UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Aplicação de Modelos Relacionais Baseados em Grafos para Detecção de Anomalias em Arquivos de Log do Web Apache

Flávia Oliveira Santos de Sá Lisboa

Trabalho de Conclusão de Curso
MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Aplicação de Modelos Relacionais Baseados em Grafos para Detecção de Anomalias em Arquivos de Log do Web Apache

Flávia Oliveira Santos de Sá Lisboa

USP - São Carlos

2025

Flávia Oliveira Santos de Sá Lisboa

Aplicação de Modelos Relacionais Baseados em Grafos para Detecção de Anomalias em Arquivos de Log do Web Apache

Trabalho de conclusão de curso apresentado ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial.

Orientador: Prof. Dr. Alneu de Andrade Lopes.

USP - São Carlos

2025

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

L769a

Lisboa, Flávia Oliveira Santos de Sá
Aplicação de Modelos Relacionais Baseados em Grafos para Detecção de Anomalias em Arquivos de Log do Web Apache / Flávia Oliveira Santos de Sá Lisboa; orientador Alneu de Andrade Lopes. – São Carlos, 2025.
79 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2025.

1. Detecção de Anomalias. 2. Grafos. 3. Redes Complexas. 4. Aprendizado de Máquina. 5. Servidor Web Apache. I. Lopes, Alneu de Andrade, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

DEDICATÓRIA

Aos pilares da minha vida: aos meus pais Josmal e Maria Nazaré, ao meu grande amor Rodrigo, aos meus filhos Árion e Pietra.

AGRADECIMENTOS

Ao meu orientador, Prof. Dr. Alneu de Andrade Lopes, pelas reuniões, discussões e questionamentos críticos que me auxiliaram no desenvolvimento desse trabalho.

Ao meu parceiro de trabalho há vinte e oito anos, Luciano Joioso, que me acompanhou mais uma vez compartilhando essa jornada de estudos fazendo juntos esse curso de MBA.

À Coordenação do MBA em IA e Big Data pela concessão da bolsa de estudos como servidora da USP, contribuindo com esse diferencial para o aperfeiçoamento técnico dos servidores técnicos e administrativos da USP.

Às ferramentas de IA utilizadas no desenvolvimento desse trabalho, auxiliando na busca de ideias e, por vezes, orientando os caminhos para a construção do trabalho.

EPÍGRAFE

Nossas dúvidas são traidoras e nos fazem perder o que, com frequência, poderíamos ganhar, por simples medo de arriscar. (*Adaptação de um trecho da peça "Measure for Measure"*).

William Shakespeare

RESUMO

LISBOA, F. O. S. S. **Aplicação de Modelos Relacionais Baseados em Grafos para Detecção de Anomalias em Arquivos de Log do Web Apache**. 2025. 79p. Monografia (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

Diante do crescente avanço dos ataques cibernéticos, a análise eficiente de registros de acesso em servidores web é fundamental para a detecção de incidentes de segurança. Contudo, o grande volume de dados torna inviável sua verificação manual, exigindo soluções automatizadas e eficazes que apoiem o trabalho do administrador de sistemas. Este trabalho tem como objetivo investigar a aplicação de modelos relacionais baseados em grafos para a detecção de anomalias em arquivos de log de acesso do servidor Apache, utilizados no ambiente institucional do Instituto de Física de São Carlos da Universidade de São Paulo (IFSC/USP). Para isso, foram coletados registros reais de acesso dos servidores web Apache e, a partir deles, desenvolvidos diferentes modelos de grafos, incluindo representações de similaridade, bipartidas, tripartidas, temporais e de comunidades. Os experimentos compararam dias de acessos normais aos servidores com um caso real de ataque por injeção de SQL (SQLi), revelando padrões estruturais distintos em cenários de invasão, como concentrações anormais de códigos de erro HTTP, tentativas sucessivas de exploração de URLs e agrupamentos de endereços IP com comportamento semelhante. Os resultados indicam que a modelagem gráfica utilizando grafos é capaz de complementar técnicas tradicionais de análise exploratória, fornecendo uma camada relacional importante que torna visíveis padrões de comportamento de difícil percepção apenas com métodos estatísticos. A proposta mostrou-se viável, de baixo custo computacional e aplicável em ambientes reais de administração de servidores web, além de apresentar potencial para monitoramento em tempo real por meio de painéis interativos.

Palavras-chave: 1. Detecção de Anomalias. 2. Grafos. 3. Redes Complexas. 4. Aprendizado de Máquina. 5. Servidor Web Apache.

ABSTRACT

LISBOA, F. O. S. S. **Application of Graph-Based Relational Models for Anomaly Detection in Apache Web Log Files.** 2025. 79p. Monograph (MBA in Artificial Intelligence and Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

In the current scenario of increasing complexity of cyberattacks, the efficient analysis of web server access logs is essential for detecting security incidents. However, the large volume of data makes manual verification unfeasible, requiring automated and effective solutions to support the work of system administrators. This study aims to investigate the application of graph-based relational models for anomaly detection in Apache web server access log files, within the institutional environment of the Instituto de Física de São Carlos from the Universidade de São Paulo (IFSC/USP). For this purpose, real access records from the web servers were collected and, from them, different graph models were developed, including similarity, bipartite, tripartite, temporal, and community representations. The experiments compared days of normal operation of the servers with a real case of SQL injection attack (SQLi), revealing distinct structural patterns in intrusion scenarios, such as abnormal concentrations of HTTP error codes, successive attempts to exploit URLs, and clusters of IP addresses with similar behavior. The results indicate that graph modeling can complement traditional exploratory analysis techniques by providing an important relational layer that makes behavioral patterns visible, which would otherwise be difficult to detect using statistical methods alone. The proposed approach proved feasible, of low computational cost, and applicable in real web server administration environments, while also showing potential for real-time monitoring through interactive dashboards.

Keywords: 1. Anomaly Detection. 2. Graphs. 3. Complex Networks. 4. Machine Learning. 5. Apache Web Server.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Na era da informação digital, a segurança de sistemas computacionais tornou-se um dos maiores desafios tecnológicos. Com a integração crescente de tecnologias em todos os setores, a exposição a ameaças cibernéticas se intensificou (Shaukat et al., 2020), exigindo soluções cada vez mais sofisticadas para a identificação e mitigação de vulnerabilidades.

Nesse cenário, a Inteligência Artificial (IA) e o Aprendizado de Máquina (AM) têm se destacado como ferramentas fundamentais. Essas tecnologias permitem automatizar a análise de grandes volumes de dados e identificar padrões que indicam comportamentos anômalos ou maliciosos, oferecendo maior eficiência no monitoramento e resposta a incidentes de segurança.

Técnicas de AM têm se destacado como ferramentas fundamentais para a automação da detecção de incidentes de segurança (Bahassi et al., 2022). Esses métodos permitem não apenas identificar anomalias, mas também prever comportamentos maliciosos com base em padrões históricos (Shaukat et al., 2020). Ao combinar dados extraídos de arquivos de log com modelos computacionais avançados, é possível desenvolver sistemas capazes de monitorar, analisar e responder a ameaças em tempo real. A integração dessas tecnologias de AM no monitoramento de segurança se apresenta, assim, como uma solução promissora para enfrentar tais desafios impostos pela era digital.

Entre as principais fontes de dados para esse tipo de análise estão os arquivos de log, que registram detalhadamente as atividades realizadas nos sistemas. Diante da inviabilidade de uma análise manual em ambientes de grande escala, a aplicação de técnicas de IA e AM nesses registros possibilita extrair conhecimento relevante em tempo real, apoiando administradores de sistemas na detecção e mitigação de ameaças de forma proativa.

1.1 Motivação

A administração e a segurança de servidores Web Apache em ambiente Linux, amplamente empregados na hospedagem de sites e sistemas institucionais, demandam monitoramento contínuo e análise eficiente dos registros de log para a identificação de ameaças e vulnerabilidades. Este trabalho adota como estudo de caso os servidores do Instituto de Física de São Carlos da Universidade de São Paulo (IFSC/USP), embora a metodologia proposta seja aplicável a qualquer ambiente web. Os arquivos de log, que documentam detalhadamente as atividades de acesso aos sistemas, são fundamentais para detectar comportamentos anômalos e possíveis ataques. Contudo, a análise manual desses dados é inviável em ambientes de grande

volume, tornando essencial o uso de ferramentas automatizadas para agilizar a detecção e resposta a incidentes de segurança.

Como ilustração aos fatores de motivação a este trabalho são apresentadas situações ocorridas de invasões e ataques sofridos por organizações públicas e privadas no decorrer do período de desenvolvimento do trabalho. Ressaltando cada vez mais a importância e o caráter essencial que ferramentas de detecção de incidentes de segurança digital têm atualmente.

- Junho de 2024: servidores Web Apache do IFSC sofreram ataque e comprometimento de sites. Redirecionamento da busca de sites IFSC no Google para jogos de apostas do "tigrinho".
- Março de 2025: ataques aos serviços da USP, comprometendo o acesso aos sistemas computacionais (Olhar Digital, 2025a), inclusive dificultando o acesso dos alunos à plataforma dos cursos no MBA. Outros ataques de negação de serviço (DDoS) atuais em grandes empresas e instituições:
    - 10/03/2025: Rede social X (OGlobo Tecnologia, 2025).
    - 15/03/2025: Aeroporto de Guarulhos (Olhar Digital, 2025b).
    - 19/03/2025: Site da FAB (Security Report, 2025).
    - 25/03/2025: Site e sistemas do TSE (MPMT, 2025).
- Maio de 2025: servidor Web Apache do IFSC sofre ataque de injeção de SQL, comprometendo conteúdo no banco de dados de sistemas institucionais.

Nesse contexto, torna-se essencial a adoção de soluções baseadas em IA para automatizar a detecção de atividades maliciosas e acessos indevidos. Além de reduzir a carga de trabalho do administrador, essas ferramentas podem acelerar a identificação de ameaças e permitir uma resposta rápida e eficaz. Diversas técnicas de AM já são amplamente empregadas em aplicações de segurança que se baseiam especificamente na análise de arquivos de log, demonstrando sua eficácia nessa área de aplicação de segurança computacional.

1.2 Objetivos

O objetivo principal deste Trabalho de Conclusão de Curso (TCC) foi investigar e aplicar modelos baseados em grafos para analisar arquivos de log de acesso gerados diariamente pelos servidores Web Apache do IFSC. A proposta buscou identificar padrões presentes nesses registros para detectar determinados tipos de ataques web, como tentativas de injeção de SQL, varredura de diretórios, ataques de força bruta e exploração de vulnerabilidades em scripts.

As etapas definidas para execução desta proposta foram:

1) Coleta e pré-processamento dos dados: Selecionar e coletar os logs apropriados dos servidores Web Apache, realizar a limpeza e reestruturação dos dados para torná-los adequados ao uso pelos algoritmos de construção de grafos.
2) Transformação e modelagem dos dados: Extrair atributos relevantes dos logs e transformá-los no formato ideal para a modelagem dos grafos.
3) Aplicação de técnicas de AM: Investigar modelos de Aprendizado Relacional Baseado em Grafos para detectar padrões que indiquem possíveis ataques web nos arquivos de log.
4) Análise e interpretação: Identificar padrões presentes nos registros de log para detectar indícios de atividades maliciosas, determinados tipos de ataques web como, por exemplo, alta frequência de IPs que acessam múltiplas URLs (possível varredura para ataques de injeção de SQL ou de força bruta), URLs acessadas por diversos IPs com cargas maliciosas semelhantes (ataques coordenados de negação de serviço), subconjuntos densamente conectados indicando comunidades de ataque ou atividades repetitivas suspeitas (acesso indevido ou varreduras para descoberta de vulnerabilidades).
5) Validação: validar os modelos com dados de teste para modelos de grafos e ajustar parâmetros conforme necessário.

O objetivo final deste trabalho é apoiar o administrador de sistemas na identificação de atividades suspeitas em servidores Web Apache, possibilitando respostas mais rápidas e eficazes a incidentes de segurança. Além disso, busca-se apresentar uma abordagem metodológica que possa ser replicada permitindo a aplicabilidade dos resultados em outros contextos.

1.3 Escopo do Trabalho

O escopo deste trabalho foi especificamente direcionado à análise de arquivos de log de acesso gerados por servidores Web Apache no formato padrão CLF (*Combined Log Format*) (W3C Standards, 2025). Esse formato é amplamente utilizado para registrar informações detalhadas sobre as requisições feitas ao servidor, incluindo dados como endereço IP do cliente de acesso, data e horário da requisição, método HTTP, URL acessada, status da resposta e tamanho da resposta. A escolha desse tipo de arquivo de log justifica-se por sua disponibilidade constante nos servidores Web do IFSC, bem como por sua relevância para a identificação de

padrões de acesso e detecção de possíveis anomalias. Os servidores analisados são responsáveis por hospedar diversos sites institucionais e de grupos de pesquisa do IFSC, proporcionando um contexto variado de acessos que favorece a investigação de padrões de comportamento e a aplicação de técnicas de AM.

1.4 Notas da Autora

Durante o desenvolvimento deste trabalho foi utilizado o ChatGPT4 (OpenAI, 2025) como ferramenta de apoio na codificação com Python e, também, como auxiliar na redação do texto realizando revisões gramaticais e de coesão textual.

1.5 Organização do Texto

No Capítulo 2 é apresentada uma revisão teórica que abrange a revisão bibliográfica sobre aplicação de aprendizado de máquina em detecção de anomalias e o referencial teórico sobre grafos e tipos de ataques web. O Capítulo 3 apresenta a sistemática da coleta de dados e uma breve análise exploratória dos dados coletados. No Capítulo 4 é apresentada a metodologia utilizada neste trabalho e no Capítulo 5 são apresentados os detalhes sobre os experimentos conduzidos. Por fim, o Capítulo 6 discute os resultados e conclusões deste trabalho.

2 REVISÃO TEÓRICA

A revisão bibliográfica desenvolvida para este trabalho foi estruturada com base na interseção entre dois temas centrais: os tipos de ataques (incidentes de segurança) que afetam servidores web e a aplicação de técnicas de AM para detecção de anomalias em registros de log.

O objetivo principal do referencial teórico foi identificar as práticas mais utilizadas que possam ser aplicadas ao contexto específico deste trabalho, contribuindo para a implementação de uma solução baseada em modelos de grafos para detecção de anomalias.

