# Caracterização Escalável de Vulnerabilidades de Segurança: um Estudo de Caso na Internet Brasileira

**Lucas M. Ponce**[1] **Matheus Gimpel**[1] **Elverton Fazzion**[2,1]
**Ítalo Cunha**[1] **Cristine Hoepers**[3] **Klaus Steding-Jessen**[3]
**Marcelo H. P. C. Chaves**[3] **Dorgival Guedes**[1] **Wagner Meira Jr.**[1]

[1]DCC/UFMG [2]DCOMP/UFSJ [3]CERT.br/NIC.br

{lucasmsp,matheusgimpel,cunha,dorgival,meira}@dcc.ufmg.br
{cristine,jessen,mhp}@cert.br fazzion@ufsj.edu.br

***Abstract.*** *Monitoring services such as Shodan are increasingly popular for tracking applications and vulnerabilities on the Internet. In this paper we analyze monitoring data from Shodan to characterize vulnerabilities found in Brazilian networks. In addition, we discuss Data Science methods to scale and improve the depth of the analyses, and combine external network and vulnerability metadata to support richer results and conclusions. Our characterization exposes several vulnerabilities of high severity, and some of them remain widespread despite being five years old. We hope that the analyses presented in this paper will encourage organizations to deploy updates and protection mechanisms to mitigate these threats.*

***Resumo.*** *Serviços de monitoramento como o Shodan são cada vez mais populares no rastreamento de aplicações e vulnerabilidades na Internet. Neste artigo caracterizamos e discutimos vulnerabilidades encontradas na Internet brasileira utilizando dados de monitoramento provenientes do Shodan. Além disso, discutimos métodos de Ciências dos Dados para melhorar a escalabilidade e a qualidade de análises; também combinamos metadados de fontes complementares sobre rede e vulnerabilidades para extrair resultados e conclusões mais assertivos. Nossa caracterização expõe diversas vulnerabilidades de alta severidade na rede brasileira, algumas catalogadas há mais de cinco anos e que continuam prevalentes até hoje. Esperamos que as análises apresentadas neste artigo incentivem organizações a implantarem atualizações e mecanismos de proteção para mitigar essas ameaças.*

## 1. Introdução

O número de dispositivos de Internet das Coisas (IoT) e serviços conectados à Internet vem aumentando a cada ano. No entanto, esse rápido crescimento vem tornando os métodos tradicionais de segurança de rede ineficientes [Mousavi et al. 2020]. Um levantamento[1] apontou que o número de ciberataques por semana em redes corporativas cresceu pelo menos 50% em comparação com 2020. Esses ataques, cada vez mais frequentes, impulsionam o aprimoramento de técnicas de monitoramento e um consequente aumento de volume e de complexidade dos dados coletados, exigindo soluções cada vez mais complexas e maiores investimentos na área de cibersegurança.

[1]https://blog.checkpoint.com/2022/01/10/check-point-research-cyber-attacks-increased-50-year-over-year/

Sistemas de buscas como o Shodan (https://shodan.io) auxiliam no rastreamento de vulnerabilidades. Sistemas desse tipo varrem (*scan*) a Internet em busca de dispositivos executando aplicações acessíveis pela rede, permitindo análises puntuais e longitudinais. Por exemplo, equipes de segurança podem usar esses motores de busca como uma das etapas em testes de penetração (*Pentest*) para a coleta de informações sobre os dispositivos e suas vulnerabilidades, melhorando a segurança de redes. No entanto, apesar da popularidade desses sistemas, o processamento de análises complexas nem sempre é uma tarefa fácil devido a ruídos nos dados bem como à dificuldade de escala face à grande quantidade de dados coletados. Motores de buscas como o Shodan são capazes de sondar e coletar informações de aproximadamente 500 destinos por segundo. Nesse cenário, mesmo utilizando técnicas de amostragem, uma simples operação de filtragem pode ser inviável em uma arquitetura tradicional.

Nosso trabalho apresenta técnicas para escalar e melhorar a qualidade das análises de dados provenientes de motores de busca. As técnicas desenvolvidas agregam metadados às informações coletadas pelos sistemas de monitoramento e permitem especialistas com conhecimento do domínio responder perguntas práticas sobre os dados de forma eficiente e escalável, utilizando técnicas de mineração de dados e aprendizado de máquina em um ambiente de processamento massivo de dados (*Big Data*). Além disso, demonstramos a sua efetividade apresentando uma caracterização de vulnerabilidades na Internet brasileira, identificando, por exemplo, que vulnerabilidades graves ficam expostas por meses sem ser corrigidas ou protegidas.

Nossas técnicas podem ser utilizadas por outros grupos de pesquisa e operadores de rede como ponto de partida para análises mais amplas e em outros contextos. Esperamos ainda que nosso trabalho motive organizações e operadores de redes a implantarem atualizações de *software* e mecanismos de proteção como *firewalls* para mitigar ameaças. Por último, nossos resultados são úteis para esforços de conscientização e treinamento em termos de vulnerabilidades e dispositivos vulneráveis.

## 2. Shodan

O Shodan é um motor de busca lançado em 2009 que identifica dispositivos e aplicações em execução, acessíveis na Internet. Ele varre a Internet tentando conectar a endereços IP e portas gerados aleatoriamente. As tentativas de conexão são realizadas por centenas de diferentes módulos que implementam lógica específica para coletar informações detalhadas de diferentes aplicações. Por exemplo, o módulo de sondagem do MongoDB consegue coletar informações sobre o status do serviço, a lista de *databases*, a versão e até se o serviço possui algum tipo de autenticação. O Shodan também utiliza um catálogo de informações para enriquecer os dados coletados com campos derivados como o nome ou versão de um produto, ou o sistema operacional do dispositivo.

