema proposto

A linha 1 representa as aplicações cliente que são os interessados em acessar os serviços, ou se comunicar de alguma forma com os elementos IoT através da internet. A linha 2 representa a internet, canal de comunicação no qual os elementos se comunicam com os clientes externos. Na linha 3 temos o sistema proposto intermediando a comunicação entre os elementos IoT e os clientes que desejam acesso através da internet. Por último, na linha 4 temos os elementos IoT propriamente ditos, conectados a rede local e a internet através do sistema proposto. Como podemos ver na Figura 38 todos

elementos que desejam se conectar entre si e a internet devem utilizar o sistema proposto como “ponte”. Desta forma o sistema funcionará como um modem/roteador e será o responsável por gerar a rede interna e distribuir e gerir os IPs únicos dos elementos que se conectarem a ele assim como será responsável por conectar na infraestrutura que comunicará com a internet, gerenciando os pacotes de entrada e saída e encaminhando-os aos respectivos destinos. Para realizar seus objetivos o sistema conta, internamente, com 7 módulos principais conforme podemos ver na Figura 39.

Cada um dos módulos contidos no sistema possui um objetivo específico e pertence a um ciclo bem definido no seu funcionamento.

Controle de acesso: é o responsável por um controle inicial da tentativa de acesso por parte de um cliente externo. Ele funciona como *firewall* analisando as tentativas de acesso e pacotes direcionados a elementos IoT integrados ao sistema para avaliar se este acesso é permitido e pode seguir o fluxo. Além disso o módulo tem como objetivo controlar o acesso bloqueando tentativas de força bruta e IPs que sejam identificados como maliciosos pelo analisador de tráfego ou pelos dados do servidor, criando uma *blacklist* para esses IPs, impedindo a comunicação com elementos da rede.

Gerenciador de login: tem como objetivo encaminhar as tentativas de acesso por parte de um cliente ao elemento IoT de destino. Uma das funções mais importantes do gerenciador de login é mimetizar os serviços de acesso remoto do elemento IoT interceptando o acesso que seria direto ao elemento e adicionando uma camada transparente ao cliente que acessa. Desta forma o gerenciador de login cria uma camada extra de criptografia aumentando a segurança da comunicação entre ambas as pontas.

Analisador de tráfego: analisa todo o tráfego gerado entre os clientes e os elementos IoT e entre elementos IoT e servidores da internet. Ele funciona como um IPS e é o responsável por detectar qualquer comportamento suspeito tanto por parte dos clientes e comandos executados em tempo real em acessos a elementos da rede quanto por comportamento autônomo nos elementos IoT. Ao detectar uma irregularidade baseado em suas regras, esse módulo deverá tomar decisão para cada um dos casos: (1) Se um cliente executa comandos suspeitos em um determinado elemento esta conexão específica deverá ser roteada para um *honeypot* correspondente e o banco de dados deve ser atualizado com informações desse cliente. (2) Se um elemento IoT está apresentando comportamento suspeito de forma autônoma este elemento deve ter sua comunicação com a internet e com outros elementos interrompida e o usuário informado.

Interface de rede virtual: é responsável por gerenciar a comunicação dos elementos IoT com a internet e entre si. É nesse módulo que a rede local é gerada e um SSID disponibilizado permitindo o acesso wireless ao sistema e a disponibilização de um IP para cada elemento. O sistema tem capacidade de gerar diversas interfaces virtuais podendo isolar os elementos entre si caso não haja necessidade de comunicação por rede local, evitando dessa forma propagação de malware entre os elementos.

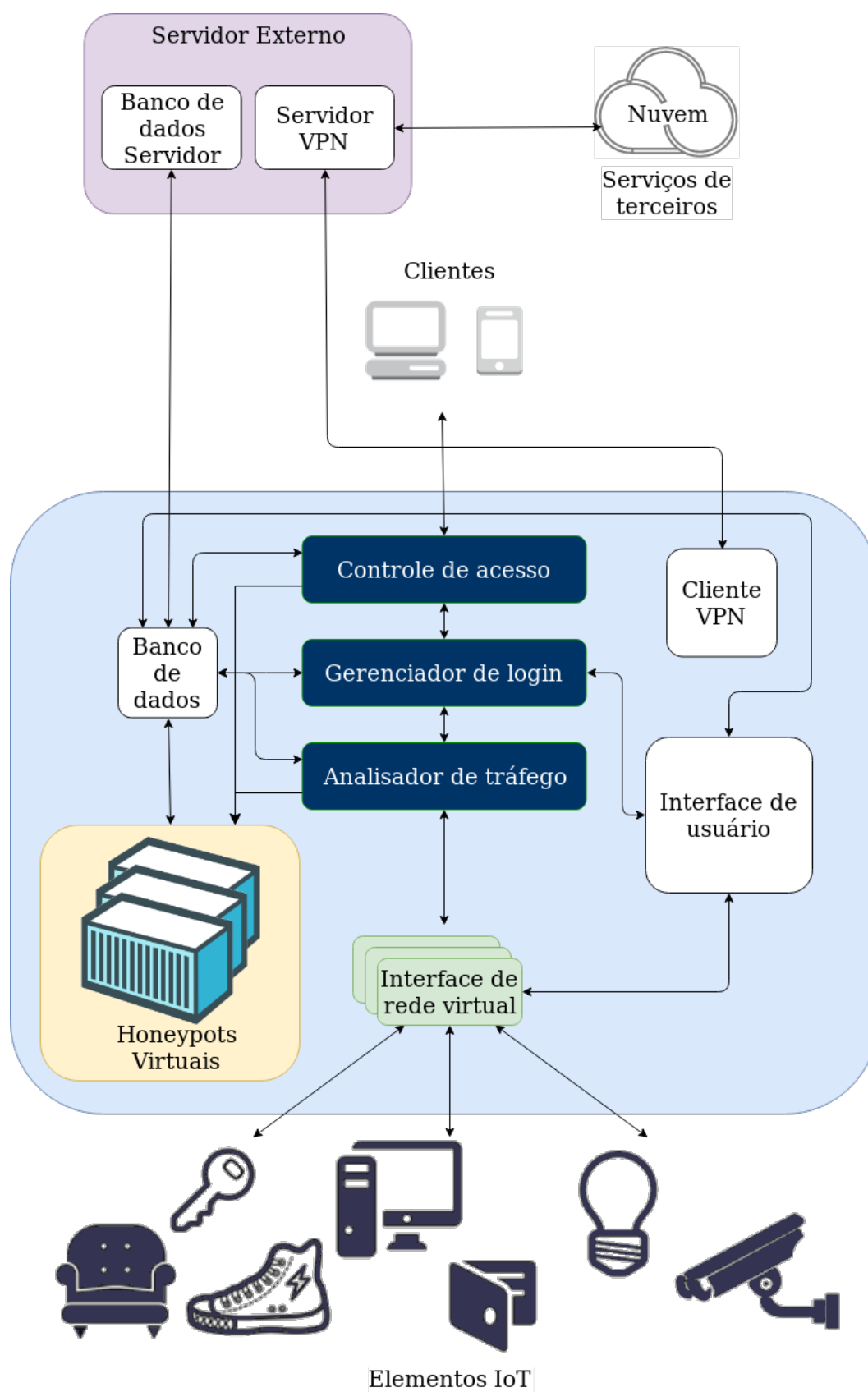


Figura 39: Módulos do sistema proposto

Interface do usuário: é através desse módulo que o usuário pode interagir com o sistema, personalizar configurações, gerenciar os elementos IoT e obter informações. Por aqui o usuário pode verificar as rotas de acesso externo aos respectivos elementos bem como suas senhas.

Honeypots virtuais: esse é o módulo responsável por adquirir informações sobre os atacantes e os métodos e ferramentas utilizados. Ao detectar um comportamento suspeito por parte de um cliente, o módulo Analisador de tráfego redireciona a conexão desse cliente a esse módulo que ao receber uma requisição de acesso aloca um *honeypot* para receber essa conexão. Os *honeypots* utilizados aqui possuem imagens já pré-configuradas e são ativadas pouco antes do roteamento de conexão. O objetivo principal desses *honeypots* é simular um elemento IoT real e fornecer um ambiente para que o atacante possa realizar um ataque real, alterando configurações do sistema, instalando ferramentas e executando scripts. Todas alterações e ferramentas utilizadas são capturados pelo módulo e enviados ao banco de dados local. Ao detectar um comportamento considerado crítico - como uma tentativa de ataque a uma rede externa - a conexão deverá ser finalizada e o IP do atacante banido no módulo de Controle de acesso. Ao final de um acesso, o *honeypot* coleta as informações do ataque em forma de logs do sistema além de binários que o atacante venha a ter utilizado e os envia ao módulo de banco de dados.