2.1 Revisão Bibliográfica

O autor Baranwal (2012) explora os dois tipos mais comuns de ameaças à segurança em aplicações web: a injeção de SQL (SQLi) e o *Cross-Site Scripting* (XSS). Em seu artigo são apresentadas as características desses ataques, seus impactos e algumas das principais abordagens para detecção e prevenção.

Em Hoang (2021), é proposto um modelo baseado em aprendizado de máquina para detectar ataques web comuns, como SQLi, XSS, varredura de diretórios e injeção de comandos. Utilizando logs de servidor web como entrada, o modelo analisa padrões de acesso para identificar atividades maliciosas de forma automatizada.

O LogPAI é um projeto de código aberto que visa a construção de uma plataforma baseada em IA para análise automatizada de logs (LogPAI, 2025). O projeto disponibiliza conjuntos de dados públicos e ferramentas voltadas para a pesquisa em análise de logs. Dentre os principais recursos oferecidos pelo LogPAI, destaca-se o LogHub (Zhu et al., 2023), um repositório que contém diversos conjuntos de dados de logs coletados de sistemas reais. São disponibilizadas, também, ferramentas para experimentação com diferentes técnicas de pré-processamento de logs (He, P. et al., 2016) (He, P. et al., 2017) e ferramentas para pesquisas em detecção de anomalias com aplicação de algoritmos de AM (He, P. et al., 2018) e aprendizado profundo (He, S. et al., 2016).

O modelo apresentado em Jyothy et al. (2024) utiliza o algoritmo de aprendizado não supervisionado k-*Means* para agrupar registros de log em *clusters*, separando padrões normais de possíveis anomalias. O artigo destaca a importância dos arquivos de log como fonte primária de informações para detecção de comportamentos suspeitos e enfatiza a necessidade de um pré-processamento adequado. O estudo define como critério para detecção de anomalias que

registros alocados em agrupamentos isolados ou com baixa similaridade em relação ao comportamento típico podem ser classificados como potenciais ameaças.

O artigo de Seyyar; Çatak; Gül (2018) ressalta a relevância da análise de logs HTTP do Apache para a segurança de servidores web, demonstrando como a detecção de padrões de escaneamento malicioso pode prevenir ataques direcionados. O artigo propõe uma abordagem baseada na análise dos padrões de requisição para identificar varreduras automatizadas realizadas por atacantes antes da execução de ataques mais avançados, como injeção de SQL e ataques XSS. Os autores identificam que varreduras frequentemente apresentam um número elevado de requisições em curto período, acesso a URLs específicas e não convencionais, e tentativas repetidas de acessar páginas restritas ou inexistentes.

Nos artigos de Vaarandi e Pihelgas (2014) e (2015), os autores exploram a importância dos logs como fontes valiosas de dados para o monitoramento e a segurança de sistemas. Em ambos os estudos, são propostas maneiras de automatizar a extração de informações úteis de grandes volumes de logs.

No artigo de Akoglu; Tong; Koutra (2015), os autores apresentam um panorama sobre a detecção de anomalias em grafos. Na análise de grafos, a detecção de anomalias envolve a identificação de objetos gráficos (nós e arestas) que apresentam um comportamento raro ou significativamente diferente da maioria dos elementos de referência no grafo. Para isso, cada objeto pode receber uma pontuação de anomalias (um peso indicativo do grau de discrepância em relação ao valor esperado), sendo classificado como anômalo caso ultrapasse um determinado limiar. As anomalias podem se manifestar de diversas formas, como padrões inesperados de conexões, estruturas incomuns ou comportamentos estatisticamente improváveis.

O artigo de Saket, Pandey e Singh (2024) destaca a aplicação de técnicas baseadas em teoria dos grafos na análise de logs web, e enfatiza que os arquivos de log de servidores web possuem uma estrutura complexa, composta por um conjunto de diferentes elementos, como endereços IP, URLs acessadas, data e horário (*timestamp*) e respostas do servidor. A teoria dos grafos permite modelar as interações entre esses elementos de forma estruturada, transformando os logs em redes onde os nós representam entidades (por exemplo, IP, URL) e as arestas representam conexões entre essas entidades (como repetição de acessos ou padrões suspeitos de acesso). Uma das vantagens apontadas no artigo para o uso de grafos é que permitem lidar de uma forma mais adequada com a natureza não estruturada e interconectada dos logs, possibilitando a identificação de possíveis acessos suspeitos.

Dentre os estudos analisados, destacam-se ainda os artigos de revisão ("*Survey*" e "*Review*") que oferecem uma visão abrangente sobre as técnicas de AM aplicadas à segurança e à detecção de anomalias (Bahassi et al., 2022; Bhanage, Pawar e Kotecha, 2021; Chandola, Banerjee e Kumar, 2009; Kilincer, Ertam e Sengur, 2021; Landauer et al., 2018; Shaukat et al., 2020). Esses artigos apresentam um levantamento detalhado dos algoritmos mais utilizados, discutindo suas vantagens, limitações e aplicações na área de segurança computacional.

Além disso, alguns estudos específicos mostraram-se particularmente interessantes e relevantes para este TCC, fornecendo abordagens que foram exploradas ao longo do desenvolvimento do trabalho. Destacam-se, nesse contexto, os trabalhos de Choraś e Kozik (2015), que exploram um modelo baseado em grafos e expressões regulares para a análise de logs web; e Hussein e Répás (2024), que investigam modelos supervisionados e não supervisionados para a detecção de anomalias em arquivos de log.

2.2 Revisão de Ferramentas Analisadoras de Logs

No artigo *"10 Best Apache Log Analyzers: Free & Paid Tools [2023 Comparison]"* (Sematext Blog, 2025) são elencadas dez ferramentas para análise de logs. Algumas delas foram exploradas com o objetivo de verificar se essas soluções já apresentavam algum tipo de funcionalidade com ênfase em modelagem relacional dos logs. Embora as ferramentas já consolidadas para análise de logs Apache ofereçam recursos avançados de relatórios, essas soluções se concentram principalmente na geração de estatísticas agregadas – como quantidade de requisições, distribuição de códigos de resposta HTTP e identificação de IPs mais ativos – apresentadas em tabelas, gráficos de barras e histogramas. Apesar de algumas incorporarem elementos visuais, como *heatmaps* e *timelines*, não enfatizam a representação relacional entre os dados e não oferecem mecanismos nativos para modelar conexões entre URLs acessadas, códigos de resposta HTTP e categorias de recursos, como é possível explorar com o uso de grafos.

Ao invés de apenas contagens e resumos estatísticos, a aplicação de grafos permite representar a estrutura de como os eventos se conectam entre si. A proposta deste trabalho diferencia-se por utilizar essa abordagem baseada em teoria de grafos para representar e explorar visualmente essas relações, na busca por identificar padrões, agrupamentos e comportamentos anômalos que podem não ser tão evidentes em visualizações tradicionais.

Ainda que algumas dessas ferramentas de análise de log sejam gratuitas, a maioria são soluções pagas ou possuem limitações importantes nas versões gratuitas, não sendo

integralmente de código aberto e exigindo, em alguns casos, infraestrutura robusta para operação. Por outro lado, uma solução baseada em bibliotecas livres e amplamente utilizadas – como NetworkX, Matplotlib, Plotly e Pandas – pode ser mais acessível, personalizável e adaptada ao contexto específico de cada servidor web, além de permitir a exploração de novas possibilidades investigativas com grafos, cuja aplicabilidade foi avaliada neste trabalho.

2.3 Referencial Teórico

2.3.1 Padrão dos Logs de Acesso e Códigos de Resposta HTTP

O servidor Web Apache registra cada requisição recebida em arquivos de log, sendo o formato CLF o mais utilizado por padrão (Apache Logging, 2025). Esse formato reúne informações essenciais sobre cada acesso, permitindo análises detalhadas para detecção de incidentes. No Apache, o CLF pode ser configurado pela diretiva:

```
LogFormat %h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-agent}i\" combined
```

Cada linha do log no padrão CLF apresenta os seguintes campos:

1) `%h` – Endereço IP do cliente que fez a requisição (ex: 176.34.214.129).
2) `%l` – Identidade do cliente (geralmente um hífen, pois raramente é utilizada).
3) `%u` – Usuário autenticado pelo HTTP, se existir (normalmente hífen).
4) `%t` – Data e hora da requisição (ex: [11/Dec/2024:06:28:44 -0300]).
5) `"%r"` – Método HTTP, recurso solicitado e versão do protocolo (ex: "GET /portal-ifsc/ HTTP/1.1").
6) `%>s` – Código de resposta HTTP retornado pelo servidor (ex: 200).
7) `%b` – Tamanho do conteúdo retornado ao cliente, em bytes, sem cabeçalho.
8) `"%{Referer}i"` – Página de origem do acesso (referenciador).
9) `"%{User-agent}i"` – Informações do navegador ou agente utilizado pelo cliente (ex: Mozilla).

Abaixo exemplos de registro no arquivo de log em CLF:

```
«176.34.214.129 - - [11/Dec/2024:06:28:44 -0300] "GET /portal-ifsc/ HTTP/1.1" 200 44769 "-" "Mozilla/5.0"»
«::1 - - [10/Dec/2024:10:00:00 +0000] "GET /index.html HTTP/1.1" 200 2326 "-" "Mozilla/5.0"»
```

No primeiro exemplo, o IP externo acessou a página inicial `/portal-ifsc/` e recebeu um status 200 (requisição bem-sucedida). Já na segunda linha de exemplo, ::1 indica acesso local (IPv6), requisitando `/index.html` com resposta também 200.

O código de resposta HTTP (campo `%>s`) é fundamental na análise de logs, pois indica o resultado da requisição (Apache Logging, 2025). Os códigos são divididos em cinco classes principais: 1xx (Informacional), 2xx (Sucesso), 3xx (Redirecionamento), 4xx (Erro do cliente), 5xx (Erro do servidor). Os códigos 4xx e 5xx merecem atenção especial no contexto de segurança em servidores Web, pois podem indicar tentativas de acesso indevido ou falhas no servidor. Assim, a análise criteriosa desses códigos nos logs Apache é fundamental para identificar padrões de ataques, tentativas de exploração e outros incidentes de segurança. Entre os mais relevantes para análise de incidentes estão:

- 403 Forbidden – Acesso negado; comum em tentativas de explorar diretórios restritos.
- 404 Not Found – Recurso inexistente; pode indicar varredura de URLs por scripts maliciosos.
- 405 Method Not Allowed – Uso de método HTTP não permitido (tentativa de explorar falhas de configuração).
- 500 Internal Server Error – Erro interno; pode indicar exploração de vulnerabilidades ou falhas críticas.

2.3.2 Tipos de Ataques em Servidores Web

Diversos tipos de ataques são frequentemente direcionados a servidores web (OWASP Foundation, 2025), sendo os mais comuns:

- Injeção de SQL (SQLi): o atacante explora vulnerabilidades em formulários ou parâmetros de URL para executar comandos SQL maliciosos, visando inserir ou manipular conteúdo do banco de dados.
- Varredura de diretórios (*Directory Scanning*): consiste no envio automatizado de múltiplas requisições para localizar arquivos e diretórios sensíveis ou scripts vulneráveis no servidor.
- Ataques de negação de serviço (DoS/DDoS): caracterizados pelo envio massivo de requisições ao servidor, com o objetivo de sobrecarregar os recursos do servidor e tornar o serviço indisponível.

- Exploração de vulnerabilidades em scripts: explora falhas em scripts web (como PHP e JavaScript que são muito visados) para executar ações não autorizadas ou comprometer o sistema.

Para a maioria desses ataques, uma etapa inicial comum é o envio de um grande volume de requisições ao servidor, seja tentando identificar recursos vulneráveis, seja explorando pontos fracos para, posteriormente, efetivar o ataque principal. Essa busca inicial por vulnerabilidades no servidor é uma das características que, em uma situação de ataque, torna possível a identificação de comportamentos anômalos nos logs de acesso, principalmente pela análise de padrões incomuns de requisições, URLs acessadas e códigos de resposta HTTP retornados (por exemplo, uma quantidade acima do normal de requisições retornando erro 404).

2.3.3 Redes Complexas e Grafos

As redes complexas são estruturas matemáticas utilizadas para modelar sistemas compostos por um grande número de elementos interconectados, cujas interações não seguem um padrão puramente aleatório nem totalmente regular. Essas redes são representadas por grafos, em que os vértices (nós) correspondem às entidades do sistema e as arestas (ligações) representam as conexões entre essas entidades (Newman, 2003).

Diferentemente de grafos simples com topologias regulares, as redes complexas apresentam propriedades emergentes como distribuição de grau heterogênea, presença de comunidades, caminhos curtos e alto coeficiente de agrupamento (Comin et al., 2020). Nem todo grafo, entretanto, pode ser classificado como uma rede complexa: apenas aqueles que exibem tais características estruturais e dinâmicas se enquadram nessa definição.

Os grafos construídos neste trabalho, derivados dos registros de log de acesso ao servidor Apache, apresentam essas propriedades, podendo, portanto, ser tratados como redes complexas. Contudo, para manter uma terminologia padrão e facilitar a descrição dos experimentos realizados para cada tipo de grafo gerado, adotou-se ao longo deste trabalho o termo "grafo" para se referir a essas estruturas.

Um grafo pode ser definido como uma estrutura matemática composta por um conjunto de vértices (ou nós) e um conjunto de arestas que conectam pares de vértices (Diestel, 2025). A notação formal de um grafo G é representada por um par:

$$G = (V, A)$$

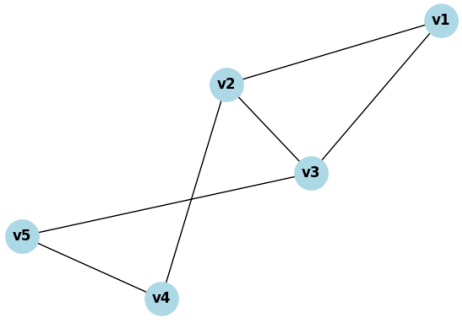
onde:

- $V = \{v_1, v_2, ..., v_n\}$ é o conjunto de vértices,

- $A = \{a_1, a_2, ..., a_m\}$ é o conjunto de arestas que conectam os vértices,
- $|V| = n$ indica o número de vértices (ou a ordem do grafo),
- $|A| = m$ indica o número de arestas.