O presente trabalho utiliza dados do Shodan do período entre 01/01/2021 e 30/04/2021, que foram compartilhados com o Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br) em uma parceria para mapear vulnerabilidades no espaço de endereçamento IPv4 do Brasil. A informação é organizada em *registros*, que representam um sumário de um acesso do Shodan a uma aplicação (endereço IP e porta), com estabelecimento de conexões e, em alguns casos, troca de dados com os serviços identificados pelos coletores Shodan. Como dispositivos podem possuir diferentes aplicações em execução, registros para um mesmo dispositivo podem

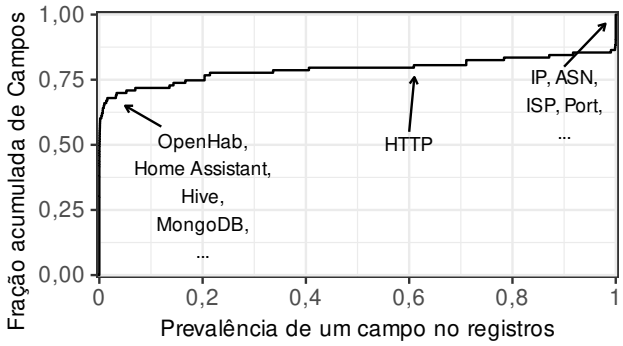


**Figura 1. Prevalência dos campos nos registros do dataset.**

conter campos distintos que dependerão da aplicação sondada. O período da amostra contém 112.167.097 registros com uma média de 89 campos cada (113 campos distintos no total) com tamanho total de aproximadamente 229 GB comprimidos com bzip2.

O gráfico da figura 1 apresenta a distribuição acumulada da frequência dos campos entre os registros. Em geral, informações de uma aplicação para a qual o Shodan possui um monitor são capturadas em um campo no registro. Para aplicações ou protocolos mais comuns, o Shodan pode criar campos derivados, como os de informações do provedor ou região de hospedagem para aplicações na nuvem, ou informações das chaves criptográficas em serviços SSH. Cerca de 62,5% dos tipos de campos existentes ocorrem em apenas 1% dos registros. Esses campos são referentes a aplicações sondadas por módulos específicos do Shodan, por exemplo: aplicações de armazenamento de dados como MongoDB, Hive e o InfluxDB; aplicações de automação residencial como Home Assistant e OpenHAB. Estes campos capturam dados detalhados sobre instâncias específicas de aplicações na Internet, permitindo análises de vulnerabilidades em granularidade fina. Um campo pode ser criado por mais de um módulo de sondagem. Embora cada módulo seja especializado em um tipo de serviço, é comum o Shodan coletar informações relacionadas, por exemplo, serviços como o SSH ou SSL podem ser identificados por diversos módulos como o http e mysql.

Embora o Shodan varra e colete informações de dispositivos na Internet, as técnicas atuais de refinamento dos dados se limitam ao enriquecimento por IP ou estruturação das informações coletadas. Além disso, a interface de busca *web* é limitada essencialmente à filtragem por campos. Expressões regulares ou execução de análises específicas, por exemplo, não são suportadas. Por causa dessas limitações, o *download* dos dados e o processamento local é necessário para a maior parte das análises usualmente realizadas. No entanto, análises de tendências de tecnologias (p.ex., popularidade ao longo do tempo) ou de reputação de rede (p.ex., política de atualização de *software*), úteis para operadores e instituições de resposta a incidentes, demandam uma infraestrutura capaz de lidar com grandes volumes de dados. É esperado que soluções de caracterização suportem consultas complexas e sejam escaláveis na realização de tais consultas, em volumes de dados que chegam a terabytes, inviáveis em uma arquitetura tradicional. Este trabalho investiga os requisitos e características dessas soluções.

## 3. Metodologia de Caracterização de Vulnerabilidades de Segurança

Nesta seção apresentamos a metodologia empregada na caracterização de vulnerabilidades de segurança na Internet brasileira a partir dos dados do Shodan. O Shodan coleta

e exporta dados em sua forma bruta, sendo suscetível a erros em campos que dependem de bases externas, como o nome de organizações associadas a endereços IP (seção 3.1). Além disso, as informações presentes nos registros podem ser complementadas por outras bases de dados para permitir estudos mais complexos (seção 3.2). Por fim, análises podem ser custosas devido ao grande volume de dados. Dessa forma, buscamos ferramentas para, por exemplo, auxiliar na transformação e redução dos dados de forma a aumentar o desempenho no processamento dos dados (seção 3.3). Nossa metodologia pode ser útil para outros pesquisadores e operadores de redes interessados em utilizar dados do Shodan.

## 3.1. Depuração dos Dados

A depuração dos dados é o processo de remover ou modificar registros incoerentes, incorretos, ou incompletos de um conjunto. No Shodan os campos org e isp são comuns e centrais para qualquer análise de vulnerabilidades. Porém, estes campos podem ter dados incoerentes, incorretos ou desatualizados, uma vez que essa informação é obtida de forma terceirizada via protocolo whois. Por exemplo, "CLARO S/A" e "CLARO S.A." são nomes diferentes mas se referem à mesma organização. As etapas a seguir realizam a depuração desses campos.

### 3.1.1. Padronização dos Campos

Nessa etapa padronizamos os campos org e isp realizando (*i*) a remoção de acentos e símbolos; (*ii*) conversão para letras minusculas; (*iii*) remoção de denominações empresariais como *s/a, ltda, inc, me*; e (*iv*) substituição de nomes a partir de uma *tabela de padronização de organizações*. Esta tabela converte o nome de uma organização para um nome padronizado. Relações nessa tabela podem ser criadas manualmente, a partir de conhecimento específico sobre a entidade (por exemplo "net virtua" para "claro" ou "telefônica brasil" para "vivo") ou de forma semi-automática, como explicado na seção 3.1.2.

### 3.1.2. Identificação de Erros de Digitação

Nesta seção apresentamos duas abordagens para a inclusão de novas regras na tabela de padronização de organizações. A primeira abordagem consiste na identificação de organizações com múltiplos nomes que divergem apenas no espaçamento (por exemplo, "Eyesnwhere Sistemas Inteligentes de Imagem" e "Eyes Nwhere Sistemas Inteligentes de Imagem"). Nessa abordagem, agrupamos as organizações a partir do campo asn, que contém o número do sistema autônomo que controla o IP sondado. A premissa envolvida nessa abordagem é que uma organização com nome *A* deve compartilhar pelo menos um número de sistema autônomo para ser considerada igual a outra organização com nome *B*. Para cada grupo, realizamos uma comparação entre todos os pares de nomes. Selecionamos os pares cujos nomes são idênticos (ignorando diferenças de espaçamento) e os adicionamos a tabela de padronização de organizações.