Banco de dados: esse é o módulo responsável por armazenar todas informações do sistema. Aqui são armazenadas preferências do usuário, regras de acesso, lista de usuários, dados dos *honeypots*, etc. Esse módulo também é responsável por atualizar os outros módulos com informações mais recentes obtidas no servidor externo do projeto bem como enviar relatórios de dados obtidos nos *honeypots* locais ao servidor.

Cliente VPN: elementos IoT frequentemente necessitam se comunicar com serviços de terceiros para seu pleno funcionamento. Um grande problema é que essa comunicação muitas vezes utilizam protocolos pouco seguros ou sem nenhum mecanismo de segurança expondo toda informação trocada entre o elemento e a nuvem. E ainda que essa comunicação seja segura, possivelmente não seja do interesse do usuário fornecer informações sobre sua localização e serviço de internet através de seu IP para terceiros. Deste forma o módulo cliente VPN atua protegendo a conexão do elemento IoT encapsulando sua comunicação com protocolo seguro, criptografando as trocas de informação de ponta-a-ponta e mascarando a localização para o destino final visto que o servidor VPN externo é responsável pela conexão com serviços de terceiros.

5.2 Aplicações do modelo proposto

O modelo aqui proposto tem por objetivo mitigar problemas de segurança da informação em elementos IoT. Para isso avaliou-se os principais problemas de segurança em IoT em 2018 listados pela OWASP:

1. **Senhas fracas ou hardcoded:** Uso de credenciais facilmente suscetíveis a força bruta, publicamente disponíveis ou imutáveis.

Elementos IoT expostos na internet contendo senhas padrão ou de fácil dedução e, no pior dos casos, com impossibilidade de alteração torna esses sistemas alvos fáceis de exploração. Para superar essa vulnerabilidade o sistema proposto age intermediando os serviços de acesso remoto. A Figura 40 apresenta um exemplo de elemento IoT contendo um serviço com senha padrão e como o sistema proposto atua como um intermediário na comunicação com o cliente. O serviço exposto aos clientes é gerado pelo sistema proposto contendo uma senha personalizada pelo usuário.

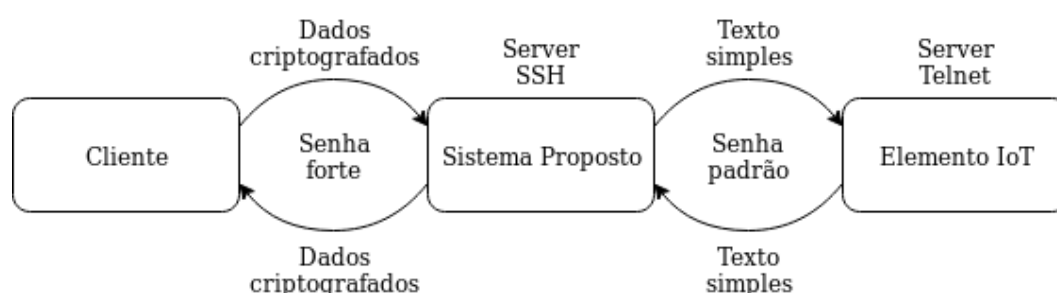


Figura 40: Sistema intermediando comunicação de um elemento IoT com uma aplicação cliente.

2. **Serviços de rede inseguros:** Serviços de rede desnecessários ou inseguros executados no próprio dispositivo, especialmente aqueles expostos à Internet, que comprometem a confidencialidade, integridade / autenticidade ou disponibilidade de informações ou permitem o controle remoto não autorizado.

Alguns elementos IoT incluem serviços desnecessários e/ou vulneráveis e muitas vezes difíceis ou impossíveis de desabilitar. O sistema proposto bloqueia a descoberta desses serviços e, se necessário, expõe um serviço intermediário ao do elemento, replicando o serviço em uma versão mais recente ou um serviço similar seguro, conforme vemos na Figura 41.

3. **Interfaces do ecossistema inseguras:** Serviços web, APIs de back-end, nuvem ou interfaces mobile inseguros no ecossistema fora do dispositivo que permitem o comprometimento do dispositivo ou de seus componentes relacionados. Problemas comuns incluem falta de autenticação/autorização, ausência de ou criptografia fraca e falta de verificação em dados de entrada e saída.

Elementos IoT normalmente utilizam uma interface web fornecendo ao usuário acesso a algumas funções internas e de controle. Essas interfaces muitas vezes apresentam vulnerabilidades que permitem acesso indevido e sniff de sua comunicação devido a falta de criptografia. A Figura 42 apresenta como o sistema pode mitigar

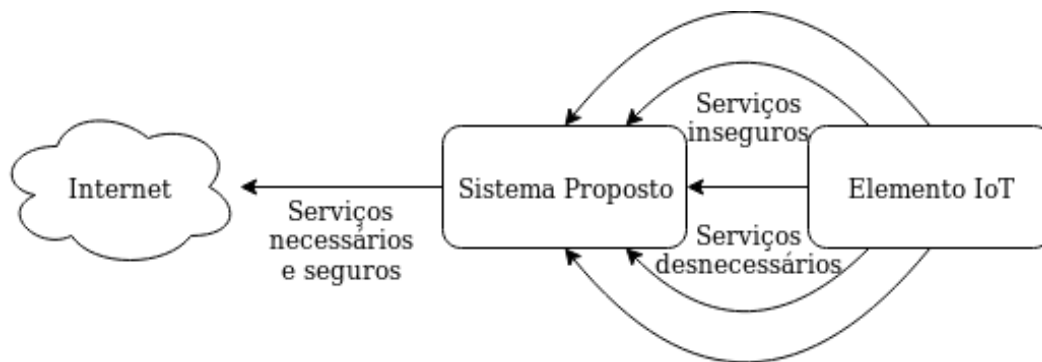


Figura 41: Sistema bloqueando serviços desnecessários e inseguros ao meio externo e expondo um serviço necessário e seguro.

esse problema através do intermédio da comunicação dos serviços web utilizando um serviço próprio que replica as funcionalidades do serviço web do elemento IoT. O serviço replicado utiliza protocolo seguro para comunicação na rede, aprimorando a confidencialidade e integridade da informação trocada.

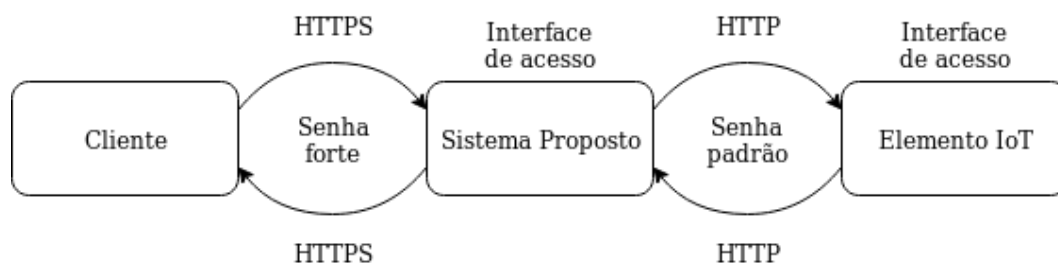


Figura 42: Sistema intermediando comunicação com interface web.

4. **Uso de componentes inseguros ou desatualizados:** Uso de componentes / bibliotecas de software obsoletos ou inseguros que podem permitir que o dispositivo seja comprometido.

Serviços com vulnerabilidades conhecidas são facilmente identificados na rede por ferramentas automatizadas ou por um atacante que deseja efetuar um ataque em um alvo bem definido. Para minimizar as chances de exploração deve-se sempre optar por um software atualizado e com o mínimo de vetores de ataque. Para isso o sistema restringe o acesso a todos serviços inseguros de comunicação aos elementos, fornecendo acesso somente aos serviços necessários e através de comunicação segura. Sendo assim, ele pode bloquear o redirecionamento de portas à serviços inseguros e utilizar o redirecionamento à serviços necessários utilizando seus protocolos seguros.