A Figura 1 apresenta uma representação visual simplificada de um grafo e demonstra como os nós podem se conectar por diferentes relações, formando estruturas que podem ser analisadas em termos de grau dos vértices, caminhos mínimos, componentes conectados e outras métricas (Newman, 2018).

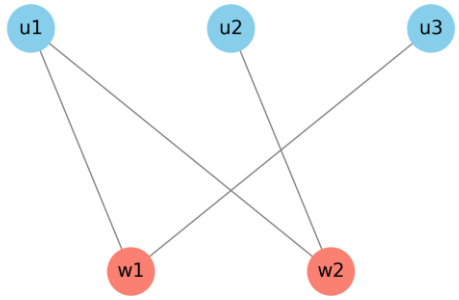
Figura 1 – Representação visual de um grafo com cinco vértices e seis arestas



2.3.3.1 Grafos Bipartidos e Tripartidos

Um grafo bipartido é um tipo especial de grafo em que o conjunto de vértices pode ser separado em dois grupos distintos, de modo que cada aresta sempre conecte um vértice do primeiro grupo com um vértice do segundo grupo. Formalmente, todo vértice em um conjunto só pode se relacionar com vértices do outro conjunto, assim um grafo $G = (V, A)$ é bipartido se seus vértices puderem ser particionados em dois conjuntos U e W, com $V=U \cup W$, tal que todas as arestas $a \in A$ conectem um vértice de U a um vértice de W (Diestel, 2025). A Figura 2 apresenta uma representação visual com um exemplo de grafo bipartido.

Figura 2 – Representação visual de grafo bipartido com três vértices no conjunto U={u1,u2,u3} e dois vértices no conjunto W={w1,w2}



Para além dos grafos bipartidos, há uma generalização conhecida por grafos k-partidos. Um grafo k-partido é aquele cujo conjunto de vértices pode ser particionado em k subconjuntos disjuntos — $V_1,V_2,...,V_k$ — de tal forma que nenhuma aresta conecta dois vértices pertencentes ao mesmo subconjunto. Ou seja, todas as arestas do grafo conectam vértices de subconjuntos diferentes, e nunca dentro do mesmo grupo. Usando essa generalização, um grafo tripartido pode ser um grafo particular dessa classe de grafos para k=3, em que o conjunto de vértices é dividido em três grupos e as conexões ocorrem sempre entre vértices de grupos diferentes. Grafos tripartidos são úteis para modelar sistemas em que existem três tipos distintos de entidades e os relacionamentos só são possíveis entre entidades de tipos diferentes, nunca entre entidades do mesmo tipo.

2.3.3.2 Grafos de Similaridades de Documentos

Um grafo de similaridades entre documentos é uma estrutura matemática utilizada para modelar e representar as relações de proximidade semântica ou léxica dentro de uma coleção de documentos (*corpus*). O objetivo fundamental é transformar dados textuais não estruturados em uma representação estruturada (o grafo). Nesta estrutura, os vértices do grafo representam cada documento do *corpus* e as arestas representam a existência de uma similaridade significativa entre o par de documentos que elas conectam com base em uma métrica definida (Aggarwal; Zhai, 2012). A construção de qualquer grafo de similaridade entre documentos envolve, conceitualmente, três fases distintas e sequenciais:

1) Vetorização dos Documentos: para que a similaridade seja calculada matematicamente, o texto não estruturado precisa ser convertido em uma representação numérica, geralmente um vetor em um espaço multidimensional (Modelo de Espaço Vetorial). Técnicas que podem ser aplicadas: BoW (Bag-of-Words), Word2Vec, TF-IDF, GloVe, BERT;
2) Quantificação da Similaridade (métricas): uma vez que os documentos estão representados numericamente (como vetores), aplica-se uma métrica de similaridade para quantificar a proximidade entre cada par de vetores. Técnicas de similaridade mais comuns: similaridade de cosseno (mede o ângulo entre vetores), distância euclidiana (mede a distância absoluta), índice de Jaccard (mede a sobreposição de conjuntos de termos);
3) Estruturação do Grafo (construção das conexões): nesta etapa final são estabelecidos os critérios para conexão entre os nós, quando se deve decidir se a similaridade

calculada é significativa o suficiente para justificar a criação de uma aresta entre dois documentos. Isso transforma uma matriz de similaridade (onde todos os documentos são comparados com todos os outros) em um grafo, geralmente mais esparso, que destaca apenas as relações mais fortes. Técnicas que podem ser utilizadas como critérios: adoção de um limite que conecta documentos apenas se a similaridade entre eles estiver acima de um valor pré-definido (por exemplo, similaridade > 0.6), aplicar a técnica k-NN conectando cada documento apenas aos seus k vizinhos mais próximos, ou ainda, criar um grafo totalmente conectado mantendo todas as conexões atribuindo pesos a todas as arestas em que baixos pesos indicam baixa similaridade.

2.3.3.3 Grafos Temporais

Os grafos temporais constituem uma extensão dos grafos convencionais ao incorporarem informações sobre a dimensão temporal das interações, o tempo em que as arestas e nós existem ou se alteram. Nesse tipo de estrutura, cada aresta ou vértice possui atributos que indicam o instante ou intervalo de tempo em que ocorrem. Em aplicações interativas, como exploração visual de logs, esses grafos permitem a navegação por diferentes janelas temporais e o acompanhamento da evolução dinâmica das relações ao longo do tempo, por exemplo, mudanças nos padrões de acesso ou surgimento de eventos anômalos (Holme; Saramäki, 2012; Jacomy et al., 2014).

2.3.3.4 Grafos de Detecção de Comunidades

A detecção de comunidades em grafos corresponde ao processo de identificar grupos de nós mais densamente conectados entre si do que com o restante do grafo. Dentre os métodos existentes, destaca-se o algoritmo de Louvain, que se baseia na maximização de uma métrica denominada modularidade e quantifica a qualidade de uma partição em comunidades. No contexto deste trabalho, a aplicação do método de Louvain pode contribuir para a identificação de agrupamentos de padrões de acesso que caracterizem comportamentos recorrentes ou potenciais incidentes de segurança. Ainda no contexto de logs, essa técnica ajuda a identificar grupos de IPs, URLs ou períodos temporais com comportamentos similares (Blondel et al., 2008; Fortunato, 2010).

3 COLETA E ANÁLISE EXPLORATÓRIA DOS DADOS

3.1 Coleta e Pré-Processamento dos Dados

Neste trabalho, os conjuntos de dados foram obtidos a partir dos arquivos de log de acesso de três servidores Web Apache em operação no IFSC, cada um responsável por funções institucionais distintas:

- Servidor A1: Servidor Apache destinado à hospedagem de sistemas institucionais da Intranet do IFSC, com aplicações desenvolvidas em PHP e banco de dados MySQL.
- Servidor A2: Servidor Apache que hospeda os principais portais institucionais do IFSC, incluindo o site oficial, portal da Biblioteca e os sites da Graduação e Pós-Graduação.
- Servidor A3: Servidor Apache utilizado para hospedagem de sites institucionais, além de páginas de grupos e laboratórios de pesquisa do IFSC.

O Apache registra automaticamente todas as requisições recebidas em arquivos denominados `access.log`, que armazenam informações detalhadas sobre cada acesso, como endereço IP de origem, data e hora da requisição, método HTTP, URL solicitada, código de resposta HTTP, entre outros parâmetros relevantes. Por padrão, em sistemas baseados em Linux, esses arquivos de log são mantidos no diretório `/var/log/apache2`.

Nos servidores analisados, pela política aplicada para armazenamento dos logs, os arquivos de acesso são criados diariamente, mantendo-se um histórico dos últimos 14 dias de registros. Assim, a cada novo dia, um novo arquivo de log é criado, e arquivos mais antigos são automaticamente removidos após duas semanas.

A coleta dos logs para este trabalho ocorreu durante um período total de nove meses (do mês de novembro de 2024 a julho de 2025). A Tabela 1 apresenta a média mensal de registros de logs coletados por servidor ao longo desse período. O volume total de linhas de logs, registrados nos três servidores analisados, ultrapassou 39 milhões de entradas, conforme sumarizado na tabela.

A partir do armazenamento desses arquivos de log diários, os dados foram utilizados nos experimentos realizados neste trabalho, tanto em análises de um único dia quanto em análises agregadas por mês, dependendo do objetivo do experimento. Os *datasets* correspondentes — compostos pelos arquivos de log processados — estão disponibilizados no repositório público do GitHub (https://github.com/flaviasalisboa/MBA-IA-2025).

Tabela 1 – Quantidade mensal de registros de logs coletados nos servidores Apache no IFSC

MÊS DA MÉDIA MENSAL	SERVIDOR A1	SERVIDOR A2	SERVIDOR A3
Novembro 2024	-	1.692.555	1.101.859
Dezembro 2024	-	1.845.009	757.714
Janeiro 2025	-	2.156.874	752.800
Fevereiro 2025	-	2.388.185	1.489.907
Março 2025	-	2.688.572	3.189.319
Abril 2025	-	2.824.186	1.704.352
Maio 2025	262.316	2.719.510	1.710.263
Junho 2025	329.717	3.001.002	2.575.821
Julho 2025	258.229	3.214.769	3.212.500

Diante do formato dos registros de log, é possível extrair oito atributos a partir de um registro que corresponde a uma linha de um arquivo de log. Veja na Tabela 2 a apresentação de todos atributos que podem ser extraídos a partir de um registro de log no formato CLF e os atributos selecionados para utilização nos experimentos desse trabalho.

Tabela 2 – Atributos extraídos de um registro de log e atributos selecionados para os experimentos

Atributos que podem ser extraídos dos logs	Nome do atributo	Tipo de atributo	Atributos selecionados para uso nos experimentos
Endereço IP de acesso	ip	categórico	✓
Data/Hora	data	timestamp	✓
Método HTTP	metodo	categórico	✓
URL acessada	url	categórico	✓
Código de Resposta HTTP	status	numérico inteiro	✓
Tamanho da resposta	tam	numérico inteiro	
Referência	ref	categórico	
Agente utilizado	agente	categórico	

Os dados de log de acesso do Apache estão armazenados em arquivos textos e apresentam uma estrutura semiestruturada: cada linha do arquivo segue um padrão fixo, mas os campos não estão organizados em colunas explícitas, como em um banco de dados relacional. Por isso, é necessário um pré-processamento para viabilizar a utilização desses atributos presentes nos registros. Para o pré-processamento dos logs foi utilizada a biblioteca Pandas, no Python. Veja na Figura 3 a saída de um *dataframe* de exemplo no Python gerado a partir de um arquivo de log.

Figura 3 – Estrutura dos atributos

ip	data	metodo	url	status	tam	ref	agente
143.107.228.199	2024-12-11 06:25:14-03:00	POST	/portal-ifsc/wp-cron.php?doing_wp_cron=1733909...	200	4531	-	WordPress/6.7.1; https://www2.ifsc.usp.br/port...
173.252.95.116	2024-12-11 06:25:12-03:00	GET	/portal-ifsc/a-importancia-da-vigilancia-epide...	200	192125	-	facebookexternalhit/1.1 (+http://www.facebook....
173.252.95.4	2024-12-11 06:25:17-03:00	GET	/portal-ifsc/wp-content/uploads/2019/11/pneumo...	200	35104	-	facebookexternalhit/1.1 (+http://www.facebook....
207.46.13.168	2024-12-11 06:25:21-03:00	GET	/portal-ifsc/category/noticias/page/213/	200	61903	-	Mozilla/5.0 AppleWebKit/537.36 (KHTML, like Ge...
47.128.116.16	2024-12-11 06:25:47-03:00	GET	/biblioteca	301	4975	-	Mozilla/5.0 (Linux; Android 5.0) AppleWebKit/5...

3.2 Análise Exploratória dos Dados

Nesta fase do trabalho, a análise exploratória dos dados foi realizada como uma etapa inicial para compreender o conteúdo dos logs de acesso do Apache e o comportamento dos acessos registrados. O principal objetivo dessa análise foi extrair informações relevantes, tais como os padrões de acesso, a frequência dos diferentes códigos de resposta HTTP, os IPs mais ativos, horários de maior atividade, e possíveis indícios de acessos anômalos ou maliciosos.

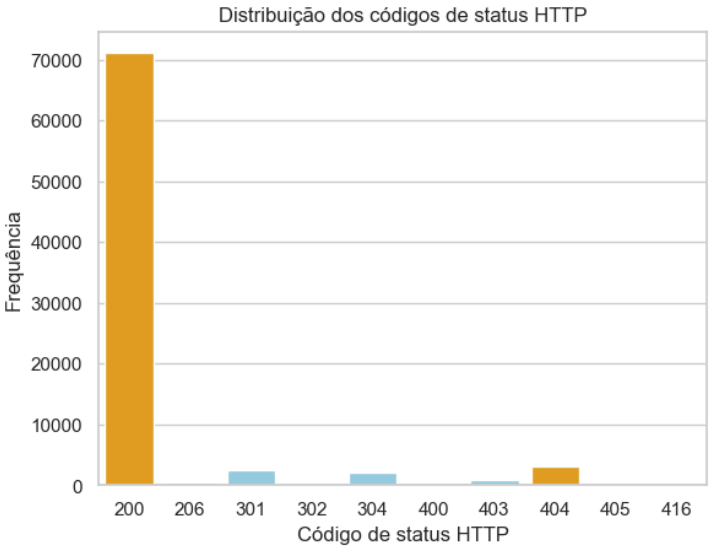
Por meio da análise exploratória, foi possível identificar características do tráfego, levantar hipóteses e direcionar a modelagem das redes (como a rede bipartida entre URLs e códigos de resposta HTTP) para investigação mais aprofundada de padrões, anomalias e potenciais ataques.

As figuras e gráficos apresentados nas seções seguintes foram gerados usando as bibliotecas Numpy, Matplotlib e Seaborn do Python utilizando um dia de log do servidor Apache do IFSC (arquivo de log do dia 11/12/2024 ao dia 12/12/2024 do Servidor A2 que abriga o site oficial do IFSC).

3.2.1 Histograma dos Códigos de Resposta HTTP

O histograma de códigos de resposta da Figura 4 apresenta a frequência de ocorrência de cada tipo de resposta HTTP do servidor (por exemplo, 200 para sucesso, 404 para não encontrado, 500 para erro interno). Isso permite identificar quais tipos de respostas são mais comuns e se há indícios de problemas (por exemplo, muitos códigos 404 podem sugerir tentativas de acesso a páginas inexistentes ou uma tentativa de varredura maliciosa).