A segunda abordagem identifica variações mais significativas no nome de uma organização para gerar regras. Por exemplo, "*Asa Group Monitoramento e Servicos de Telecom*" e "*Asa Group Monitoramento e Servios de Telecom*" representam a mesma organização, mas com um erro de escrita. Para identificar essas variações, primeiro agrupamos as organizações par-a-par a partir de um endereço IP em comum e calculamos a

similaridade $s$ entre cada par utilizando a distância de Levenshtein, *i.e.,* o número mínimo de edições necessárias para transformar uma palavra na outra, normalizada pelo tamanho do maior nome do par. Por exemplo, o par do exemplo anterior possui distância de Levenshtein 1 e tem distância relativa $1/45 = 0,02$. Como a distância de Levenshtein normalizada é sensível ao tamanho do texto de comparação, removemos as palavras frequentes (*stopwords*) que aparecem em mais de 10% das organizações para reduzir o efeito desses termos no cálculo de similaridade.

Esta segunda abordagem equilibra o relaxamento da similaridade entre os nomes (que na abordagem anterior aceitava apenas diferenças de espaçamento) com um agrupamento mais restritivo por endereço IP (que na abordagem anterior era por asn). Consideramos que duas organizações são iguais quando elas compartilham um endereço IP e cujos nomes possuem distância de Levenshtein normalizada menor do que $\alpha = 0,4$. Em nosso experimento, $\alpha = 0,4$ foi o que apresentou o melhor compromisso entre o número de pares semelhantes encontrados e a taxa de falsos verdadeiros. Com esse limiar, conseguimos identificar casos com textos longos como também textos mais curtos, por exemplo, o par "*turbonet info*" e "*turbonet*" que possui distância normalizada de 0,3846.

Atualmente essa técnica exige conferência manual das inferências. Isso porque a técnica pode criar vários pares falso-positivos como, "*prefeitura municipal de itabira*" e "*prefeitura municipal de itapira*". No entanto, comparações desse tipo são infrequentes, pois reduzimos a chance desses casos ao agrupar os nomes por IP antes da comparação. Por exemplo, em nossos experimentos, 60 e 404 pares foram encontrados, respectivamente na primeira e na segunda abordagem. Dos 404 pares da segunda abordagem, 46% tiveram $s$ igual a zero (ou seja, depois da remoção dos *stopwords*, todos outros termos eram iguais), além disso, apenas 8% foram considerados pares falsos. No geral, as equivalências inferidas são majoritariamente verdadeiras e na maioria desses casos, correspondem a pares onde um dos textos é um sub-conjunto do outro, como em "*linsat sistemas de televisao e dados*" e "*sistemas de televisao e dados*". As organizações mapeadas na *tabela de padronização de organizações* possuem uma representatividade de 42% nos registros, o que mostra que mesmo com poucos pares encontrados, a solução possui grande impacto sobre os registros. No futuro pretendemos tratar automaticamente estes casos utilizando uma base de entidades e não permitindo equivalência entre números distintos.

**Avaliação.** A figura 2 compara o ranqueamento das organizações mais frequentes nos registros utilizando a abordagem tradicional e a com o refinamento do campo org (esse mesmo refinamento pode ser aplicado no campo isp). O dataset contém 48.467 organizações distintas, que são agrupadas em 42.197 organizações após o procedimento de padronização. Podemos observar que grandes provedores de serviços, como a Akamai Technologies e Claro, são frequentemente representadas por diferentes nomes.

### 3.1.3. Atribuição de Organização por Endereço IP

Cerca de 13% dos registros não possuem o campo org. Uma forma de completar parte desses dados é obter o nome da organização a partir do endereço IP sondado no registro. Para isso, agrupamos os registros na base de dados por prefixo /24. Para os prefixos cujos registros possuem uma única organização (considerando a padronização de nomes apresentada na seção 3.1.2), completamos todos os registros sem o campo org com a

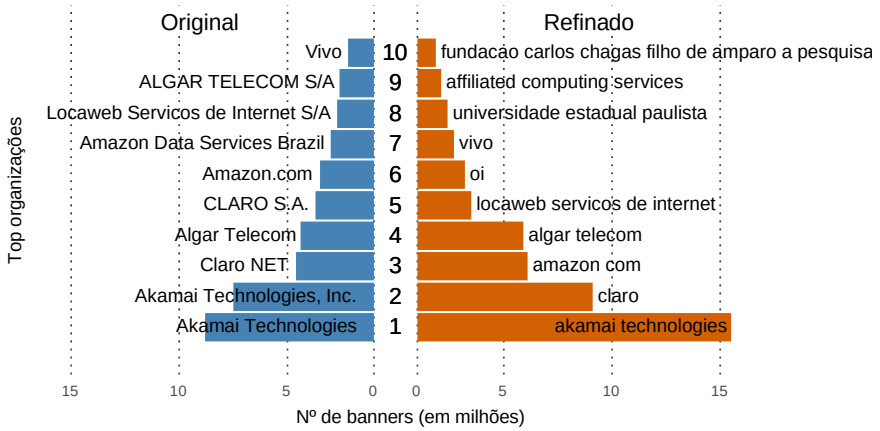


**Figura 2. Diferença de resultado na identificação das top 10 organizações, em número de registros, entre o dataset original e padronização proposta.**

organização encontrada. Adotamos o prefixo /24 nesse caso por ser o prefixo mais específico geralmente aceito em tabelas de roteamento e frequentemente utilizado como unidade mínima de alocação por provedores de rede.

**Avaliação.** Dos 221.844 prefixos /24 encontrados no dataset, 80,82% (aproximadamente 153 mil) possuem IPs associados a uma única organização, o que nos permite preencher o campo org para 70,93% dos registros com este campo faltante.

### 3.2. Integração de Metadados

A integração de metadados envolve a fusão de informações de outras fontes e permite análises mais elaboradas. A seguir descrevemos os metadados integrados às nossas análises.

**National Vulnerabilitiy Database.** Embora o Shodan forneça os códigos CVE, códigos CVSS e links de referência relativos às vulnerabilidades encontradas durante a varredura (campo vulns), essas informações podem conter imprecisões. Por exemplo, não existe uma padronização do sistema de *score* do CVSS, de forma que o *score* de um CVE pode diferir em versões diferentes. Dessa forma, integramos nosso conjunto de dados com a base National Vulnerabilitiy Database (NVD) que contém mais informações sobre CVEs e é uma fonte mais confiável dos *scores* CVSS.