5. **Transferência e armazenamento de dados inseguros:** Falta de criptografia ou controle de acesso a dados confidenciais em qualquer parte do ecossistema, incluindo dados inativos, em trânsito ou durante o processamento.

Com a capacidade de intermediar os serviços de acesso dos elementos, o sistema proposto viabiliza uma comunicação segura utilizando protocolos criptografados com controle de acesso a usuários cadastrados. Para conexões que partem dos elementos para acesso a serviços em nuvem o sistema propõe o uso padrão de uma VPN com comunicação criptografada. É possível também bloquear toda comunicação externa proveniente do elemento em questão evitando assim o envio indesejado de informações privadas para serviços de terceiros. No entanto é sabido que essa abordagem poderia inviabilizar o uso pleno das funcionalidades do elemento em questão cabendo ao usuário avaliar.

6. **Configurações padrão inseguras:** Dispositivos ou sistemas fornecidos com configurações padrão inseguras ou falta a capacidade de tornar o sistema mais seguro, impossibilitando os administradores de modificar as configurações.

Com a capacidade de intermediar os serviços de acesso externo ao elemento, todas a senhas de acesso são personalizáveis através de uma interface web com acesso local aos usuários do sistema.

5.3 Relação de funcionalidades com trabalhos relacionados

Nesta seção foi realizado um comparativo de funcionalidades e características dos trabalhos relacionados apresentados na seção 3.7 reapresentando a Tabela 3 e anexando características presentes no IoT Tunnel através da nova Tabela 6.

	IoTPOT	SIPHON	honeyTARG	HioTPOT	I.D.E.A.S.P.N.P.S.	Securebox	IoT Tunnel
Captura de tentativas de acesso	x	x	x	x			x
Deteção de intrusão	x	x		x	x	x	x
Captura de comportamento	x	x		x			x
Múltiplas arquiteturas	x	x					
Ambiente virtualizado	x		x	x		x	x
Distração de atacantes				x			x
Múltiplas frentes de ataque externas		x	x				
Transparente ao usuário	x	x		x		x	x
Escalável		x		x		x	x
Mecanismo de alerta			x	x	x	x	x
Aprendizagem de regras				x		x	x
Controle de acesso				x		x	x
Análise de tráfego em tempo real						x ¹	x
Isolamento entre elementos						x	x
Realocação transparente de conexão a um honeypot							x
Tunelamento de protocolos inseguros							x
Bloqueio de portas desnecessárias							x
Tunelamento de acessos externos							x

Tabela 6: Comparativo de funcionalidades: trabalhos relacionados e IoT Tunnel

1. **Realocação transparente de conexão a um *honeypot*:** A realocação de conexão permite que ao identificar um determinado comportamento malicioso, durante o trânsito de dados na rede, o sistema realoque a conexão estabelecida com um elemento para um *honeypot* disponível de forma transparente ao atacante.

¹ Somente tráfego que parte dos elementos para o meio externo

2. **Tunelamento de protocolos inseguros:** A utilização de uma camada extra de segurança utilizando criptografia sobre um protocolo inseguro garante integridade e confidencialidades nas informações trocadas sobre este protocolo tunelando-as através de um protocolo seguro.
3. **Bloqueio de portas desnecessárias:** Através do encapsulamento dos elementos conectados à rede e com o uso de redirecionamento de portas o sistema consegue eliminar a exposição de portas de serviços que rodam e são acessíveis externamente, por padrão. Dessa forma é possível reduzir os vetores de ataque limitando a exposição de portas que se julgam essenciais para o funcionamento do sistema.
4. **Tunelamento de acessos externos:** O uso de VPN para acessos externos aplica uma camada de criptografia de ponto-a-ponto entre os elementos que necessitam acessar servidores externos e não utilizam protocolos seguros garantindo que comunicação entre o VPN e os elementos não estarão sujeitos a espionagem e alteração de informação além de garantir a privacidade.

5.4 Estudo de caso

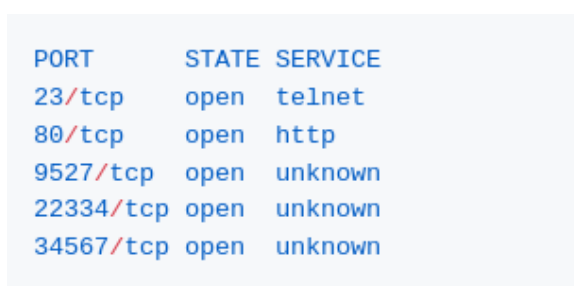
Para um estudo de caso foi avaliado um elemento IoT que apresenta uma vulnerabilidade conhecida no seu serviço de acesso remoto Telnet. O objeto de estudo se baseia em um falha de segurança descoberta em uma família de câmeras IP chinesas do tipo Wireless Ip Camera 360. Nos CVE's (MITRE, 2018a), (MITRE, 2018b), (MITRE, 2018c), (MITRE, 2018d) foram reportados diversos problemas de segurança e serão debatidos a seguir.



Figura 43: Wireless IP Camera 360. Fonte: (GARCIA, 2018)

A Wireless IP Camera 360 (Figura 43) é uma câmera fabricada na China que permite a transmissão de vídeo e áudio por meio de um aplicativo próprio e também oferece um sistema de alerta. Este é um tipo de sistema relativamente popular, uma busca simples no Shodan ² lista mais de 100 mil câmeras deste tipo vulneráveis conectadas à Internet.

A câmera usa uma configuração de hardware semelhante à de outros modelos de câmeras IP chinesas, contendo um processador ARM integrado e sem fio. O software disponível nesta câmera é composto por uma compilação personalizada do Linux em torno do busybox e de binários personalizados.



PORT	STATE	SERVICE
23/tcp	open	telnet
80/tcp	open	http
9527/tcp	open	unknown
22334/tcp	open	unknown
34567/tcp	open	unknown

Figura 44: Escaneamento de portas Wireless IP Camera 360. Fonte: (GARCIA, 2018)

Conforme vemos na Figura 44, 5 portas TCP foram encontradas ativas em um escaneamento de portas. A porta 23 é responsável pela comunicação Telnet com a câmera. Essa conexão fica ativa constantemente havendo pelo menos um usuário válido no sistema, com uma senha criptografada SHA-512. Essa câmera apesar de utilizar um algoritmo eficiente para criptografar sua senha, por possuir uma senha padrão com um hash conhecido pode sofrer um ataque de força bruta, a fim de encontrar a senha original. Também pode funcionar como um backdoor e fornecer acesso remoto ao fabricante ou quem mais possua essa senha.

A porta 9527 é usada para debug remoto. Quando essa porta é acessada via Telnet, a câmera exibe várias informações de debug.

Na Figura 45 vemos que ao realizar o acesso Telnet a porta mostra um prompt, aguardando credenciais de login. As credenciais fazem parte das informações de debug transferidas anteriormente e não podem ser alteradas pelo usuário. Nesse caso, o login é admin e a senha é 123456, ou seja, hash (123456) = nTBCS19C.

Ao realizar o acesso utilizando a senha capturada e executar o comando de ajuda é possível visualizar uma série de comandos, como vemos na Figura 46

Nesta lista podemos ver uma série de comandos que possibilitam a um atacante extrair diversas informações do sistema, desliga-lo ou reinicia-lo, alterar informações e configurações mas de todos os comandos o que mais se destaca é o comando *shell*. Com esse comando é possível obter uma *shell* linux ao sistema interno da câmera com pri-

²Shodan é um sistema de busca muito popular que indexa elementos vulneráveis por todo o mundo e possibilita que usuários busquem por esses elementos utilizando diversos tipos de filtros de busca.

```

user@kali$ telnet 192.168.3.4 9527
Trying 192.168.3.4...
Connected to 192.168.3.4.
Escape character is '^]'.

login: admin
Password:
login(admin, *****, Console, address:)

_____password = nTBCS19C
admin$
Connection closed.
user@kali$

```

Figura 45: Acesso Telnet na porta 9527. Fonte: (GARCIA, 2018)

```

ability Net Ability Utility!
ad AD debug interface!
alarm Alarm status!
autoshtut auto shut the DVR
bitrate Dump BitRate infomation!
cfg Config Help Utility!
encode Encode commands!
fs Fs debug interface!
heap Dump heap status!
help Try help!
log Log utility!
netitf NetInterFace Dump!
netm NetManager Dump!
packet Packet usage!
quit Quit!
reboot Reboot the system!
record Record console utility!
resource CPU usage!
shell Linux shell prompt!
shutdown Shutdown the system!
snap Snap Console Utility!
thread Dump application threads!
time Set SystemTime!
timer Dump application timers!
upgrade Upgrade utility!
user Account Information!
ver version info!