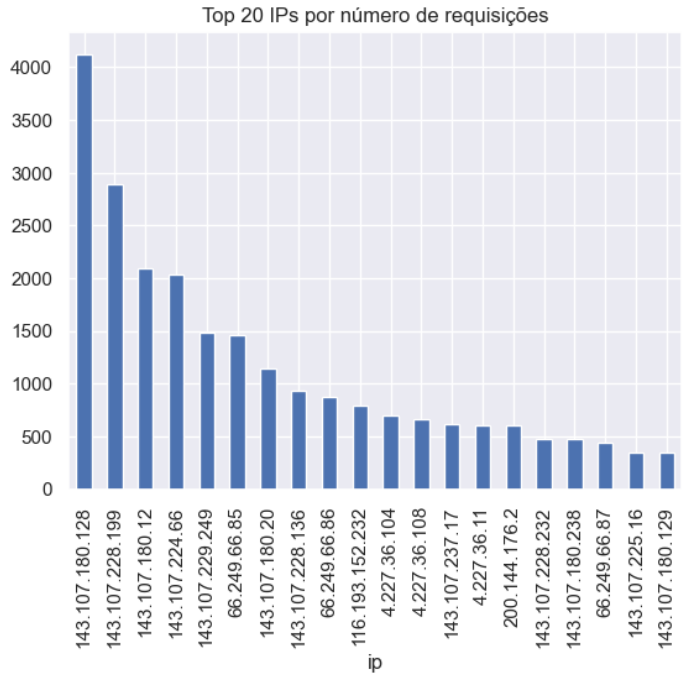
Figura 4 – Histograma de frequência das respostas HTTP de um dia de log



3.2.2 Gráfico de Barras dos IPs com Maior Número de Acessos

Neste gráfico de barras são exibidos os IPs que mais realizaram requisições ao servidor, o que pode evidenciar possíveis padrões de uso intenso, escaneamentos automatizados ou tentativas de ataque por força bruta. Nessa situação da Figura 5 as duas barras mais altas do histograma identificam dois IPs da própria rede do IFSC, caracterizando que a maioria dos acessos realizados nesse dia foram acessos internos normais.

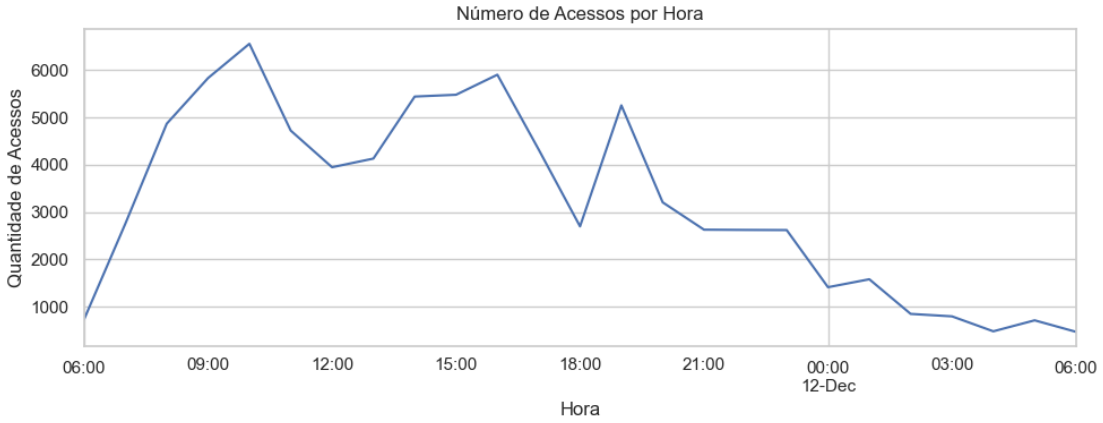
Figura 5 – Top 20 IPs com maior número de acessos em um dia de log



3.2.3 Gráfico de Linhas com Volume de Acessos por Hora

No gráfico de linhas do número de acessos por hora (Figura 6) é possível visualizar como o volume de acessos varia ao longo do tempo, o que pode auxiliar na identificação de horários de pico e comportamentos fora do padrão, como tentativas de ataque concentrados em determinados períodos.

Figura 6 – Gráfico de linhas representando o número de acessos por hora



3.2.4 Mapa de Calor (*Heatmap*) de Acessos por Hora/Dia

Os *heatmaps* filtrados pelos códigos de resposta HTTP, nas figuras 7 e 8, evidenciam quando e em quais dias ocorrem maiores concentrações de erros (404) e de acessos bem-sucedidos (200), podendo ser úteis para monitorar incidentes e tentativas de ataque. Na Figura 7 observa-se um pico expressivo no dia 11/12/2024, por volta das 19h, quando foram registrados 569 acessos resultando em erro 404 em apenas uma hora. O volume atípico de erros 404 em um curto intervalo de tempo pode ser um indício de atividade suspeita, e essa visualização pode auxiliar em apontar para uma investigação mais detalhada desses acessos, como a identificação dos IPs envolvidos e os padrões de URLs requisitadas nesse horário.

Figura 7 – Mapa de calor (*heatmap*) de acessos por dia e hora com código de resposta 404

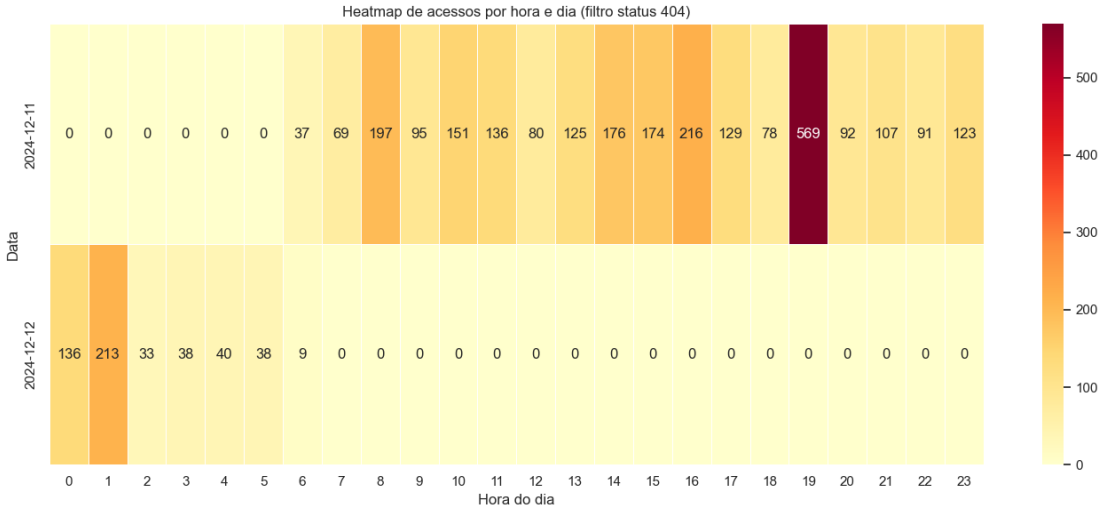
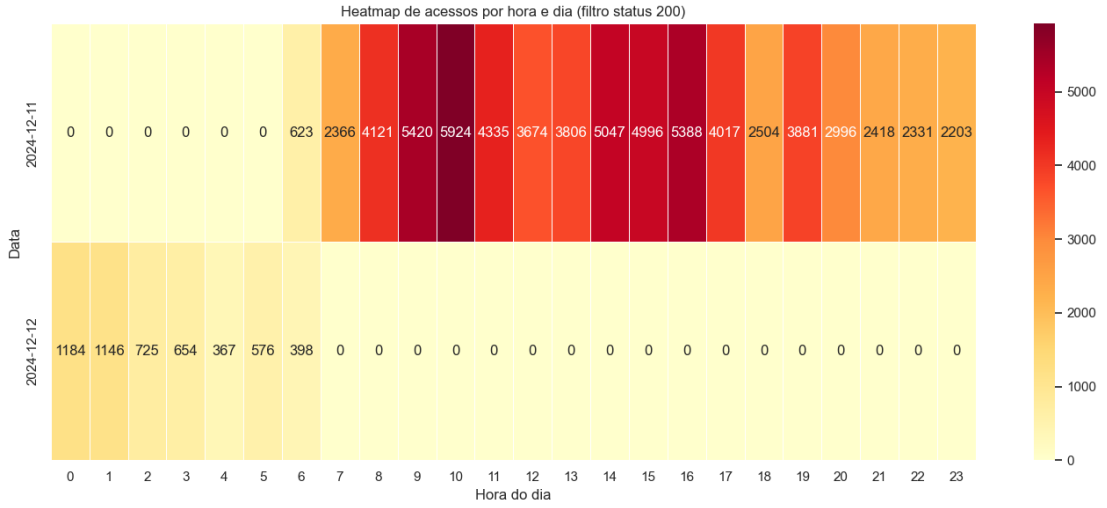


Figura 8 – Mapa de calor (*heatmap*) de acessos por dia e hora com código de resposta 200



4 METODOLOGIA

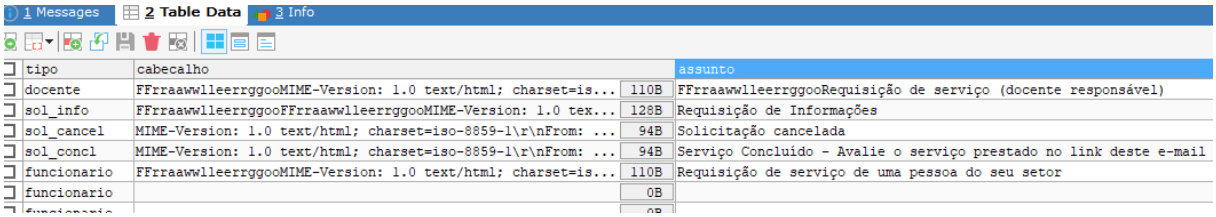
A metodologia adotada neste trabalho foi motivada por um caso concreto de invasão por injeção de SQL (SQLi) ocorrido em um dos servidores Apache do IFSC, no mês de maio do corrente ano. A partir desse incidente, surgiu a proposta de desenvolver uma estratégia experimental baseada na análise dos arquivos de log gerados durante a invasão que evidenciam uma ação real de ataques por SQLi no banco de dados do servidor. Com isso, explorou-se o uso de modelos de representação em grafos como abordagem principal para a análise desses dados.

4.1 Estudo de Caso: Ataque de Injeção de SQL no Servidor Apache

Em maio de 2025, houve um ataque de injeção de SQL no servidor Apache do IFSC que abriga sistemas institucionais em PHP e banco de dados MySQL (Servidor A1). Notou-se, pelos e-mails que são enviados pelos sistemas, que no “Assunto” de algumas mensagens haviam caracteres estranhos (Figura 9), possivelmente inseridos por algum tipo de invasão sofrida. Os dados de envio das mensagens feito pelo sistema estão configurados em uma tabela no banco de dados MySQL (ver Figura 10). Verificando os e-mails enviados pelo sistema, percebeu-se que a partir do dia onze de maio (Figura 9), os e-mails estavam com texto comprometido (caracteres estranhos inseridos no começo do “Assunto” das mensagens) e que antes dessa data estavam com o texto normal.

Figura 9 – Tela dos e-mails enviados pelos sistemas já comprometidos por injeção de SQL

Figura 10 – Inserção de caracteres estranhos (FFrraawwlleerrggoo) por injeção de SQL no banco de dados



1 Messages | 2 Table Data | 3 Info

tipo	cabecalho		assunto
docente	FFrraawwlleerrggooMIME-Version: 1.0 text/html; charset=is...	110B	FFrraawwlleerrggooRequisição de serviço (docente responsável)
sol_info	FFrraawwlleerrggooFFrraawwlleerrggooMIME-Version: 1.0 tex...	128B	Requisição de Informações
sol_cancel	MIME-Version: 1.0 text/html; charset=iso-8859-1\r\nFrom: ...	94B	Solicitação cancelada
sol_concl	MIME-Version: 1.0 text/html; charset=iso-8859-1\r\nFrom: ...	94B	Serviço Concluido - Avalie o serviço prestado no link deste e-mail
funcionario	FFrraawwlleerrggooMIME-Version: 1.0 text/html; charset=is...	110B	Requisição de serviço de uma pessoa do seu setor
funcionario		0B	

Com essa descoberta, passou-se a verificar os arquivos de log do Apache no servidor em busca de vestígios da invasão: por qual caminho foi possibilitada a entrada, de onde veio o ataque, qual tipo de ataque foi sofrido, se havia mais algum comprometimento além das inserções indevidas no banco de dados e outros indicativos do ataque sofrido. Três arquivos de log foram selecionados e investigados, no período abrangendo o dia anterior ao ocorrido, o dia em que a invasão efetivamente ocorreu (dia em que houve a inserção indevida no banco de dados) e o dia posterior ao da inserção indevida. A investigação nesses arquivos de log com registros da invasão buscou detectar os padrões utilizados pelo atacante para comprometer o sistema com esse tipo de ataque, identificado como um ataque de injeção de SQL.

No dia anterior ao ataque bem-sucedido (inserção efetiva de dados por injeção de SQL), verificou-se nos arquivos de log muitos registros de acesso a URLs variadas provenientes de vários endereços de IP externos, caracterizando tentativas de descoberta de scripts vulneráveis no servidor. E, assim, foi de fato descoberto um script vulnerável no servidor (o script `click.php` no diretório `/counter-downloads`) que foi o vetor utilizado para a injeção de SQL no banco de dados de um dos sistemas hospedado nesse servidor.

4.2 Estratégia Adotada nos Experimentos Usando Logs da Invasão ao Servidor

A partir do incidente e da investigação dos arquivos de log da invasão, surgiu a proposta de desenvolver uma estratégia experimental baseada na análise desses registros, comparando-os com os logs de dias de acessos considerados normais no servidor. O objetivo foi identificar padrões característicos de cada situação, utilizando modelos de grafos para permitir uma visualização das diferenças entre os comportamentos observados.

Para isso, foram utilizados dois conjuntos de dados: um arquivo contendo os logs de dias normais de acesso ao servidor e outro contendo os registros correspondentes aos dias em que o ataque por injeção de SQL ocorreu. A partir desses dados, foram exploradas diferentes

formas de representação em grafos com o intuito de evidenciar padrões distintos entre os acessos legítimos e os acessos maliciosos.