**Identificação de dispositivos.** Um dispositivo pode possuir múltiplos endereços IP ou ter um endereço IP dinâmico que muda ao longo do tempo. Isso pode levar a situações em que múltiplos registros com IPs diferentes se referem a um mesmo dispositivo. Buscamos identificar registros associados a um mesmo dispositivo. Como registros coletados por módulos diferentes do Shodan possuem campos diferentes, os registros referentes a um mesmo dispositivo podem ter baixa interseção de informações, o que complica a identificação de registros referentes a um dispositivo. Para contornar este desafio, propomos um conjunto de regras, cada uma permitindo a identificação de dispositivos a partir de dados de uma aplicação específica. Para cada regra, um conjunto de campos é considerado para identificar a qual dispositivo um registro se refere. Por exemplo, o Shodan utiliza a API REST do ElasticSearch para obter o nome da instância, versão e configuração do *hardware* do hospedeiro, podemos utilizar esses campos para identificar um dispositivo.

**CAIDA AS Rank e AS Classification.** Para enriquecer os dados com informações adi-

cionais sobre sistemas autônomos, integramos os dados com as bases AS-Rank e AS Classification da CAIDA[2] via o campo asn. Dessa forma, as duas bases complementam informações sobre sistema autônomos com, por exemplo, nome, tamanho, organização responsável, país de registro e classificação de acordo com o seu tipo de negócio.

### 3.3. Transformação e Redução dos Dados

A escolha de um formato de dados para a representação tem um papel fundamental na redução do volume de dados e no tempo de execução de análises. Apesar do Shodan utilizar o formato *JSON* para armazenar registros, optamos por utilizar o Apache Parquet por permitir acesso mais eficiente aos dados. A conversão, que pode ser realizada via Apache Spark, consiste em uma estruturação dos dados, transformando campos frequentes do formato original (*JSON* e bzip2) para uma coluna no formato Parquet. Os campos infrequentes são armazenados como colunas do tipo *JSON* para evitar uma explosão do número de colunas. Essa solução é geral e permite que campos infrequentes armazenados em *JSON* sejam posteriormente estruturados em múltiplas colunas caso necessário.

Uma outra alteração no formato de dados é o campo vulns, que contém originalmente um dicionário com informações relacionadas às vulnerabilidades identificadas no dispositivo sondado. Esse campo foi reduzido em duas listas: uma contendo os códigos de CVEs das vulnerabilidades e outra contendo os CVEs do subconjunto de vulnerabilidades verificadas. Substituímos outras informações sobre as vulnerabilidades nos registros do Shodan por dados do NVD (seção 3.2), que são mais detalhados e evitam duplicação de informações, economizando espaço de armazenamento.

**Avaliação.** Avaliamos o impacto de utilizar o Spark e o Parquet no processamento de dados do Shodan. A tabela 1 apresenta a comparação do tempo de execução, considerando apenas um dia de amostra, para uma busca por dispositivos com vulnerabilidades que aparecem em pelo menos 10 registros e que têm EPSS $\geq 0{,}8$. Consideramos três combinações entre processamento em Python *vs.* Spark e entre armazenamento dos dados em formato bzip2 *vs.* Parquet. Utilizamos 20 *cores* de processadores Intel Xeon Silver 4114 e 40 GiB de RAM em um servidor Debian 11. Existem pelo menos dois fatores que contribuem para o aumento significativo de desempenho proporcionado pelo Spark sobre a solução em Python: *(i)* enquanto execuções Python são nativamente *single-core*, Spark é nativamente distribuído e *multi-core*; *(ii)* a maioria dos operadores Spark no ambiente PySpark encapsulam código escrito em Scala, uma linguagem compilada que permite otimizações de código impossíveis no Python. Aliado a isso, a utilização do Parquet permite ao Spark um outro conjunto de otimizações: *(i)* enquanto o formato *bzip2* exige serialização da leitura para de descompressão dos dados, o formato Parquet permite paralelismo ao dividir o arquivo em blocos; *(ii)* como os dados já são armazenamentos em um formato estruturado, seus tipos de dados já são definidos, evitando o processo de inferência; *(iii)* por fim, por causa do formato colunar do Parquet, o Spark desconsidera colunas desnecessárias para uma tarefa, minimizando o número de operações de entrada e saída.

Em média, a conversão de dados para a informação de um dia (1,90 GB em média) leva 4 minutos e 43 segundos. Embora o tempo gasto dessa conversão possa parecer alto, a avaliação experimental mostrada na tabela 1 indica que esse custo é menor do que o custo de processar o dado no formato original e permite aumento significativo da vazão

[2]Disponíveis em https://asrank.caida.org/ e https://catalog.caida.org/details/dataset/as_classification.

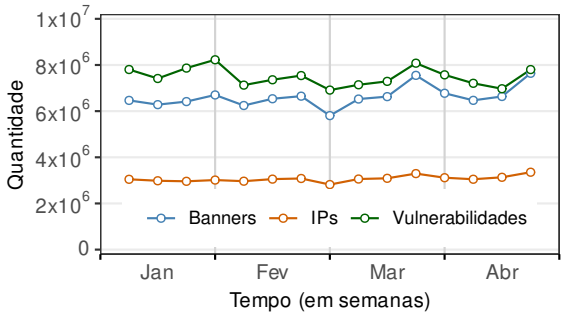


**Figura 3. Relação das vulnerabilidades ao longo do tempo.**

**Tabela 1. Comparação entre abordagens em segundos.**

| ABORDAGEM | TEMPO (MÉDIA ± CV) |
|---|---|
| Python + bzip2 | 300,0 ± 0,01 |
| Spark + bzip2 | 42,3 ± 0,02 |
| Spark + Parquet | 4,2 ± 0,39 |

**Tabela 2. Distribuição da criticidade CVSSs v2.**

| ATRIBUTO | GRAU | | |
|---|---|---|---|
| | BAIXA | MÉDIA | CRÍTICA |
| FAIXA | 0 - 3,9 | 4 - 6,9 | 7 - 10 |
| CVES (%) | 8,73 | 66,50 | 24,66 |
| NVD (%) | 15,42 | 54,23 | 30,35 |
| BANNERS (%) | $< 0{,}01$ | 3,53 | 3,27 |
| IP (%) | $< 0{,}01$ | 4,67 | 3,62 |

para processamentos e análises subsequentes. Além disso, a conversão reduziu o espaço de armazenamento em 9%, de 229 GB (bzip2) para 210 GB (Parquet).