```

Figura 46: Lista de comandos disponíveis. Fonte: (GARCIA, 2018)

vilégio *root*, permitindo controle total das funcionalidades, acesso a todos arquivos e permissão de modificar todo o sistema.

Também na porta 9527 é possível expor a senha de acesso protocolo *Real Time Streaming* Protocol (RTSP) ativo na porta 22334 observando as tentativas de acesso do mesmo através do acesso Telnet. O RTSP é o protocolo do serviço de *streaming* da câmera responsável por compartilhar as imagens obtidas pela câmera. Ao obter acesso RTSP o atacante adquire capacidade visualizar as imagens da câmera em tempo real. Na Figura 47 podemos ver o *token* de acesso RTSP sendo vazado em uma tentativa de conexão *InsertConnect*.

```
user@kali$ telnet 192.168.3.4 9527
Trying 192.168.3.4...
Connected to 192.168.3.4.
Escape character is '^]'.

Transport: New Client ID[0]@[192.168.3.4:49465] Connect___!!!___
======> [ConnectCallback] : 470,ConnectCallback,../../../../
*****
InsertConnect : 69f738
*****
```

Figura 47: Vazamento do *token* de acesso RTSP. Fonte: (GARCIA, 2018)

Por último foi verificado que a câmera possui um serviço de acesso remoto que comunica com a nuvem do projeto, no entanto ao analisar o tráfego da conexão foi possível perceber que a comunicação não utiliza nenhum tipo de criptografia, facilitando muito a interceptação dessa comunicação e possibilitando captura de credenciais, informações de vídeo e do usuário.

5.5 Implementação

Para mitigar os problemas citados no estudo de caso, foi implementado um sistema de testes que emula alguns serviços da *Wireless IP Camera 360* a fim de comprovar sua eficácia em reduzir a efetividade de ataques a esses serviços. Inicialmente foi criada uma máquina virtual contendo uma distribuição Linux que funcionou como base para simular a câmera em questão e seus serviços e outra para simular o sistema proposto e suas funcionalidades.

A Figura 48 apresenta a topologia dos elementos utilizados no experimento demonstrando as conexões realizadas entre eles e como estavam organizados nos testes. Foi utilizado uma máquina virtual responsável por simular o sistema experimental (VM1), outra máquina virtual responsável por simular a câmera (VM2), ambas executando uma distribuição linux personalizada, e por fim uma aplicação cliente responsável por simular um usuário ou atacante. O cliente e o sistema experimental (VM1) comunicam-se entre si através da rede local a qual estão conectados. O sistema experimental (VM1) e a câmera

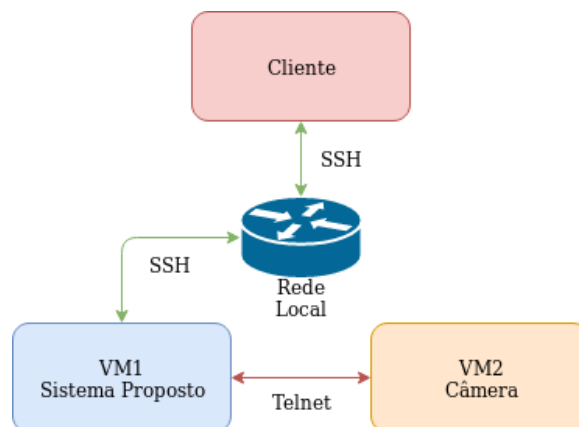


Figura 48: Topologia do experimento.

(VM2) comunicaram-se através de uma rede local gerada pelo sistema, o qual funciona como um *Access Point* permitindo que elementos IoT - nesse caso a câmera - se conectem a ele para que essa conexão seja gerenciada para comunicar os elementos conectados a outras redes e/ou a internet.

Como visto na sessão 5.4 a *Wireless IP Camera 360* possui uma conexão Telnet de *debug* e uma conexão Telnet para acesso remoto, ambas sempre ativas e aguardando conexão. Para simular essas características, na VM2, foi instalado um serviço de Telnet padrão com acesso restrito a um usuário com senha definida aguardando por novas conexões constantemente.

Para interagir com a câmera o cliente teve a disposição um serviço SSH gerado pelo sistema experimental e propagado pela rede local. Ao obter acesso ao serviço SSH a conexão do cliente foi roteada pelo sistema, de forma transparente, para a câmera através do serviço Telnet que esteve disponível para o sistema através da conexão direta que o mesmo obteve com a câmera.

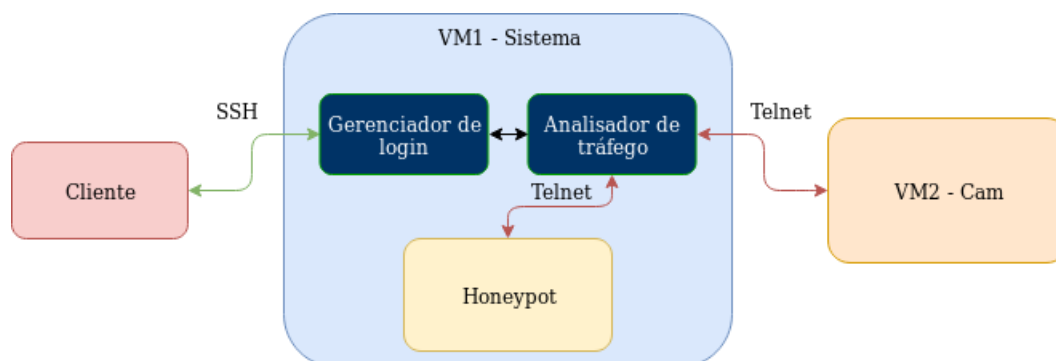


Figura 49: Visão geral do sistema experimental.

Na Figura 49 temos uma visão mais detalhada de como o sistema experimental está organizado internamente e quais módulos estão ativos nesse experimento. Como descrito na sessão 5.1 o módulo **Gerenciador de login** é responsável por gerenciar a tentativas

de acesso provenientes dos clientes. Nesse caso o módulo identifica as novas conexões direcionadas ao serviço SSH do sistema, se a tentativa de acesso obter sucesso de login essa conexão é redirecionada a porta de destino do serviço Telnet em execução na VM2. Para realizar esse redirecionamento de conexão e de forma transparente, o sistema - que roda sobre um kernel linux - sofreu algumas alterações na forma como o interpretador de comandos bash lida com novas conexões de acesso remoto SSH, redirecionando conexão de novos clientes para a VM2 e suas mensagens de *banner* e *login* que foram removidas a fim de tornar imperceptível o redirecionamento.

O **analisador de tráfego** ficou responsável por analisar o comportamento do cliente em relação a câmera, obtendo acesso em tempo real de qualquer interação da aplicação cliente para tomadas de decisão. Para isso utilizou-se o *strace* - uma ferramenta que monitora as chamadas de sistema e os sinais recebidos pela aplicação para diagnóstico e *debug* em tempo de execução - em conjunto com um *shell script* personalizado. O *script shell* foi configurado para identificar uma pequena lista de comandos considerados maliciosos como "wget" e "ping". Quando identificado o uso de um comando dessa lista, o cliente é considerado suspeito e a conexão Telnet com a câmera é finalizada e redirecionada ao *honeypot*.