A parte experimental do trabalho seguiu essa linha de atuação, realizando análises comparativas entre as duas situações – dias normais e dias de ataque – por meio da construção e interpretação de modelos em grafos, buscando verificar visualmente as diferenças nos padrões de acesso ao servidor.

4.3 Estrutura da Parte Experimental

Os experimentos foram realizados utilizando-se de quatro modelos de grafos detalhados a seguir:

1) Grafos de similaridade entre arquivos de log.

- Utilizam-se as técnicas TF-IDF + similaridade cosseno e TF-IDF + similaridade k-NN.
- Comparar cada log diário com o log de invasão do caso concreto do ataque de injeção de SQL e detectar se algum dia apresenta padrão semelhante ao log do dia de ataque.

2) Grafos bipartidos e tripartidos.

- Construção de grafos bipartidos de arquivos de log suspeitos, em que os nós são dois atributos extraídos dos registros de log com o objetivo de comparação das relações entre esses atributos com os mesmos atributos do log de invasão.
  - URL x Códigos de Resposta HTTP: relaciona URLs acessadas com os códigos de resposta HTTP (por exemplo, 200, 404, 403, 500).
- Construção de grafos tripartidos de arquivos de log suspeitos em que os nós são os três atributos extraídos dos registros de log.
  - IP × URL × Códigos de Resposta HTTP: grafo com três tipos de nós, mostrando o caminho completo de acesso, de quem acessou (IP), o que acessou (URL) e com qual resultado (código de resposta HTTP – erro ou sucesso no acesso).

3) Grafos temporais por minuto e por hora com animação.

- Temporal x Código de Resposta de Erro 404: considera o tempo como dimensão (em minutos) e relaciona aos códigos de resposta de erro 404 registrados no período os IPs ou URLs.

- Temporal com Janelas por Hora (interativo e animado): variações com relações entre URL x Código de Resposta HTTP, IP x URL e bipartido URL x Código de Resposta HTTP (filtro por 404 e 200).

4) Grafos de detecção de comunidades usando o algoritmo Louvain.

- Detectar comunidades de endereços IP com comportamentos similares, a fim de identificar grupos de endereços IP possivelmente maliciosos, por exemplo, aqueles endereços que geram muitos registros de log com erro de acesso com os códigos de resposta de erro mais comuns como 404, 403 e 500.
  - Nós do grafo: endereços IP.
  - Arestas do grafo: conectam endereços IP que receberam o mesmo código de resposta HTTP.

4.4 Linguagem e Bibliotecas Utilizadas na Experimentação

Para a realização dos experimentos, foram desenvolvidos notebooks Jupyter em Python 3.12.6, disponibilizados no repositório público do GitHub (https://github.com/flaviasalisboa/MBA-IA-2025), e executados em uma estação com processador Intel Core i5, 16 GB de memória RAM e GPU NVIDIA GeForce RTX.

Para a construção e visualização dos grafos, foram utilizadas bibliotecas da linguagem Python:

- Pandas, re, Numpy: para extração dos dados dos arquivos de log utilizando expressões regulares com re e armazenando os atributos em variáveis Pandas.
- NetworkX, Matplotlib, Plotly: para a criação das estruturas dos grafos e suas visualizações interativas em HTML.
- Community: para usar algoritmos de detecção de comunidades como Louvain.

Em relação ao desempenho computacional, a geração dos grafos mostrou-se eficiente mesmo em contextos com grande volume de dados. Embora os modelos de similaridade e, principalmente, os grafos tripartidos apresentem maior tempo de execução em comparação às demais abordagens, seu processamento ainda é considerado ágil. Nos experimentos realizados, por exemplo, a construção de um grafo tripartido a partir de um arquivo contendo aproximadamente 260 mil linhas de log foi concluída em cerca de 30 minutos, sem a necessidade de infraestrutura computacional avançada.

5 EXPERIMENTAÇÃO

Durante o desenvolvimento da modelagem com grafos, foram conduzidas diversas experimentações utilizando diferentes abordagens para construção dos grafos e aplicando os modelos sobre os arquivos de logs coletados ao longo de um período de nove meses (ver Capítulo 3 para detalhes sobre o período de coleta).

No entanto, para fins de apresentação neste capítulo, foram selecionados apenas os experimentos considerados mais relevantes, seja pela representatividade dos padrões identificados e comportamentos observados ou pelo potencial de contribuição para os objetivos deste trabalho. Os experimentos aqui destacados concentram-se em alguns logs de dia/mês específico, nos quais foram identificadas ocorrências mais expressivas ou atípicas, que também se repetiram ou puderam ser notadas em outros dias/meses analisados.

5.1 EXPERIMENTAÇÃO 1: Grafos de Similaridade entre Logs

Para essa experimentação foram utilizados os dados do arquivo de log de invasão e dos arquivos de log coletados no mês de junho do corrente ano de dois servidores Apache em operação no IFSC: Servidor A1 e Servidor A2. O log de invasão utilizado nos experimentos foi criado com a união de três arquivos de log dos dias de ataque ao Servidor A1, compreendendo o log do dia em que a invasão efetivamente ocorreu (datada do dia 11/05/2025), o dia anterior ao da invasão (10/05/2025) e o dia posterior ao da invasão (12/05/2025). Na Tabela 3 são apresentadas as quantidades de registros de log do arquivo de log da invasão e do mês de junho para cada um dos dois servidores.

Cada arquivo de log de um dia foi tratado como um “documento textual” e aplicada a técnica de TF-IDF com n-gramas para o agrupamento por similaridade, usando cosseno e k-NN para compará-los ao log de invasão[1]. O objetivo é detectar se algum dos dias de log do mês armazenados no servidor Apache teve comportamento semelhante ao log da invasão, criando um mapa visual da similaridade comportamental entre os dias de log diários com o log de invasão integrados ao grafo.

[1] A ideia e a aplicação da técnica de similaridade entre documentos utilizada neste experimento foram inspiradas no conteúdo do Curso 11 – Inteligência Analítica com Mineração de Dados, ministrado para a Turma 4 do MBA em IA e Big Data. Foi adaptado, especificamente para este trabalho, um notebook Jupyter apresentado durante as aulas (Marcacini, 2025), mostrando a relevância prática dos conceitos abordados no curso e sua contribuição direta para o desenvolvimento dos trabalhos dos alunos no contexto do MBA.

Tabela 3 – Quantidade de registros de logs coletados dos servidores e do arquivo de log da invasão

Servidor	Mês	Quantidade de registros de log por dia							
A1	JUNHO	1	2	3	4	5	6	7	8
		808	11.871	12.089	11.871	98.342	11.175	722	2.365
		9	10	11	12	13	14	15	16
		11.755	11.736	11.717	11.496	12.654	2.203	11.218	12.228
		17	18	19	20	21	22	23	24
		11.852	10.293	522	630	985	641	11.870	12.672
		25	26	27	28	29	30		
		11.757	10.414	9.884	586	904	12.457		
A2	JUNHO	1	2	3	4	5	6	7	8
		53.989	115.112	105.023	109.969	227.253	95.204	166.805	65.364
		9	10	11	12	13	14	15	16
		122.624	105.463	100.918	121.568	128.770	56.189	60.238	129.786
		17	18	19	20	21	22	23	24
		113.763	98.479	64.933	59.712	70.817	66.727	140.600	112.896
		25	26	27	28	29	30		
		106.946	99.785	85.553	43.721	64.033	108.762		
LOGS DA INVASÃO	MAIO	9	10	11	12	13	14	15	16
		-	270	2.573	2.385	-	-	-	-

Para os experimentos com cada servidor foram gerados dois modelos de grafos nomeados por grafo G1 e grafo G2, modelados utilizando-se das técnicas descritas a seguir. As etapas para esse experimento foram delineadas por:

1) Pré-processamento dos logs por dia
   a. Coleta dos logs diários de todos os dias do mês utilizado no experimento, tratando-os como documentos distintos (log1, log2, log3... e assim por diante, até log30). O log de invasão também é tratado como um “documento textual”.
   b. Os documentos foram previamente pré-processados para conter tokens compostos pelos atributos selecionados por método HTTP + URL + código de resposta HTTP (por exemplo, "`GET /admin 403`").
2) Cálculo do TF-IDF (Frequência dos Termos e Inverso da Frequência nos Documentos)
   a. Utiliza a técnica de vetorização TF-IDF (usando `TfidfVectorizer`) aplicada sobre as sequências de tokens extraídos dos logs. O modelo considera n-gramas de 1 a 3 termos, o que permite capturar não só tokens isolados, mas também sequências de 2 e 3 tokens (por exemplo, "`GET /admin`", "`GET /admin 403`" além de apenas o “`GET`”). O TF-IDF transforma o texto de cada dia de log em um vetor esparso de pesos. Cada peso representa a importância relativa de um token (ou sequência de tokens) naquele documento em relação aos demais.
   b. Armazena todos os vetores TF-IDF gerados de cada dia de log.

3) Cálculo da similaridade
   a. Para o cálculo da similaridade no grafo G1: compara entre si todos os vetores TF-IDF gerados usando a métrica de similaridade de cosseno e obtém a matriz de similaridade entre os dias de log. A similaridade de cosseno mede o ângulo entre dois vetores e resulta em um valor de 0 a 1, sendo que 1 indica vetores idênticos (comportamentos quase idênticos nos logs) e 0 indica vetores ortogonais (comportamentos totalmente diferentes). Gera a matriz de similaridade entre os dias de log.
   b. Para o cálculo da similaridade no grafo G2: aplica a técnica k-NN na construção da matriz de adjacência para preparar para a construção do grafo, criando as arestas entre os k mais próximos dos vetores TF-IDF gerados para os arquivos de log.
4) Construção do grafo
   a. Cada nó do grafo é um dia de log, representando um documento de texto.
   b. Na construção do grafo G1: usada a biblioteca NetworkX para construir o grafo. Após a obtenção da matriz de similaridade entre os dias de log, foi construído um grafo não direcionado, em que cada aresta indica uma relação de similaridade relevante (acima de um limite pré-definido). Neste grafo as arestas são criadas entre pares de dias cuja similaridade, previamente calculada via TF-IDF + cosseno, seja maior ou igual ao limite definido no experimento (por exemplo, similaridade cosseno > 0.6).
   c. Na construção do grafo G2: usada a biblioteca Plotly para construir o grafo. Neste grafo as arestas são criadas aplicando-se o método k-NN que conectam os k mais próximos dias de log (por exemplo, com k=3 cria arestas dos 3 dias mais próximos).
   d. Na construção dos grafos o log de invasão foi destacado visualmente, em vermelho, para facilitar a visualização dos dias de log que estão conectados a ele, o que indica que esses logs têm alta similaridade com o log de invasão e foram destacados em laranja; os demais dias destacados em azul-claro.

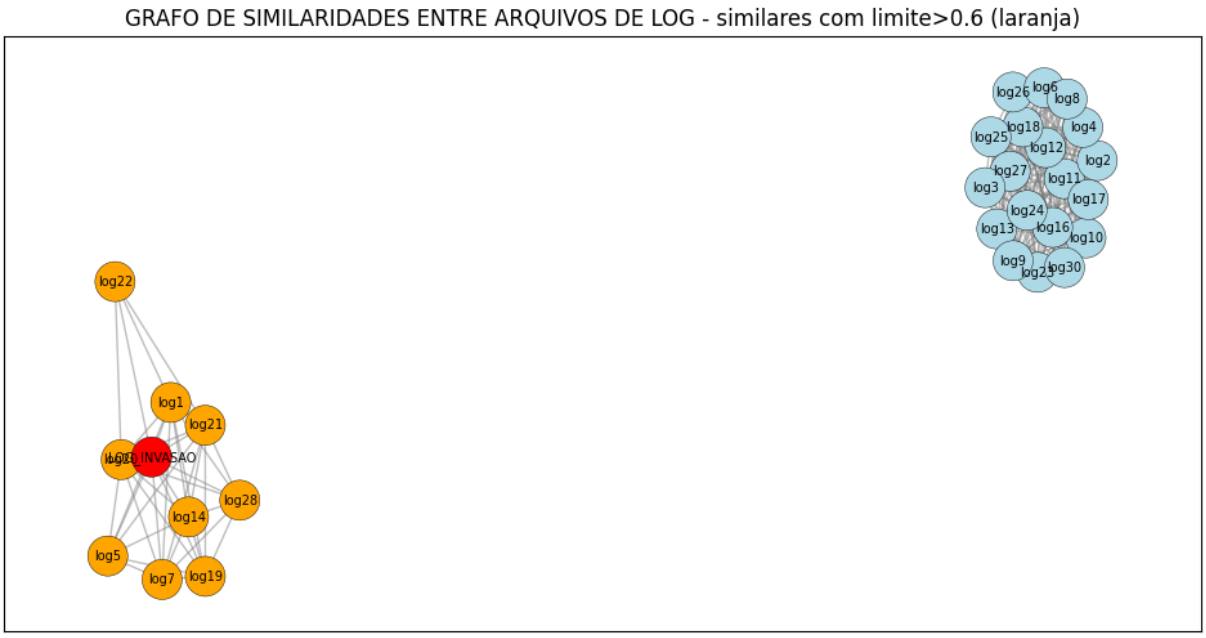
A seguir são apresentadas as imagens dos grafos gerados nos experimentos e uma sistemática de identificação aplicada para facilitar a referência dos diferentes testes realizados. A Tabela 4 apresenta a nomenclatura adotada para cada experimento, juntamente com suas principais características: o servidor utilizado, o modelo de grafo aplicado (G1 ou G2), o mês correspondente aos arquivos de log analisados, e os parâmetros específicos empregados em cada abordagem — como o limite de similaridade pelo cosseno (no caso do modelo G1) ou o valor de k no algoritmo k-NN (no caso do modelo G2).