## 4. Análises de Vulnerabilidades na Internet Brasileira

Nesta seção apresentamos a caracterização das vulnerabilidades e dispositivos encontrados em endereços atribuídos pelo NIC.br. Identificamos na seção 4.1 que o número de vulnerabilidades ao longo do tempo é aproximadamente constante, sendo mais frequentes as de grau médio e crítico. Na seção 4.2 encontramos que 72,8% dos serviços PHP hospedados no Brasil utilizam a versão 5, já desatualizada há pelo menos 2 anos, dependendo do sistema operacional. A seção 4.3 apresenta dois ASes nos quais pelo menos 99% dos seus IPs foram identificados com vulnerabilidades críticas. Por fim, na seção 4.4, observamos que protocolos obsoletos de SSL e TLS continuam a ser utilizados no Brasil.

### 4.1. Análise Temporal das Vulnerabilidades

Entender o comportamento de vulnerabilidades ao longo do tempo permite quantificar o nível de exposição de diferentes redes e quão rapidamente instalações vulneráveis são atualizadas. A figura 3 apresenta, por semana, a quantidade de registros, endereços IP e vulnerabilidades com CVE encontradas pelo Shodan em redes brasileiras. Os resultados mostram que as métricas são estáveis ao longo do período. Em média, o Shodan coleta 6,4 milhões de registros por semana, sondando 3 milhões de endereços IP, e encontra 7,3 milhões de vulnerabilidades.

Semanalmente são identificados, em média, 1.751 CVEs distintos. O conjunto de CVEs identificados possui pouca variação: ao todo o Shodan identificou 1.833 vulnerabilidades distintas no período (até fevereiro de 2022, o NVD possuía 141 mil CVEs do tipo `AV:N`, ou seja, com vulnerabilidades exploráveis a partir da rede). Apesar do baixo valor, acreditamos que esse número é subestimado, pois existem vulnerabilidades em sistemas de *back-end* que não são acessíveis remotamente (pela rede), e existem vulnerabilidades que o Shodan não é capaz de detectar. A quantidade média de endereços IP com pelo menos uma vulnerabilidade identificada pelo Shodan por semana é 2.629 (não mostrado na figura) e também permanece estável no período.

Acreditamos que o número estável de vulnerabilidades está relacionado com a falta de atualizações e correções de segurança por parte dos usuários domésticos e de

organizações. Por exemplo, uma vulnerabilidade frequente em todo o período da amostra é a de código CVE-2017-7269, de grau crítico (9,8 no CVSS v3), que permite a execução remota de código arbitrário em servidores rodando o Internet Information Services (IIS) versão 6.0. Para esse CVE, foram encontrados 385 endereços IP com pelo menos um registro no qual o Shodan testou e verificou a existência da vulnerabilidade.

A tabela 2 expande a caracterização da figura 3 apresentando a criticidade das vulnerabilidades. Utilizamos o sistema CVSS v2 para a classificação, pois muitos CVEs ainda não tiveram seu *score* divulgado no CVSS v3. A tabela apresenta, para cada faixa de criticidade: (1) a fração de CVEs na base do *NVD* naquela faixa; (2) a fração de CVEs na base do Shodan naquela faixa; (3) a fração de *registros* do conjunto de dados cuja vulnerabilidade mais crítica está naquela faixa; e (4) a fração de IPs cuja vulnerabilidade mais crítica está naquela faixa. O resultado aponta que a maioria das vulnerabilidades (CVEs) identificadas pelo Shodan são de categoria média, que também possui o maior número de endereços IP distintos e registros. Esse resultado é consistente com a distribuição da criticidade dos CVEs no NVD.

### 4.2. Vulnerabilidades em Serviços PHP

Dentre as vulnerabilidades relacionadas na tabela 2, muitas ocorrem em serviços *web* como o Apache HTTP e o PHP. Essa prevalência pode ser explicada pela popularidade dessas aplicações. Muitas dessas vulnerabilidades foram registradas em 2017 e 2018, o que indica uma predominância de dispositivos executando *software* desatualizado. Para ter um panorama da situação, identificamos que as duas versões mais prevalentes de PHP são a versão 5, com 72,8% dos IPs executando PHP, e a versão 7, com 27,8%. Já um relatório de prevalência global de versões do PHP,[3] para o mesmo período de dados, aponta uma situação inversa: 64,6% utilizavam a versão 7 e apenas 34,9% na versão 5.

Essa desatualização dos serviços PHP é particularmente perigosa tendo em vista que a interrupção de serviço e execução remota de código são os tipos de vulnerabilidades mais frequentes em PHP.[4] Por exemplo, a vulnerabilidade CVE-2019-9023 é uma das vulnerabilidades de grau crítico (CVSS v3 9,8) mais comuns em nossos dados. Essa vulnerabilidade permite a um invasor remoto, não autenticado, explorar o sistema enviando uma expressão regular mal-intencionada para modificar arquivos no sistema ou interromper a disponibilidade do serviço. Mesmo depois de dois anos de registro dessa vulnerabilidade, ela ainda continua comum em dispositivos da rede brasileira.

### 4.3. Vulnerabilidades Críticas em Sistemas Autônomos (AS)

A figura 4 apresenta a distribuição da relação dos IPs identificados pelo menos uma vez com vulnerabilidades críticas pela quantidade de IPs observados para aquele AS. Os resultados para todos os 8.798 sistemas autônomos (ASes) que fazem parte da Internet brasileira são pessimistas, pois dispositivos com IP dinâmico inflacionam a fração de endereços IPs com vulnerabilidades críticas. Cerca de 75% dos ASes possuem menos de 6% de IPs com vulnerabilidades críticas. Porém, um resultado alarmante é que cerca de 1% dos ASes identificados no Shodan possuem pelo menos 69% de IPs com vulnerabilidades críticas. Mesmo considerando a existência de endereços IPs dinâmicos nesses ASes, esse número ainda seria alto. Esta alta fração de dispositivos vulneráveis pode comprometer não apenas os AS hospedeiros, mas também outras redes na Internet.