A implementação do **honeypot** consistiu em gerar um container dentro da VM1 com uma imagem pré-configurada para executar um serviço Telnet. Para isso foi implementado um gerenciador de containers responsável por disponibilizar a imagem gerada e previamente configurada com base na câmera, ao sistema. No experimento em questão, basicamente, o container contava com um SO linux idêntico à VM2 além de estar com seu serviço Telnet ativo, assim como a câmera. O container forneceu um ambiente completo e interativo para o cliente que obteve acesso a ele, possibilitando a execução de comandos e instalação de binários e scripts. O nome de usuário e o *hostname* da VM2 foram preservados a fim de minimizar a chance de detecção na troca de ambiente. A intenção aqui foi fornecer um sistema completo e o mais similar possível ao sistema o qual se tentou explorar - neste caso a câmera - a fim de ludibriar o possível atacante. Tratando-se de um container rodando na VM1, todas alterações no *honeypot* estiveram isoladas do sistema original, evitando comprometer o sistema raiz. Em conjunto com o *honeypot* a ferramenta *strace* presente na VM1 possibilitou também capturar o comportamento do cliente atuando no *honeypot* garantindo a captura dos comandos executados para análise e entendimento do atacante.

5.6 Resultados experimentais com dados sintéticos

Com a implementação do sistema experimental, como previamente descrito, testes controlados foram executados a fim de comprovar o funcionamento do experimento. Para isso manteve-se o sistema rodando em uma rede local sem conexão com a internet evi-

tando assim interferência externa. Simulou-se tentativas de acesso utilizando um cliente bash com conexão proveniente de outro computador na rede local. Para testar o funcionamento dos módulos e suas diferentes abordagens foram realizadas tentativas de acesso aplicando comportamentos considerados suspeitos e legítimos pelo sistema. A seguir estão listados os resultados obtidos separados pelos módulos presentes.



Figura 50: Captura de tráfego com conexão Telnet.

Gerenciador de login: Foi possível implementar um gerenciador de login que executou corretamente sua função. Ele serviu uma conexão SSH que ao receber conexão direcionou o cliente para a câmera de forma transparente e em tempo real. A principal vantagem aqui foi que a conexão SSH diferentemente da conexão Telnet (presente na câmera) possui uma conexão segura. A Figura 50 nos apresenta uma captura da comunicação realizada pelo cliente diretamente à câmera utilizando sua conexão Telnet. Como podemos ver é possível visualizar e até mesmo alterar todas as trocas de mensagens incluindo login e senha. No retângulo 1 é possível ver o login "teste" e sua respectiva senha "1234". No retângulo 2, com o usuário já tendo realizado *login*, foi possível capturar o comando

“cat /etc/passwd” que exibe em tela o arquivo *passwd* do linux. Esse arquivo contém todos usuários do sistema e foi capturado o *print* como vemos no retângulo 3. Já na Figura 51 observamos uma conexão feita através do sistema utilizando sua conexão SSH. A comunicação é totalmente criptografada não sendo possível extrair informação útil, tão pouco alterar seu conteúdo. O retângulo destacado nessa figura explicita que a informação interceptada apesar de conter exatamente os mesmos comandos da Figura 50, por estar criptografada, não representa nada para o observador. Desta forma como resultado da utilização do sistema obtivemos uma conexão segura com um elemento que não possuía comunicação segura.

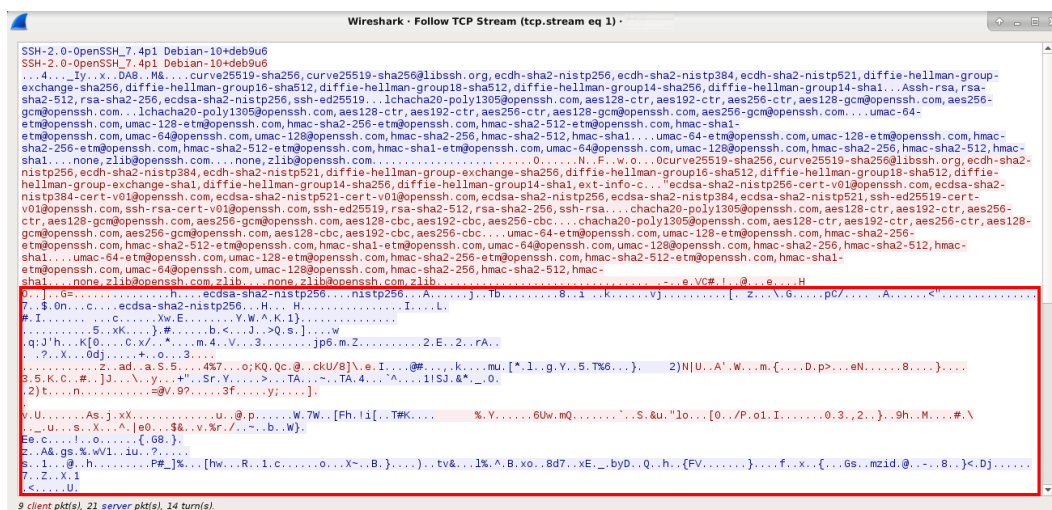
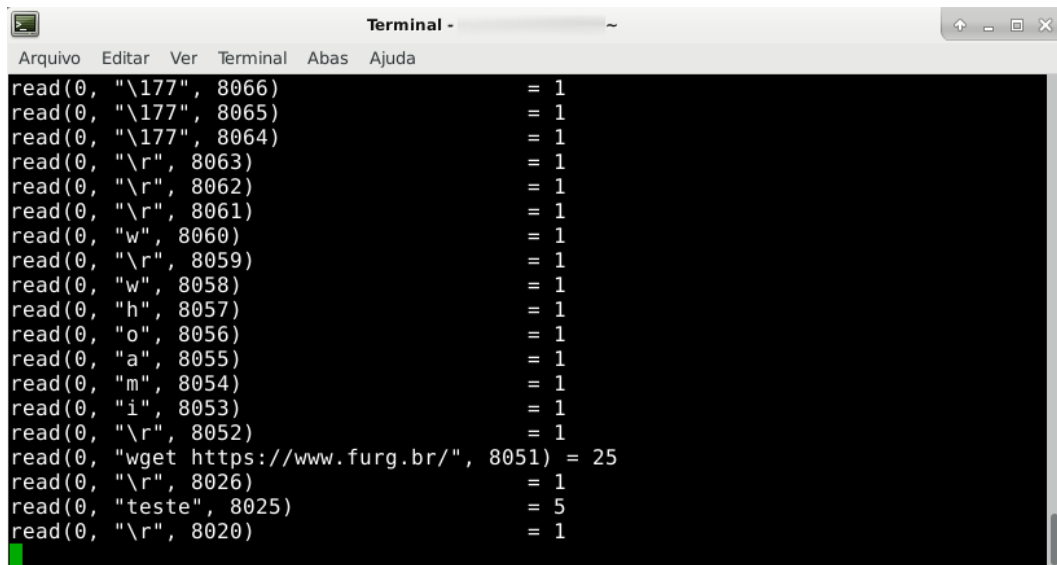


Figura 51: Captura de tráfego com conexão SSH.

Analisador de tráfego: O módulo conseguiu identificar qualquer tecla pressionada na aplicação cliente no momento de execução bem como comandos enviados. A Figura 52 ilustra a captura em tempo real de um acesso por parte do cliente ao serviço SSH do sistema. Como se pode ver na Figura o módulo consegue capturar além de comandos prontos enviados também as teclas que estão sendo pressionadas pelo usuário enquanto utiliza o serviço SSH. Isso permitiu ao sistema interceptar todas comunicação que foi direcionada a câmera - através da conexão Telnet - antes que um comando fosse executado na mesma. Ao identificar uma tentativa de comando considerado malicioso, baseado em sua lista pré definida, o módulo finalizou a conexão Telnet com a câmera e a redirecionou ao *honeypot*.

Honeypot: O *honeypot* forneceu um ambiente virtualizado simulando os serviços da câmera, isolado do sistema principal e totalmente funcional. Esse ambiente possibilitou mimetizar o comportamento da câmera e todas ações provenientes do cliente puderam ser capturadas para análise. Para captura foi utilizado também a ferramenta strace diretamente no kernel hospedeiro, neste caso o sistema principal, evitando assim a possibilidade de detecção por parte do cliente visto que o container está virtualmente isolado. A Figura 53 apresenta um *log* gerado por um cliente que teve sua conexão roteada ao *honeypot*.