Tabela 4 – Características dos experimentos realizados na EXPERIMENTAÇÃO 1

EXPERIMENTO	SERVIDOR	MODELO GRAFO	MÊS DOS LOGS	LIMITE COSSENO	VALOR k (k-NN)
A1-G1-JUNHO	A1	G1	JUNHO	0.6	-
A1-G2-JUNHO	A1	G2	JUNHO	-	3
A2-G1-JUNHO	A2	G1	JUNHO	0.6	-
A2-G2-JUNHO	A2	G2	JUNHO	-	3

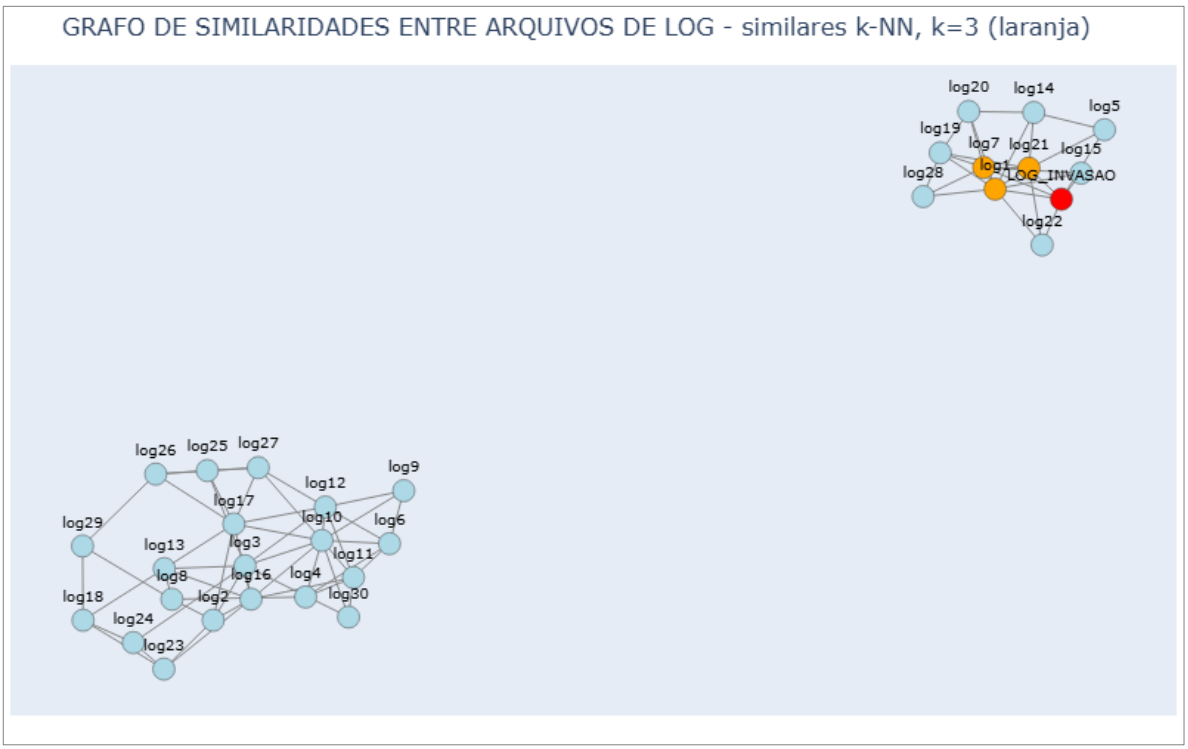
As figuras 11 e 12 representam os grafos de similaridade gerados para os experimentos com o Servidor A1 nomeados A1-G1-JUNHO e A1-G2-JUNHO. Em todos os grafos, cada nó representa um dia de log e as arestas indicam relações de similaridade entre eles, calculadas a partir dos vetores TF-IDF.

Figura 11 – Grafo de similaridades gerado pelo experimento A1-G1-JUNHO



O grafo G1 foi construído aplicando a similaridade de cosseno com um limite definido (similaridade>0,6) para o grafo da Figura 11. Nessa abordagem, apenas os dias de log com comportamento suficientemente próximo são conectados. Visualmente, isso gera clusters mais bem definidos, destacando grupos de dias com forte semelhança entre si. O log de invasão (em vermelho) conecta-se apenas a um subconjunto reduzido de dias (em laranja), evidenciando os períodos com maior semelhança de padrão em relação ao ataque, enquanto outros dias permanecem isolados ou pouco conectados. No experimento A1-G1-JUNHO da Figura 11 os nós mais similares (em laranja) foram log1, log5, log7, log14, log19, log20, log21, log22, log28.

Figura 12 – Grafo de similaridades gerado pelo experimento A1-G2-JUNHO



O grafo G2, apresentado na Figura 12, foi construído aplicando o método k-NN (k=3). Nesse caso, cada dia de log é conectado aos três mais próximos, independentemente do valor absoluto da similaridade. Essa abordagem resulta em uma rede mais conectada, com menos isolamento de nós, mas pode gerar conexões adicionais com similaridade menor, o que dilui levemente a separação entre grupos distintos. Ainda assim, observa-se no grafo do experimento A1-G2-JUNHO (Figura 12) um subgrafo fortemente associado ao log de invasão, formado pelos dias com maior proximidade comportamental.

Tabela 5 – Métricas calculadas na EXPERIMENTAÇÃO 1 para o Servidor A1 (grafos G1 e G2)

Servidor A1			
Mês	**Métrica**	**G1**	**G2**
JUNHO	Nro nós	29	31
	Nro arestas	210	78
	Nro componentes conectados	2	2
	Grau médio	14,48	5,0323
	Densidade do grafo	0,5172	0,1677
	Diâmetro do grafo	não conexo	não conexo
	Nó mais central (grau)	0,6429 (log10)	0,3333 (log10)
	Nó mais central (proximidade)	0,6429 (log10)	0,4011 (log10)

A análise das métricas da Tabela 5 reforça as observações descritas, anteriormente, das diferenças relativas aos dois modelos de grafos analisados, G1 e G2.

O grafo G1, por ser mais restritivo, apresenta:

- Maior grau médio (14,48), pois os dias altamente similares (em laranja) estão densamente conectados, dentro dos grupos de que fazem parte.
- Densidade mais elevada (0,5172), refletindo a formação de clusters bem definidos.
- Presença de menos componentes desconexos (2).

Já o grafo G2 apresenta:

- Grau médio menor (5,0323), o que decorre das conexões limitadas ao valor de k definido (k=3).
- Densidade menor (0,1677), o que indica uma rede mais distribuída.
- Dois componentes principais, separando os dias similares ao log de invasão dos demais.

Além disso, ao observar as métricas de centralidade, nota-se que, em ambos os modelos, os nós com maior centralidade correspondem a dias próximos ao log de invasão, o que corrobora a capacidade dos dois métodos de destacar os períodos de maior relevância para análise de comportamento anômalo.

Essas análises e observações podem ser visualizadas, também, nos experimentos conduzidos com os logs do Servidor A2, que são apresentadas a seguir.

Na Figura 13 (grafo A2-G1-JUNHO) não foi identificado nenhum dia de log com similaridade acima do valor de similaridade maior que 0,6 em relação ao log de invasão. Esse resultado indica que os padrões de acesso registrados neste servidor durante o período analisado apresentam diferenças significativas em relação ao comportamento observado no log do ataque, não atendendo ao limite de similaridade definido. Essa ausência de conexões é um comportamento esperado quando o limite de similaridade é mais restritivo, evidenciando que não houve dias com indícios fortes de atividades semelhantes à invasão.

Por outro lado, na Figura 14 (grafo A2-G2-JUNHO), o modelo k-NN conecta cada dia de log aos três mais similares. Dessa forma, mesmo que não haja dias com similaridade elevada, ainda são criadas conexões com os três dias mais próximos. Nesse caso, o log de invasão aparece conectado aos dias log5, log14 e log18, representando os dias relativamente mais próximos, ainda que com menor grau de similaridade.

Figura 13 – Grafo de similaridades gerado pelo experimento A2-G1-JUNHO

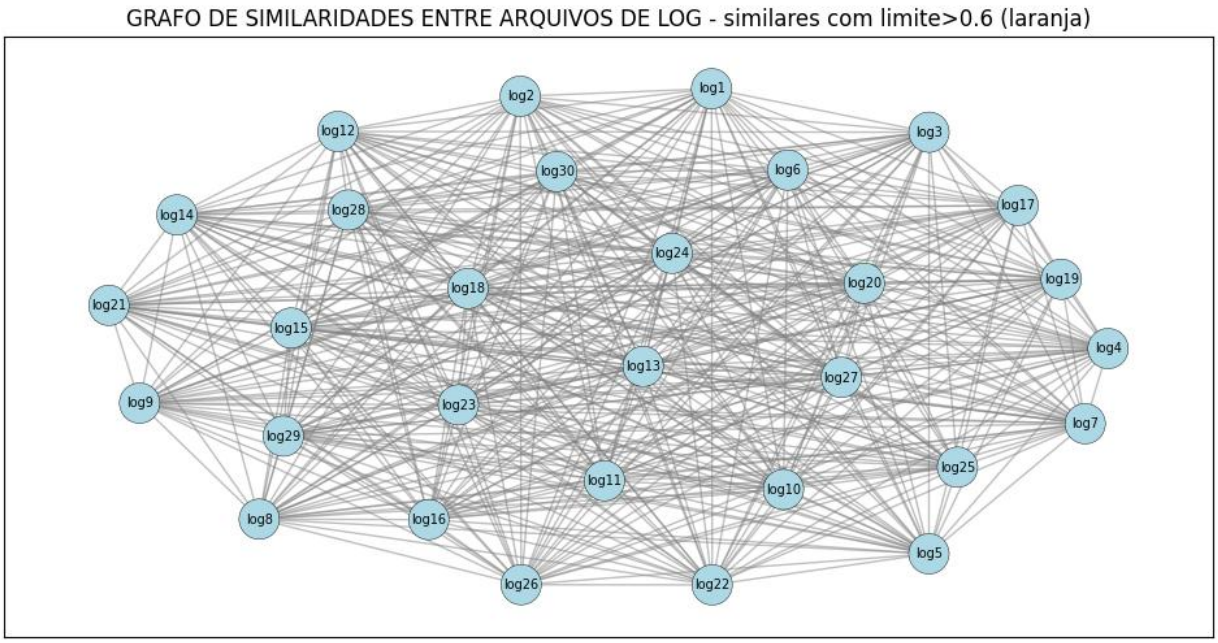


Figura 14 – Grafo de similaridades gerado pelo experimento A2-G2-JUNHO

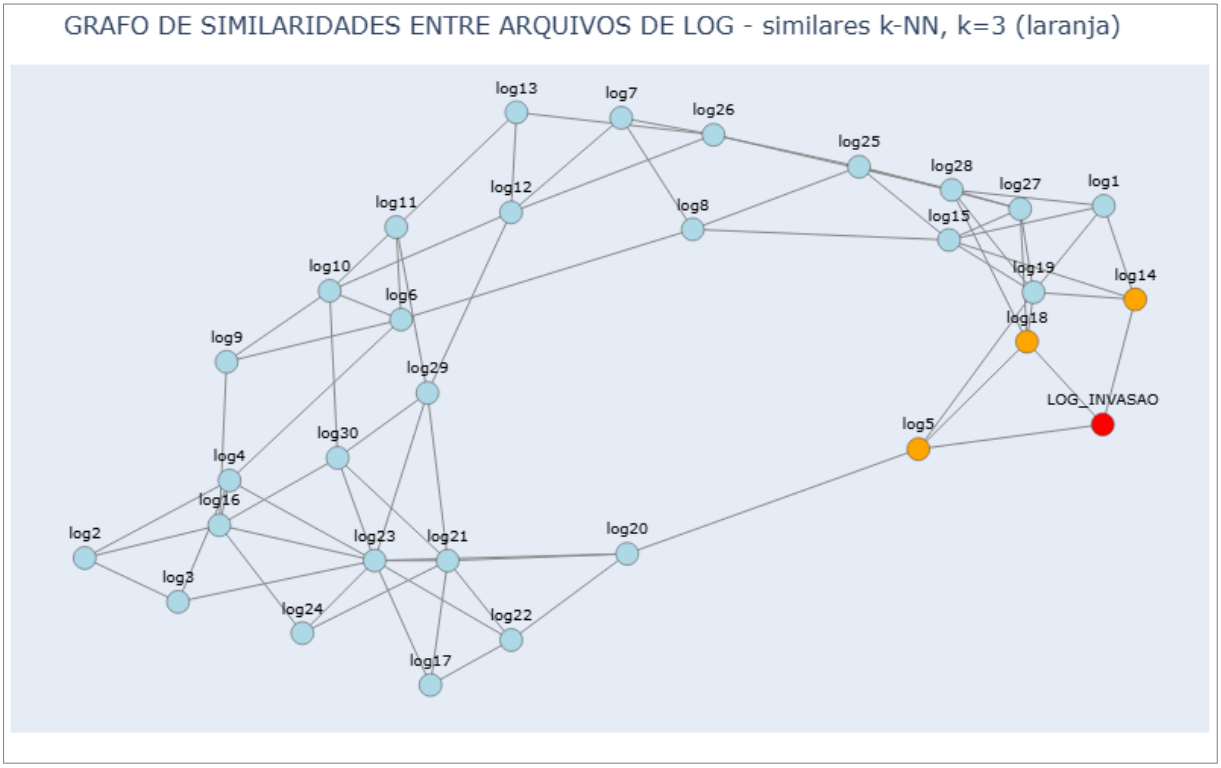


Tabela 6 – Métricas calculadas na EXPERIMENTAÇÃO 1 para o Servidor A2 (grafos G1 e G2)

Servidor A2			
Mês	**Métrica**	G1	G2
JUNHO	Nro nós	30	31
	Nro arestas	435	72
	Nro componentes conectados	1	1
	Grau médio	29	4,6452
	Densidade do grafo	1	0,1548
	Diâmetro do grafo	1	5
	Nó mais central (grau)	1 (log1)	0,3333 (log23)
	Nó mais central (proximidade)	1 (log1)	0,4348 (log23)

Pelas métricas apresentadas na Tabela 6, observa-se que o grafo G1 possui densidade igual a 1 e diâmetro igual a 1, o que significa que ele é um grafo totalmente conectado. Nesse caso, cada dia de log está diretamente ligado a todos os outros dias, formando uma rede completa sem a presença de subgrupos isolados. Esse comportamento reforça a ausência de padrões específicos de similaridade com o log de invasão, já que a aplicação do limite de 0,6 resultou em um grafo em que todos os nós são igualmente conectados entre si, sem destacar conexões relevantes com o dia do ataque.