[3] https://w3techs.com/technologies/history_details/pl-php

[4] https://www.cvedetails.com/product/128/PHP-PHP.html?vendor_id=74

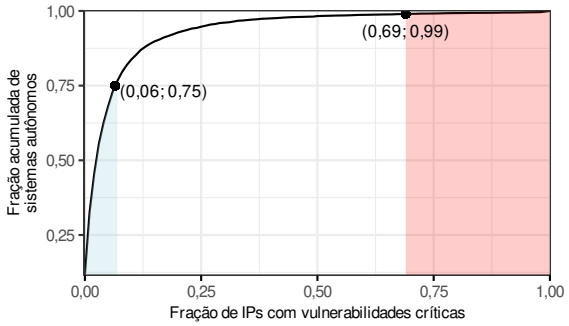


**Figura 4. Prevalência de IPs com vulnerabilidades em um AS.**

**Tabela 3. Relação dos dois ASes com maior número de vulnerabilidades críticas.**

| ATRIBUTO | AS $X$ | AS $Y$ |
|---|---|---|
| CVES | 39.859 | 10.240 |
| CVE DISTINTOS | 59 | 10 |
| % IPS | 99,9 | 100 |
| | (de 1024) | (de 1024) |
| % BANNERS | 78,2 | 82,4 |
| | (de 12.196) | (de 45.996) |
| PHP | 5.5.8 | 5.3.16 |

A tabela 3 detalha o resultado obtido para dois ASes (anonimizados para protegê-los de retaliação) com alta fração de IPs com vulnerabilidades críticas e com grande número de vulnerabilidades críticas. Esses condicionantes nos permitem ignorar ASes com alta fração de IPs vulneráveis mas que controlam poucos IPs, bem como ASes com grande número de vulnerabilidades mas distribuídas em um espaço de endereçamento muito vasto. Os dois ASes apresentados na tabela 3 têm pequeno porte e são controlados por organizações brasileiras. Os cinco CVEs mais frequentes em ambos os ASes são relacionados a falhas em versões antigas do PHP, resultado que corrobora o apresentado na seção anterior. Todos os registros com serviços PHP em cada AS indicam a mesma versão do PHP, como mostrado na tabela 3. É interessante observar que ambas as versões já estão obsoletas. Para a linha de versionamento 5.3.X, a última atualização do PHP Group, órgão desenvolvedor do PHP, foi a 5.3.29, em 2014; já para a linha 5.5.X, a última atualização foi a 5.5.38, em 2015. Mesmo com o suporte de longo prazo (LTS) de sistemas operacionais como o Debian, as atualizações são realizadas apenas para as versões mais recentes do pacote. No caso do Debian, por exemplo, a última atualização do pacote php5 foi em 2020 e apenas para a versão 5.6.40.[5]

**AS $X$.** A maioria (99,9%) dos endereços IP controlados pelo AS $X$ possuem vulnerabilidades críticas; além disso, o número de vulnerabilidades críticas por IP é muito alto, com quase 40 mil vulnerabilidades em um espaço de endereçamento de 1024 IPs. Ao todo, para esse AS, foram identificados 59 CVEs distintos em cerca de 12 mil registros. Esses números são alarmantes. Mesmo considerando que alguns registros podem ser derivados de IPs dinâmicos, a alta porcentagem de registros com vulnerabilidades (78,2%) reforça a conclusão de que a maior parte dos dispositivos estão vulneráveis.

Realizando uma análise mais detalhada, encontramos que 77,7% dos registros com vulnerabilidades identificadas pelo Shodan foram derivados de dispositivos Mimosa Network B5c, roteadores de sinal de longa distância via rádio, nas portas 7777 e 9500. Para os registros coletados a partir de outras portas, apenas 1,1% apresentaram vulnerabilidades, todos derivados de sondagem do *website* do provedor. Estes resultados indicam que o AS é utilizado para prover serviço de acesso à Internet via rádio utilizando pontos de acesso Mimosa B5c. A prevalência de registros com vulnerabilidades indica que os pontos de acesso executam serviços PHP obsoletos e devem estar implantados nas premissas de vários clientes do provedor.

[5] https://tracker.debian.org/pkg/php5

Apesar de não termos certeza sobre o real estado de vulnerabilidades desses dispositivos visto que o Shodan identifica os CVEs a partir de metadados e que neste projeto não realizarmos varreduras ativas por princípio, alguns indícios corroboram a conclusão de que o AS está vulnerável: (1) a versão PHP não recebe suporte há mais de 6 anos; (2) como os dispositivos são embarcados, atualizações de *firmware* pelos fabricantes costumam ser lentas (quando existem) e dispendiosas de implantar por parte dos provedores, podendo requerer a locomoção de técnicos a áreas de difícil acesso; e (3) a última menção de uma atualização do PHP na página de atualizações do fabricante é de 2016.[6]

**AS $Y$.** No AS $Y$ 82% dos registros possuem vulnerabilidades em serviços relacionados ao PHP e ao Apache HTTP. Para esse AS, os registros coletados buscavam mapear serviços em oito portas distintas, mas segundo os registros, todos os IPs desse AS executam o sistema Debian e possuem vulnerabilidades nas portas 80, 8001, e 8090. Quando os registros são coletados para essas três portas as informações são semelhantes e todas relacionadas a um serviço de hospedagem (anonimizado) da organização responsável pelo AS. Esse resultado é um indício que o provedor de hospedagem encaminha todas essas portas para um servidor com seu *website*. Apesar da fração de IPs vulneráveis estar inflacionada devido ao redirecionamento das portas, o dispositivo vulnerável provavelmente tem capacidade computacional e conectividade de rede superiores à de dispositivos embarcados como no AS $X$, e pode executar outros serviços críticos para o AS.

**Implicações.** Acreditamos que esse tipo de análise pode identificar cenários de vulnerabilidade e permitir a construção de documentação para facilitar as correções dessas vulnerabilidades. Por exemplo, o AS $X$ poderia implantar um *firewall* em sua rede para bloquear acessos externos aos pontos de acesso B5c, enquanto o AS $Y$ deveria considerar atualizar o seu servidor *web*. Notamos ainda que vários outros ASes podem ser vulneráveis. Escolhemos apenas dois ASes devido a restrições de espaço, mas um estudo aprofundado poderia considerar outros ASes e vulnerabilidades de criticidade média ou baixa. A notificação e o acompanhamento dos ASes também é um tópico de interesse.