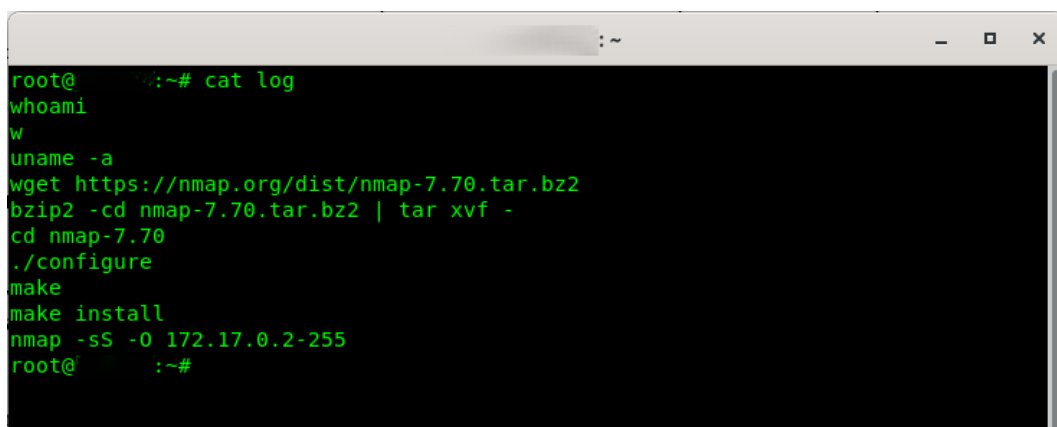
```

Terminal -
Arquivo  Editar  Ver  Terminal  Abas  Ajuda
read(0, "\177", 8066)      = 1
read(0, "\177", 8065)      = 1
read(0, "\177", 8064)      = 1
read(0, "\r", 8063)        = 1
read(0, "\r", 8062)        = 1
read(0, "\r", 8061)        = 1
read(0, "w", 8060)          = 1
read(0, "\r", 8059)        = 1
read(0, "w", 8058)          = 1
read(0, "h", 8057)          = 1
read(0, "o", 8056)          = 1
read(0, "a", 8055)          = 1
read(0, "m", 8054)          = 1
read(0, "i", 8053)          = 1
read(0, "\r", 8052)         = 1
read(0, "wget https://www.furg.br/", 8051) = 25
read(0, "\r", 8026)         = 1
read(0, "teste", 8025)      = 5
read(0, "\r", 8020)         = 1

```

Figura 52: Captura de comandos em tempo de execução.

Como pode ser visto, esse log apresenta um histórico de comandos de um cliente que teve um comportamento malicioso. Inicialmente o cliente verifica seu nível de permissão no sistema e ao identificar que seu acesso é *root* verifica se há alguém conectado no sistema além dele. Ele então obtém informações do sistema no qual está conectado. Em seguida faz download da ferramenta nmap e sua instalação. Finalmente ele executa uma varredura na rede local a fim encontrar outras máquinas que possa comprometer.



```

root@ ~:~# cat log
whoami
w
uname -a
wget https://nmap.org/dist/nmap-7.70.tar.bz2
bzip2 -cd nmap-7.70.tar.bz2 | tar xvf -
cd nmap-7.70
./configure
make
make install
nmap -sS -O 172.17.0.2-255
root@ ~:~#

```

Figura 53: Log de comandos gerado pelo honeypot.

5.7 Conclusão do estudo de caso

Conforme visto, foi possível implementar um sistema experimental capaz de intermediar a comunicação de um elemento IoT virtualizado e uma aplicação cliente, fornecendo maior segurança em sua comunicação além de identificar um comportamento suspeito por parte do cliente, baseado em dados previamente definidos, a fim de evitar o comprome-

timento do elemento que recebeu o tunelamento de sua comunicação. Para o estudo de caso utilizou-se como referência a câmera *Wireless IP Camera 360* com vulnerabilidades conhecidas e bem documentas para a simulação do experimento. Os resultados comprovaram que é possível criar um sistema que intermedeie a comunicação de forma transparente ao cliente agregando uma camada extra de segurança na comunicação através do uso de protocolo criptografado. Ainda foi demonstrado a possibilidade de compartilhar o ambiente com um *honeypot* virtualizado e isolado que ficou responsável e teve capacidade de capturar comandos recebidos de clientes considerados maliciosos pelo sistema.

5.8 Limitações

Como um grande número de dispositivos de IoT opera de maneira autônoma, sem ou com políticas e metodologias limitadas de proteção contra adulteração, um invasor pode aproveitar o acesso físico a um dispositivo para causar danos significativos. Um acesso físico pode fornecer ao atacante a possibilidade de alterar os serviços de um elemento IoT, ou fornecer acesso ilimitado aos dados armazenados em sua memória. Desta forma, um acesso físico ao hardware possibilita modificar parâmetros de inicialização do sistema, extrair a senha de root e aprender outras informações confidenciais/privadas além de vários tipos de ataques a rede (WURM et al., 2016). O IoTTunnel não possui nenhuma abordagem que seja capaz de mitigar esse tipo de ataque, possibilitando que um atacante que possua esse tipo de acesso a um elemento ou uma rede elementos conectados ao IoTTunnel possa explorar esse(s) elemento(s) de forma a alterar o comportamento do(s) mesmo(s) prejudicando o funcionamento total ou parcial da solução proposta. Mais especificamente o IoTTunnel atua na camada de rede tendo impactos na camada de processamento e de aplicação, no entanto ataques que visem explorar a camada de percepção, por exemplo, não poderão ser detectados pelo sistema. Do ponto de vista da implementação, a utilização de containers para criação de *honeypots* locais limitam a simulação de um ambiente com a mesma arquitetura do hardware hospedeiro do sistema impossibilitando, por exemplo, a simulação fidedigna de múltiplas arquiteturas que elementos conectados ao sistema venham a ter. Dessa forma binários que tenham sido compilados para arquiteturas diferentes da utilizada pelo IoTTunnel ou que devam ser compiladas no sistema em execução não poderão ser executadas, limitando a capacidade do *honeypot* em extrair informações de comportamento do atacante e até mesmo facilitar a detecção de que o sistema é na verdade *honeypot*.

6 CONCLUSÃO

Sistemas IoT são muito populares hoje em dia e a previsão é que sua popularidade continue a aumentar nos próximos anos. Eles fazem parte de um paradigma conhecido como computação ubíqua e pervasiva o qual é definido por estar sempre presentes em nossas vidas e por toda parte, interagindo entre si e com as pessoas de forma imperceptível. No entanto essas características apesar de nos trazerem várias vantagens tecnológicas acabam nos tornando vulneráveis a falhas que possam existir e ser exploradas colocando em risco nossa segurança e privacidade. Levando em consideração esses fatores, esse trabalho propôs um sistema para mitigar as vulnerabilidades e formas de exploração desses tipos de elementos. Para isso foi implementado um *honeypot* de baixa interatividade com o intuito de analisar o comportamento de atacantes e ferramentas automatizadas e identificar incidência de ataques na região. Baseando-se em vulnerabilidades e pesquisas conhecidas e a análise dos resultados experimentais obtidos e documentados no *honeypot* foi apresentado a proposta de um sistema para mitigação de vulnerabilidades na comunicação de elementos IoT. O estudo propôs um sistema modular capaz de intermediar a comunicação dos elementos IoT - potencialmente vulneráveis - com o meio externo, agregando uma camada de segurança extra através do uso de protocolos mais seguros e atualizados. Ainda o sistema se propõe a capturar o comportamento de ataques que eventualmente tenham sido bem sucedidos, através do uso de um *honeypot* interno ao mesmo. Foi implementado um sistema experimental baseado em um elemento com vulnerabilidades conhecidas e documentadas. Esse sistema experimental implementou parcialmente o sistema proposto e foi possível comprovar a viabilidade do funcionamento de alguns módulos e suas funcionalidades. Como resultado foi apresentado o aprimoramento na segurança da comunicação do elemento através do uso de um protocolo de acesso remoto criptografado e a possibilidade de captura de comportamento malicioso por parte do cliente através do uso de um *honeypot* atuando em acessos considerados indevidos. Considerando isso a proposta se mostrou viável e uma possibilidade na luta contra cibercrime e agentes maliciosos que visam explorar vulnerabilidades na comunicação de elementos IoT.