Já no grafo G2, o método k-NN (k=3) gera uma estrutura mais seletiva, com densidade de 0,1548 e diâmetro igual a 5, evidenciando uma rede menos densa e mais adequada para observar as relações relativas entre os dias de log. Nesse modelo, é possível identificar um subconjunto reduzido de dias mais próximos ao log de invasão (log5, log14 e log18), mesmo que com baixa similaridade absoluta, destacando as diferenças entre os dois métodos: G1, nessa situação, evidencia homogeneidade, enquanto G2 indica que pode haver uma análise mais exploratória (por exemplo, investigando esses k mais próximos).

Essas diferenças analisadas entre os modelos de grafos G1 e G2, para os experimentos com os logs dos dois servidores, evidenciam como o método G1 (limite por similaridade) é mais rigoroso e adequado para destacar apenas casos com forte correlação, enquanto o modelo G2 (k-NN) é mais útil para análises exploratórias, permitindo observar quais seriam os dias mais próximos do log de invasão mesmo em cenários com baixa similaridade geral. Esses resultados consolidam a efetividade do G1 para identificar padrões fortemente correlacionados ao log de invasão, enquanto o G2 se mostra útil como suporte para análise exploratória.

5.2 EXPERIMENTAÇÃO 2: Grafos Bipartidos e Tripartidos

Nesta experimentação, foram explorados grafos bipartidos e tripartidos a partir dos dados extraídos do arquivo de log da invasão, bem como dos logs do mês de junho do corrente

ano, coletados do Servidor A1. O principal objetivo desta experimentação foi realizar uma análise comparativa entre os padrões visuais observados nas diferentes situações, buscando identificar, por meio das representações gráficas, possíveis comportamentos anômalos semelhantes ao ataque registrado.

A seguir, são apresentados alguns dos grafos gerados ao longo dos experimentos, especificamente aqueles que evidenciaram padrões mais distintos ou representativos. As figuras apresentadas ilustram os grafos organizados em pares, sendo cada par composto pelo grafo gerado a partir de um dia de log considerado normal para o servidor analisado e outro correspondente ao dia de ataque. Esse método comparativo adotado busca evidenciar visualmente as possíveis anomalias de acesso presentes em um dia de ataque, contrastando com o comportamento típico observado em dias normais de acessos ao servidor analisado.

Nesses experimentos foram utilizados o log do dia da invasão e o log de um dia de um dos servidores para a análise comparativa visual entre eles. Nas figuras apresentadas a seguir está contemplado o arquivo de log do dia 03/06/2025 do Servidor A1.

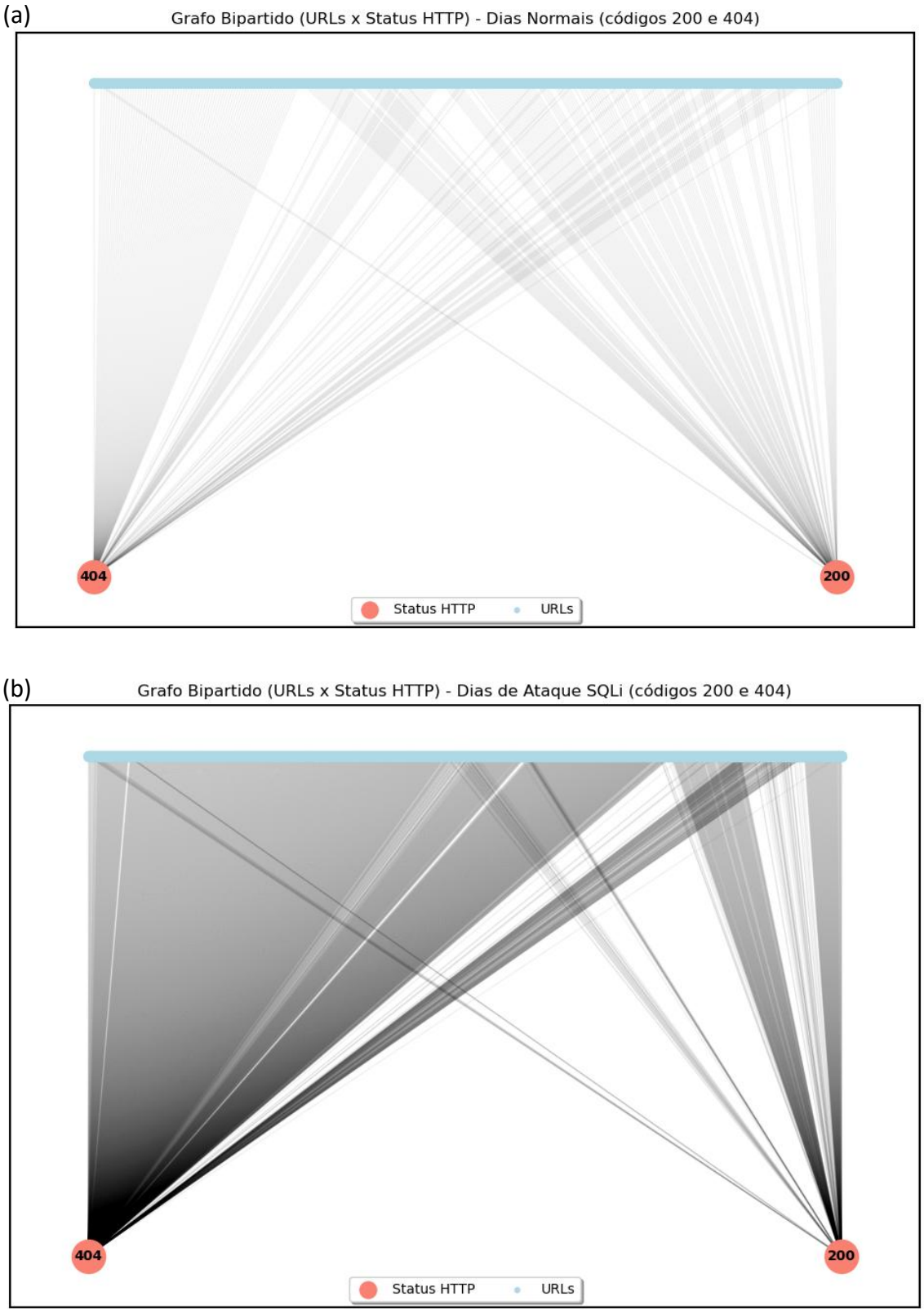
5.2.1 Experimentos Comparativos com Grafos Bipartidos

Este experimento apresenta um comparativo entre dias normais de operação do servidor e o dia do ataque SQLi, utilizando grafos bipartidos. O objetivo é analisar a relação entre os códigos de resposta HTTP (200 e 404) e os nós que representam URLs acessadas.

Na Figura 15, os grafos bipartidos foram construídos considerando URLs específicas. No grafo correspondente aos dias normais (Figura 15a), observa-se uma distribuição mais equilibrada de conexões entre URLs e os códigos de resposta, com menor densidade de arestas associadas ao status 404, refletindo o comportamento esperado de acesso a recursos válidos e normais de uso do servidor. Já no grafo do dia de ataque (Figura 15b), nota-se um aumento significativo na concentração de arestas ligadas ao status 404, indicando grande volume de tentativas de acesso a URLs inexistentes, característica comum de varreduras automatizadas realizadas durante ataques.

Essa comparação entre dias normais e de ataque permite identificar visualmente o impacto de eventos maliciosos no padrão de acessos, indicando a utilidade da modelagem bipartida como ferramenta visual para detecção de comportamentos anômalos.

Figura 15 – Grafos bipartidos (URL x Código de Resposta HTTP – 200 e 404). (a) Dias normais: distribuição equilibrada com baixa densidade de erros 404. (b) Ataque SQLi: alta concentração de conexões com código 404, indicando varreduras

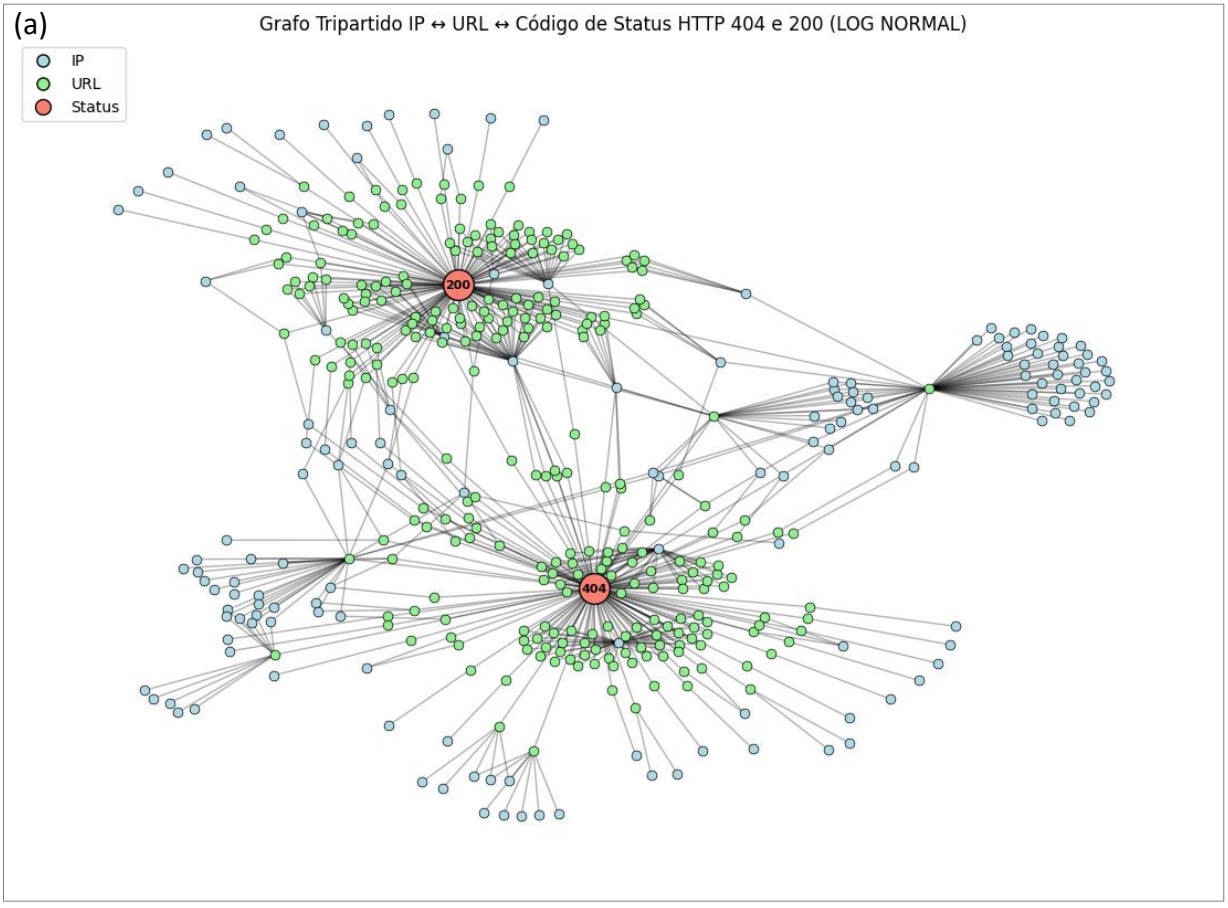


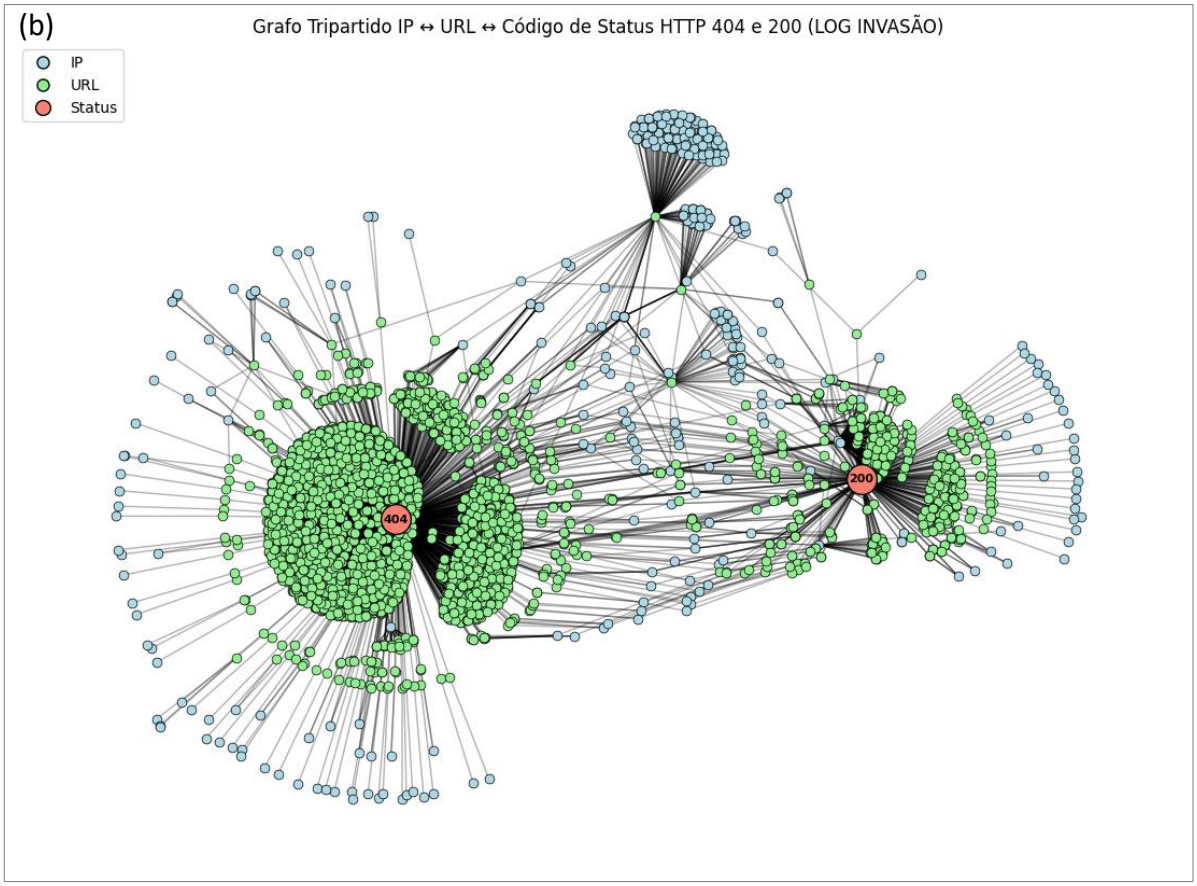
5.2.2 Experimentos Comparativos com Grafos Tripartidos

Nesses experimentos são apresentados os resultados realizados com grafos tripartidos, nos quais foram modeladas as relações entre IPs, URLs e códigos de resposta HTTP. Essa abordagem permite uma visão detalhada das interações, destacando como cada IP está conectado às URLs acessadas e, por sua vez, aos diferentes códigos de resposta retornados pelo servidor.