### 4.4. Protocolos SSL e TLS

Desde a identificação da vulnerabilidade CVE-2014-0160 em 2014, apelidada de *Heartbleed*, vários estudos foram realizados sobre vulnerabilidades envolvendo certificados digitais [Al-Alami et al. 2017, Durumeric et al. 2015]. Desde então, muitas outras vulnerabilidades foram identificadas nos protocolos SSL e TLS, várias com grau elevado de criticidade. O gráfico da figura 5 apresenta a relação de alguns dos CVEs relacionados a vulnerabilidades em protocolos SSL e TLS identificados na base do Shodan. Dos 3.658 CVEs rotulados na base de vulnerabilidades do NIST/NVD relacionados a falhas no SSL/TLS, 644 possuem grau alto ou crítico segundo o sistema de ranqueamento CVSS v3, sendo 31 identificados na nossa base de dados. Desses 31, apresentamos na figura 5 uma série temporal do número de endereços IP associados aos 6 CVEs mais prevalentes.

A quantidade de dispositivos vulneráveis é estável, com indícios de que poucos foram atualizados durante o período: apenas 2% dos endereços IP sondados que apresentaram os CVEs considerados foram observados em um momento posterior sem a vulnerabilidade. Esse é um resultado otimista, pois pode conter falsos positivos induzidos por dispositivos com endereços IP dinâmicos. Esse resultado indica que muitos dispositivos

[6] http://backhaul.help.mimosa.co/backhaul-firmware-release-notes

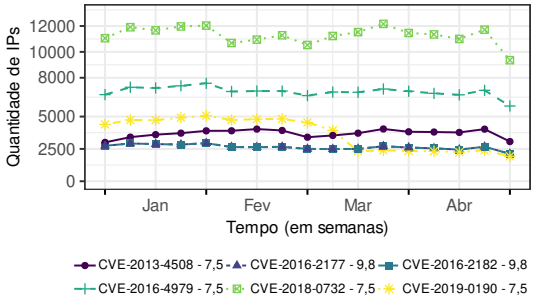


**Figura 5. Vulnerabilidades em protocolos SSL/TLS.**

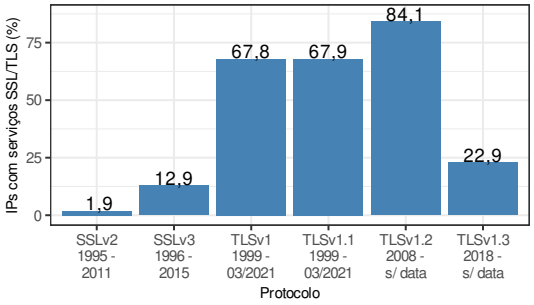


**Figura 6. Versões de protocolos SSL/TLS.**

vulneráveis a falhas identificadas há mais de 5 anos continuam vulneráveis sem nenhuma prática de correção, o que é alarmante. Dessa lista, vulnerabilidades como a CVE-2018-0732, CVE-2016-2177 e CVE-2016-2182 permitem o lançamento de ataques de negação de serviço (DoS). Outras, como a CVE-2016-4979, permitem que invasores remotos ignorem restrições de acesso. Mesmo vulnerabilidades que foram bastante discutidas na época de identificação, como o *Heartbleed*, continuam presentes: No nosso conjunto de dados, o *Heartbleed* é o 21º CVE mais frequente entre os de níveis alto ou crítico.

Muitas das vulnerabilidades citadas anteriormente são falhas de implementações em pacotes como o OpenSSL. Outras são vulnerabilidades intrínsecas ao protocolo. Certificados digitais devem utilizar protocolos mais novos para se manterem atualizados e seguros. A figura 6 apresenta a relação das versões suportadas dos protocolos SSL/TLS para os registros que sondam esses serviços. Em média, os dispositivos avaliados suportam de 2 a 3 versões de protocolos, sendo atualmente o TLS v1.2 o mais frequentemente suportado (aproximadamente 84% dos dispositivos). No entanto, versões como TLS v1 e TLS v1.1 continuam sendo bastantes frequentes, mesmo com a recente obsolescência destes protocolos anunciada em março de 2021 pelo Internet Engineering Task Force (IETF).[7] Além disso, não foi percebida nenhuma mudança de comportamento na base de dados sobre a migração dos protocolos TLS v1 e TLS v1.1 para o protocolo TLS v1.2 para adequação às novas diretrizes. Procolos SSL v2 e v3, embora suportados menos frequentemente, ainda continuam sendo utilizados mesmo depois de 6 anos de obsolescência.

Outro ponto de interesse avaliado sobre os certificados digitais são os *hashes* de validação das assinaturas. A tabela 4 apresenta a prevalência de funções de *hash* utilizadas em certificados em dispositivos nas redes brasileiras. O algoritmo SHA-256 atualmente é o padrão mais utilizado (88% de todos os certificados). Cerca de 10% dos certificados continuam a utilizar os padrões MD5 e o SHA-1, mesmo já obsoletos. No conjunto de dados, 7,32% dos certificados em utilização estão expirados. Como esperado, em uma comparação entre distribuições das funções de *hash* de todos os certificados e dos certificados expirados observamos que os certificados expirados têm um viés para funções de *hash* obsoletas. Por fim, a última linha indica que uma fração mais expressiva dos certificados utilizando funções de *hash* obsoletas estão expirados.