6.1 Trabalhos Futuros

Como trabalhos futuros, planeja-se comparar os resultados obtidos no *honeypot* e tratados com dados públicos nacionais, para avaliar os riscos de expor um elemento IoT na internet com mais precisão. Analisar a viabilidade do uso de wormholes na realocação da conexão para possibilitar o uso de servidores externos como *honeypots* fornecendo assim um ambiente com maior capacidade de processamento viabilizando o uso de múltiplas arquiteturas. Implementar virtualmente os outros módulos presentes na proposta do sistema a fim de testar todo potencial funcional do mesmo. Após testes virtuais planeja-se implantar a aplicação final em um hardware para testes utilizando elementos IoT reais para então obter resultados mais realísticos. Finalmente após obter um sistema funcional e testado em laboratório planeja-se expor o sistema na internet para observar o comportamento de atacantes e analisar potenciais ataques que sejam capturados no *honeypot*.

REFERÊNCIAS

ABNT. **ABNT NBR ISO/IEC 27001:2013**. [Online; acessado 28-ago-2019], <https://www.abntcatalogo.com.br/norma.aspx?ID=306580>.

ANALYTICS, I. **State of the IoT 2018**: Number of IoT devices now at 7B. [Online; acessado 09-abr-2019], <https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/>.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Computer networks**, [S.l.], v.54, n.15, p.2787–2805, 2010.

BARCENA, M. B.; WUEEST, C.; LAU, H. **Security Response**: How safe is your quantified self? [Online; acessado 10-abr-2019], https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/how-safe-is-your-quantified-self.pdf.

BARRETT, D.; KIPPER, G. **Virtualization and forensics**: A digital forensic investigator's guide to virtual environments. [S.l.]: Syngress, 2010.

BLAAUW, G. A.; BROOKS JR, F. P. **Computer architecture**: Concepts and evolution. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 1997.

BURANDE, A.; PISE, A.; DESAI, S.; MARTIN, Y.; D'MELLO, S. Wireless Network Security by SSH Tunneling. **Int. J. Sci. Res. Publ**, [S.l.], v.4, n.1, p.2250–3153, 2014.

CARDENAS, A.; AMIN, S.; SINOPOLI, B.; GIANI, A.; PERRIG, A.; SASTRY, S. Challenges for securing cyber physical systems. In: WORKSHOP ON FUTURE DIRECTIONS IN CYBER-PHYSICAL SYSTEMS SECURITY, 2009. **Anais...** [S.l.: s.n.], 2009. p.5.

CERT.BR. **honeyTARG Honeynet Project**. [Online; acessado 19-ago-2019], <https://honeytarg.cert.br/>.

CHEN, K.; ZHANG, S.; LI, Z.; ZHANG, Y.; DENG, Q.; RAY, S.; JIN, Y. Internet-of-things security and vulnerabilities: taxonomy, challenges, and practice. **Journal of Hardware and Systems Security**, [S.l.], v.2, n.2, p.97–110, 2018.

CHERDANTSEVA, Y.; HILTON, J. A reference model of information assurance & security. In: AVAILABILITY, RELIABILITY AND SECURITY (ARES), 2013 EIGHTH INTERNATIONAL CONFERENCE ON, 2013. **Anais...** [S.l.: s.n.], 2013. p.546–555.

CISCO. **New VPNFilter malware targets at least 500K networking devices worldwide.** [Online; acessado 10-abr-2019], <https://blog.talosintelligence.com/2018/05/VPNFilter.html>.

DOCKER. **Enterprise Container Platform.** [Online; acessado 022-jul-2019], <https://www.docker.com/>.

ECONOMIST, T. **Planet of the phones.** [Online; acessado 28-ago-2019], <https://www.economist.com/leaders/2015/02/26/planet-of-the-phones>.

ESET. **The sad stats on state of cybersecurity: 70% attack go unchecked.** [Online; acessado 28-ago-2019], <http://www.welivesecurity.com/2015/09/09/cybercrime-growing-concern-americans/>.

FOROUSHANI, V. A.; ADIBNIA, F.; HOJATI, E. Intrusion detection in encrypted accesses with SSH protocol to network public servers. In: INTERNATIONAL CONFERENCE ON COMPUTER AND COMMUNICATION ENGINEERING, 2008., 2008. **Anais...** [S.l.: s.n.], 2008. p.314–318.

GALEGALE, G. P.; SIQUEIRA, É.; SILVA, C. B. H.; SOUZA, C. A. d. THE INTERNET OF THINGS APPLIED TO BUSINESS: A BIBLIOMETRIC STUDY. **JISTEM-Journal of Information Systems and Technology Management**, [S.l.], v.13, n.3, p.423–438, 2016.

GANDHI, U. D.; KUMAR, P. M.; VARATHARAJAN, R.; MANOGARAN, G.; SUNDARASEKAR, R.; KADU, S. HIoT POT: surveillance on IoT devices against recent threats. **Wireless personal communications**, [S.l.], v.103, n.2, p.1179–1194, 2018.

GARCIA, E. **IT Security Research WirelessIP camera family.** [Online; acessado 09-abr-2019], https://github.com/eloygn/IT_Security_Research_WirelessIP_camera_family.

GARTNER, I. **Gartner.** "[Online; acessado 28-ago-2019]", "<http://www.gartner.com/newsroom/id/3404817>".

GIUSTO, D.; IERA, A.; MORABITO, G.; ATZORI, L. **The internet of things: 20th Tyrrhenian workshop on digital communications.** [S.l.]: Springer Science & Business Media, 2010.

GOLLMANN, D.; KROTOFIL, M. Cyber-Physical Systems Security. In: **The New Co-debreakers.** [S.l.]: Springer, 2016. p.195–204.

GUARNIZO, J. D.; TAMBE, A.; BHUNIA, S. S.; OCHOA, M.; TIPPENHAUER, N. O.; SHABTAI, A.; ELOVICI, Y. Siphon: Towards scalable high-interaction physical honeypots. In: ACM WORKSHOP ON CYBER-PHYSICAL SYSTEM SECURITY, 3., 2017. **Proceedings...** [S.l.: s.n.], 2017. p.57–68.

HACKOHACKU. **VPN vs Proxy**. [Online; acessado 022-jul-2019], <http://hackohacku.blogspot.com/2015/07/vpn-vs-proxy.html>.

HAFEEZ, I.; DING, A. Y.; SUOMALAINEN, L.; KIRICHENKO, A.; TARKOMA, S. Securebox: Toward safer and smarter IoT networks. In: ACM WORKSHOP ON CLOUD-ASSISTED NETWORKING, 2016., 2016. **Proceedings...** [S.l.: s.n.], 2016. p.55–60.

HENNESSY, J. L.; PATTERSON, D. A. **Computer architecture**: a quantitative approach. [S.l.]: Elsevier, 2011.

INSTITUTE, G. T. **Emerging Cyber Threats Report 2016**. [Online; acessado 28-ago-2019], http://www.iisp.gatech.edu/sites/default/files/documents/2016_georgiatech_cyberthreatsreport_online

IPWITHEASE. **FIREWALL VS IPS VS IDS**. [Online; acessado 022-jul-2019], <https://ipwithease.com/firewall-vs-ips-vs-ids/>.

KASPERSKY, L. **CYBERTHREAT REAL-TIME MAP**. "[Online; acessado 19-ago-2019]", "<https://cybermap.kaspersky.com/>".

KHAN, R.; KHAN, S. U.; ZAHEER, R.; KHAN, S. Future internet: the internet of things architecture, possible applications and key challenges. In: 2012 10TH INTERNATIONAL CONFERENCE ON FRONTIERS OF INFORMATION TECHNOLOGY, 2012. **Anais...** [S.l.: s.n.], 2012. p.257–260.

KOLYSHKIN, K. Virtualization in linux. **White paper, OpenVZ**, [S.l.], v.3, p.39, 2006.

KUKULSKA-HULME, A. Introduction. In: **Mobile Learning**. [S.l.]: Routledge, 2007. p.17–22.