**Discussão.** Esses resultados indicam que vulnerabilidades relacionadas a falhas nos protocolos SSL ou TLS, incluindo utilização de versões obsoletas, são frequentes em redes brasileiras. Aliado a isso, muitas organizações continuam a utilizar certificados inse-

[7] https://datatracker.ietf.org/doc/rfc8996/

**Tabela 4. Relação dos algoritmos Hash utilizados nos certificados digitais.**

| Função | | MD5 | SHA-1 | SHA-256 | SHA-384 | SHA-512 |
|---|---|---|---|---|---|---|
| Certificados (%) | Todos | 0,76 | 9,43 | 88,04 | 1,30 | 0,47 |
| | Expirados | 0,47 | 2,36 | 4,41 | 0,03 | 0,01 |
| Fração de Expirados (%) | | 62,18 | 25,41 | 5,00 | 2,56 | 2,74 |

guros ou expirados. Essas vulnerabilidades expõem muitos serviços *web* a ataques de personificação, possibilitando, por exemplo, roubo e vazamento de dados. Se considerarmos que algumas dessas páginas podem estar executando alguma aplicação crítica (como bases de dados ou comércio eletrônico), o impacto pode ser ainda mais alto. Certificados expirados podem também causar interrupções de serviço e permitir outros tipos de ataques como *man-in-the-middle*. Por exemplo, em 2013, o Microsoft Azure passou por uma interrupção mundial, ficando fora do ar por horas, devido a um certificado expirado.[8]

## 5. Trabalhos Relacionados

Serviços como o Shodan permitem uma série de análises sobre o comportamento da Internet. Existem trabalhos que atuam na identificação de vulnerabilidades [Genge e Enăchescu 2016]. Já outros propõem métodos para análises gerais da Internet, como a classificação de fabricantes de dispositivos de rede ou identificação de serviços em execuções a partir de portas não padrão [Holland et al. 2020, Izhikevich et al. 2021].

Similar ao nosso trabalho, [Al-Alami et al. 2017] apresenta resultados de análises de dispositivos e suas potenciais vulnerabilidades na Jordânia usando o Shodan em maio de 2017. Embora apresente resultados interessantes como a identificação de certificados expirados ou dispositivos com *Heartbleed*, entre outras vulnerabilidades, a amostra se limitou a aproximadamente 41 mil registros, um número significativamente inferior ao que analisamos neste trabalho. Nosso trabalho também propõe novos mecanismos e fontes de dados para refinamento e complementação das informações do Shodan.

Sistemas de supervisão e aquisição de dados (do inglês, *Supervisory Control and Data Acquisition* ou SCADA) são sistemas geralmente utilizados para monitorar e supervisionar as variáveis e os dispositivos de processos industriais. Dispositivos desse tipo podem apresentar risco não apenas à segurança cibernética, mas também à instalação física no qual se encontra. Pelo grau de periculosidade envolvido em dispositivos desse tipo, muitos trabalhos foram realizados para identificar e analisar dispositivos SCADA [Samtani et al. 2018, Hasselquist et al. 2019]. Por exemplo, [Samtani et al. 2018] utiliza uma abordagem baseada em mineração de texto na analise do conteúdo de cada registro para classificação dos dispositivos SCADA. Embora nosso trabalho não tenha focado nesse tópico, a metodologia desenvolvida pode ajudar, principalmente pelo aumento de escalabilidade alcançado. Por exemplo, trabalhos anteriores utilizando Python exigem máquinas com muita memória para suportar o grande volume de dados e código específico para paralelização das tarefas de processamento [Samtani et al. 2018], que é um desafio tratado de forma transparente na nossa proposta.

## 6. Conclusão e Trabalhos Futuros

O número de ataques a dispositivos conectados à Internet vem crescendo a cada ano. É imperativo que as organizações monitorem suas vulnerabilidades e atualizem seus siste-

[8] https://www.computerworld.com/article/2495453/microsoft-s-azure-service-hit-by-expired-ssl-certificate.html

mas. Um ataque pode não apenas causar vazamento de dados como também utilizar os recursos vulneráveis em um ataque de DDoS. Nesse contexto, este trabalho apresenta uma caracterização de vulnerabilidades encontradas na Internet brasileira. Encontramos diversas vulnerabilidades críticas e um grande número de organizações que utilizam *software* desatualizado, em alguns casos sem suporte há mais de cinco anos. Além disso, identificamos serviços reais altamente vulneráveis que podem servir de base para construção de novas práticas com vistas a melhorar a segurança de redes no Brasil. Esperamos que a metodologia desenvolvida para o processamento, armazenamento e refinamento de dados auxilie pesquisadores e operadores de rede não apenas na melhoria da segurança, mas também em outros contextos. Os resultados aqui apresentados podem ser utilizados pelo CERT.br em ações para tentar conscientizar os operadores de ASes nacionais.

Trabalhos futuros incluem novas análises e um maior período de dados. Além disso, a metodologia pode ser estendida para, por exemplo, integração de registros coletados pelo Shodan com dados do whois, como a data de registro de um domínio ou o intervalo de IPs que a organização controla. Tais informações podem ser úteis no preenchimento de dados ausentes e também na identificação de dispositivos vulneráveis.

## Agradecimentos

Este trabalho foi parcialmente financiado pelo NIC.br, RNP/CTIC (2955), FAPEMIG, CNPq, CAPES, MASWEB, INCT-Cyber e EUBra-Atmosphere.

## Referências

[Al-Alami et al. 2017] Al-Alami, H., Hadi, A., e Al-Bahadili, H. (2017). Vulnerability scanning of iot devices in jordan using shodan. In *Int. Conf. on the Applications of Information Technology in Developing Renewable Energy Processes Systems (IT-DREPS)*.

[Durumeric et al. 2015] Durumeric, Z. et al. (2015). A Search Engine Backed by Internet-Wide Scanning. In *Proc. of ACM SIGSAC Conf. on Computer and Comm. Security*.

[Genge e Enăchescu 2016] Genge, B. e Enăchescu, C. (2016). ShoVAT: Shodan-based vulnerability assessment tool for Internet-facing services. *Security and Communication Networks*, 9(15):2696–2714.

[Hasselquist et al. 2019] Hasselquist, D., Rawat, A., e Gurtov, A. (2019). Trends and Detection Avoidance of Internet-Connected Industrial Control Systems. *IEEE Access*, 7:155504–155512.

[Holland et al. 2020] Holland, J. et al. (2020). Classifying Network Vendors at Internet Scale. *Computing Research Repository*, abs/2006.13086.

[Izhikevich et al. 2021] Izhikevich, L., Teixeira, R., e Durumeric, Z. (2021). LZR: Identifying Unexpected Internet Services. In *USENIX Security*.

[Mousavi et al. 2020] Mousavi, S. H., Khansari, M., e Rahmani, R. (2020). A fully scalable big data framework for Botnet detection based on network traffic analysis. *Information Sciences*, 512:629–640.

[Samtani et al. 2018] Samtani, S. et al. (2018). Identifying SCADA Systems and Their Vulnerabilities on the Internet of Things: A Text-Mining Approach. *IEEE Intelligent Systems*, 33(2):63–73.