LABORATORY, W. **Cyber-physical Systems (CPS)**. [Online; acessado 28-ago-2019], <https://www.jaist.ac.jp/is/labs/lim-lab/research.php>.

LEE, E. A. Cyber physical systems: Design challenges. In: IEEE INTERNATIONAL SYMPOSIUM ON OBJECT AND COMPONENT-ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC), 2008., 2008. **Anais...** [S.l.: s.n.], 2008. p.363–369.

LINDNER, M. **Online sales will reach \$523 billion by 2020 in the U.S**. [Online; acessado 28-ago-2019], <https://www.digitalcommerce360.com/2016/01/29/online-sales-will-reach-523-billion-2020-us/>.

MITRE. **CVE-2017-11632**. [Online; acessado 09-abr-2019], <https://www.cvedetails.com/cve/CVE-2017-11632/>.

MITRE. **CVE-2017-11633**. [Online; acessado 09-abr-2019], <https://www.cvedetails.com/cve/CVE-2017-11633/>.

MITRE. **CVE-2017-11634**. [Online; acessado 09-abr-2019], <https://www.cvedetails.com/cve/CVE-2017-11634/>.

MITRE. **CVE-2017-11635**. [Online; acessado 09-abr-2019], <https://www.cvedetails.com/cve/CVE-2017-11635/>.

MOKUBE, I.; ADAMS, M. Honeypots: concepts, approaches, and challenges. In: OF THE 45TH ANNUAL SOUTHEAST REGIONAL CONFERENCE, 2007. **Proceedings...** [S.l.: s.n.], 2007. p.321–326.

NASCIMENTO, E. N. Lucas do; PIETZ, F.; WATANABE. Processador ARM Cortex-A9. In: . [S.l.: s.n.], 2015.

NESHENKO, N.; BOU-HARB, E.; CRICHIGNO, J.; KADDOUM, G.; GHANI, N. Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations. **IEEE Communications Surveys & Tutorials**, [S.l.], 2019.

NEWS, T. H. **Friday's Massive DDoS Attack Came from Just 100,000 Hacked IoT Devices**. [Online; acessado 10-abr-2019], <https://thehackernews.com/2016/10/ddos-attack-mirai-iot.html>.

NEWS, T. H. **New Rapidly-Growing IoT Botnet Threatens to Take Down the Internet**. [Online; acessado 10-abr-2019], <https://thehackernews.com/2017/10/iot-botnet-malware-attack.html>.

NEWS, T. H. **Mirai Variant Adds Dozen New Exploits to Target Enterprise IoT Devices**. [Online; acessado 10-abr-2019], <https://thehackernews.com/2019/03/mirai-botnet-enterprise-security.html>.

OWASP. **OWASP TOP10 Internet of Things 2018**. [Online; acessado 09-abr-2019], https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project.

PA, Y. M. P.; SUZUKI, S.; YOSHIOKA, K.; MATSUMOTO, T.; KASAMA, T.; ROS-SOW, C. IoTPOT: analysing the rise of IoT compromises. **EMU**, [S.l.], v.9, p.1, 2015.

PA, Y. M. P.; SUZUKI, S.; YOSHIOKA, K.; MATSUMOTO, T.; KASAMA, T.; ROS-SOW, C. IoTPOT: A novel honeypot for revealing current IoT threats. **Journal of Information Processing**, [S.l.], v.24, n.3, p.522–533, 2016.

PONEMON, I. **2015 Cost of Cyber Crime Study**: Global. [Online; acessado 28-ago-2019], <https://ssl.www8.hp.com/ww/en/secure/pdf/4aa5-5207enw.pdf>.

PONEMON, I. **THE COST OF CYBERCRIME**. [Online; acessado 09-abr-2019], https://www.accenture.com/t20190305T185301Z_w_/us-en/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf.

PORTNOY, M. **Virtualization essentials**. [S.l.]: John Wiley & Sons, 2012. v.19.

PROJECT, H. **The Honeynet Project**. [Online; acessado 19-ago-2019], <https://www.honeynet.org/about>.

PROVOS, N. et al. A Virtual Honeypot Framework. In: USENIX SECURITY SYMPOSIUM, 2004. **Anais...** [S.l.: s.n.], 2004. v.173, p.1–14.

PROVOS, N.; HOLZ, T. **Virtual honeypots**: from botnet tracking to intrusion detection. [S.l.]: Pearson Education, 2007.

RESEARCH, J. **Cybercrime the Internet of Threats 2017**. [Online; acessado 28-ago-2019], <https://www.juniperresearch.com/document-library/white-papers/cybercrime-the-internet-of-threats-2017>.

RICHARDS, R. J. **The next wave**. [Online; acessado 10-abr-2019], <https://www.nsa.gov/Portals/70/documents/resources/everyone/digital-media-center/publications/the-next-wave/TNW-21-2.pdf>.

SALTZER, J. H.; SCHROEDER, M. D. The protection of information in computer systems. **Proceedings of the IEEE**, [S.l.], v.63, n.9, p.1278–1308, 1975.

SECURITY, R. **2016**: Current State of Cybercrime. [Online; acessado 12-ago-2019].

SHACKLEFORD, D. **Virtualization security**: protecting virtualized environments. [S.l.]: John Wiley & Sons, 2012.

SIKORSKI, M.; HONIG, A. **Practical malware analysis**: the hands-on guide to dissecting malicious software. [S.l.]: no starch press, 2012.

SINGULARITY. **Microprocessor Cost Per Transistor Cycle**. [Online; acessado 19-ago-2019], <http://www.singularity.com/charts/page17.html>.

SPITZNER, L. **Honeypots**: tracking hackers. [S.l.]: Addison-Wesley Reading, 2003. v.1.

SPITZNER, L. Honeypots: Catching the insider threat. In: COMPUTER SECURITY APPLICATIONS CONFERENCE, 2003. PROCEEDINGS. 19TH ANNUAL, 2003. **Anais...** [S.l.: s.n.], 2003. p.170–179.

SPITZNER, L. The honeynet project: Trapping the hackers. **IEEE Security & Privacy**, [S.l.], v.99, n.2, p.15–23, 2003.

SYMANTEC. **Security Center Archived Publications**. [Online; acessado 10-abr-2019], <https://www.symantec.com/security-center/archived-publications#accordion-threat-reports>.

TANENBAUM, A. S.; VAN STEEN, M. **Distributed systems**. [S.l.]: Prentice-Hall, 2007.

TECHNOLOGIES, P. **Practical ways to misuse a router**. [Online; acessado 09-abr-2019], <http://blog.ptsecurity.com/2017/06/practical-ways-to-misuse-router.html>.

TENABLE. **Nessus Professional**. [Online; acessado 28-ago-2019], <https://pt-br.tenable.com/products/nessus/nessus-professional>.

VENTURES, C. **Hackerpocalypse: A Cybercrime Revelation**. [Online; acessado 28-ago-2019], <https://www.herjavecgroup.com/hackerpocalypse-cybercrime-report/>.

VIRUSTOTAL, G. **VirusTotal, Free Online Virus, Malware and URL Scanner**. [Online; acessado 28-ago-2019], <https://www.virustotal.com/>.

WEISER, M. The Computer for the 21 st Century. **Scientific american**, [S.l.], v.265, n.3, p.94–105, 1991.

WHITMAN, M. E.; MATTORD, H. J. **Principles of information security**. [S.l.]: Cengage Learning, 2011.

WURM, J.; HOANG, K.; ARIAS, O.; SADEGHI, A.-R.; JIN, Y. Security analysis on consumer and industrial IoT devices. In: ASIA AND SOUTH PACIFIC DESIGN AUTOMATION CONFERENCE (ASP-DAC), 2016., 2016. **Anais...** [S.l.: s.n.], 2016. p.519–524.

XIA, F.; YANG, L. T.; WANG, L.; VINEL, A. Internet of things. **International Journal of Communication Systems**, [S.l.], v.25, n.9, p.1101, 2012.

YUN, M.; YUXIN, B. Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid. In: INTERNATIONAL CONFERENCE ON ADVANCES IN ENERGY ENGINEERING, 2010., 2010. **Anais...** [S.l.: s.n.], 2010. p.69–72.