Na Figura 16, observa-se o grafo ao aplicar um filtro apenas para os códigos 200 e 404. No grafo de dias normais (Figura 16a), os IPs estão na sua maioria vinculados a URLs com respostas 200, enquanto no grafo do dia de ataque (Figura 16b), observa-se um aumento significativo de conexões associadas ao status 404, concentradas em grupos de IPs possivelmente maliciosos.

Figura 16 – Grafos tripartidos (IP x URL x Código de Resposta HTTP – filtro 200 e 404). (a) Dias normais: IPs em sua maioria vinculados a URLs com respostas 200. (b) Ataque SQLi: concentração de conexões com status 404 em IPs suspeitos





Essa comparação mostrou que essa modelagem tripartida permite agregar mais informação e ampliar a capacidade de análise em relação aos grafos bipartidos, fornecendo uma representação das interações entre IPs e recursos acessados, além daquela apenas com URLs e códigos HTTP da modelagem bipartida apresentada na seção anterior.

5.3 EXPERIMENTAÇÃO 3: Grafos Temporais

Nesta etapa, foram construídos grafos temporais com base nos dados extraídos do log correspondente a uma invasão e dos logs do mês de junho do corrente ano, referentes ao Servidor A1. O principal objetivo desta experimentação foi analisar visualmente a evolução temporal das conexões registradas, utilizando agregações por minuto e por hora. Para isso, foram geradas visualizações interativas com animações que possibilitam observar, de forma dinâmica, a variação dos acessos ao longo do tempo.

As observações visuais dos grafos ao longo da linha do tempo permitem detectar indícios de comportamentos semelhantes ao registrado durante o ataque. Esse tipo de visualização permite identificar padrões recorrentes ou anômalos quadro a quadro (hora por

hora), podendo ser utilizada como uma ferramenta tanto para análise exploratória dos logs quanto para aplicações práticas em sistemas de monitoramento em tempo real.

Para os experimentos a seguir, foram utilizados os dados do log do dia 03/06/2025, referente a um dia comum do Servidor A1, e o log do dia do ataque, possibilitando uma análise visual comparativa entre os dois cenários.

5.3.1 Experimentos Comparativos com Grafos Temporais Por Minuto

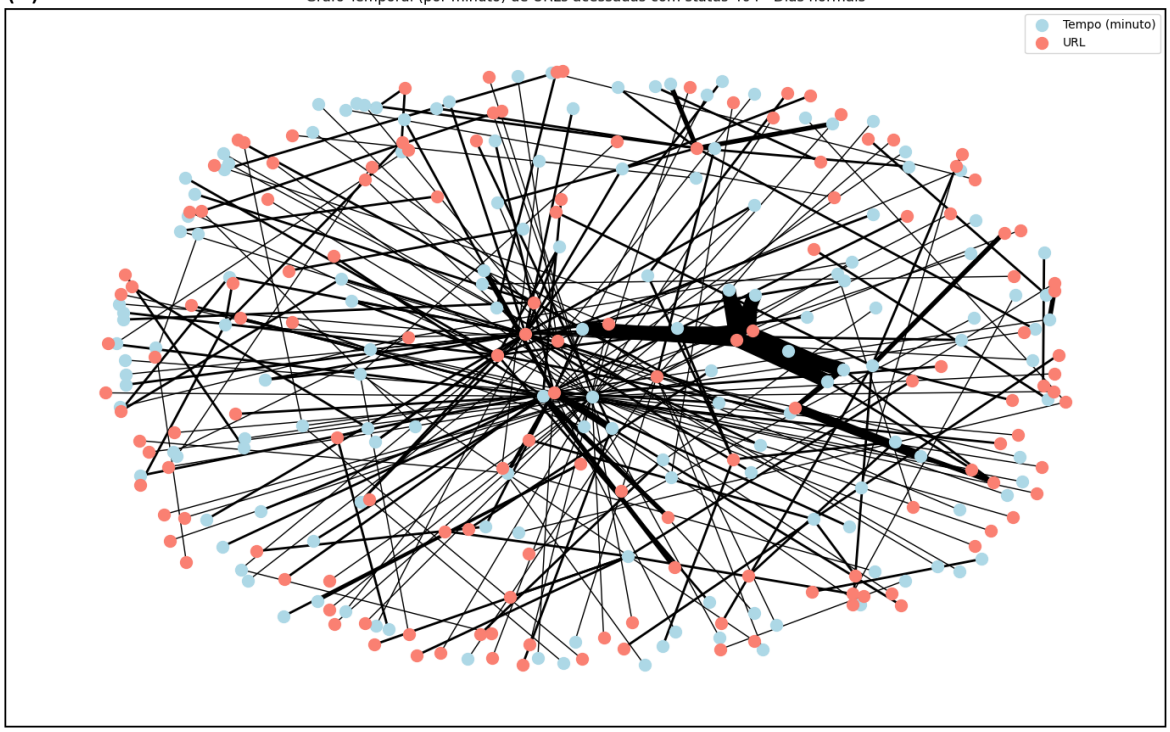
As Figuras 17 e 18 ilustram grafos temporais construídos a partir dos acessos registrados por minuto, considerando exclusivamente as requisições que resultaram em erro HTTP 404 (recurso não encontrado). Na Figura 17, os nós representam as URLs acessadas, enquanto na Figura 18 os nós correspondem aos IPs que realizaram as requisições. Em ambos os casos, os nós de tempo representam os minutos em que essas requisições ocorreram, formando grafos bipartidos entre o tempo e o outro elemento (URL ou IP).

As subfiguras (a) apresentam os padrões registrados em um dia considerado normal, e as subfiguras (b) retratam os padrões observados no log do dia da invasão.

Essa forma de representação gráfica pode auxiliar a identificar, de maneira visual, um aumento repentino e/ou um espalhamento das requisições 404 ao longo do tempo em casos de ataque — o que pode ser verificado pelo crescimento do número de conexões e pela densidade das arestas. Desta forma, os grafos podem revelar o contraste entre o comportamento típico e o comportamento anômalo, contribuindo para a detecção e compreensão de possíveis estratégias de ataque, como varreduras automatizadas ou tentativas de exploração em massa.

Figura 17 – Grafos temporais por minuto de acessos às URLs registrados nos logs que geraram código de resposta 404. (a) Grafo representando logs de dias normais. (b) Grafo representando logs do dia de ataque SQLi

(a)



(b)

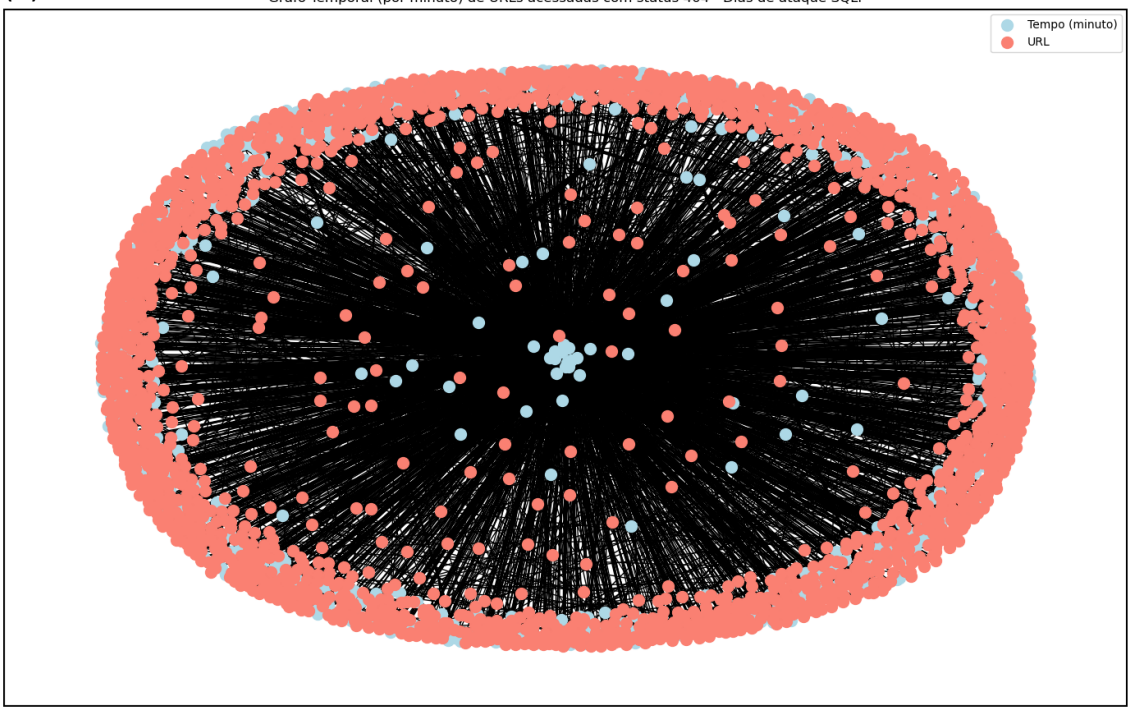
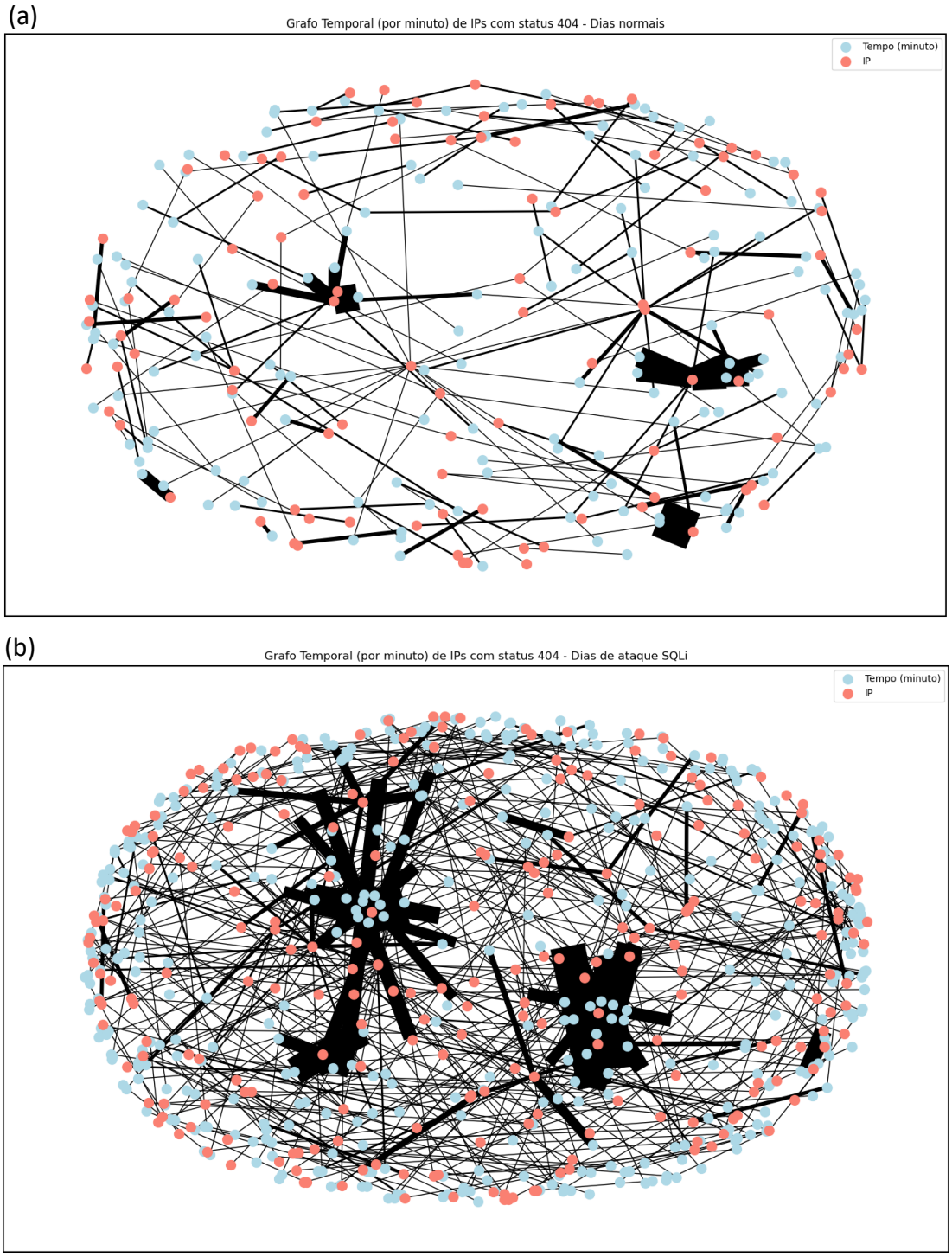


Figura 18 – Grafos temporais por minuto de acessos de IPs registrados nos logs que geraram código de resposta 404. (a) Grafo representando logs de dias normais. (b) Grafo representando logs do dia de ataque SQLi



5.3.2 Experimentos Comparativos com Grafos Temporais com Animação e Interatividade

Na sequência de imagens quadro a quadro apresentada na Figura 19, cada grafo representa os acessos às URLs registrados em um intervalo de uma hora, filtrados

especificamente pelas requisições que resultaram no código de resposta HTTP 404 (recurso não encontrado) nos logs do dia de invasão. Por exemplo, o primeiro quadro refere-se ao dia 10/05/2025, no intervalo entre 07:00 e 07:59, enquanto o segundo quadro representa o período entre 08:00 e 08:59 do mesmo dia, e assim sucessivamente até o quadro 49, que cobre o horário das 07:00 às 07:59 do dia 12/05/2025. A sequência de quadros da Figura 19 deve ser comparada com a sequência de imagens quadro a quadro da Figura 20 que apresenta os grafos formados naquele intervalo de hora dos logs do dia de acesso normal, também com filtro das requisições que retornaram código de erro 404.

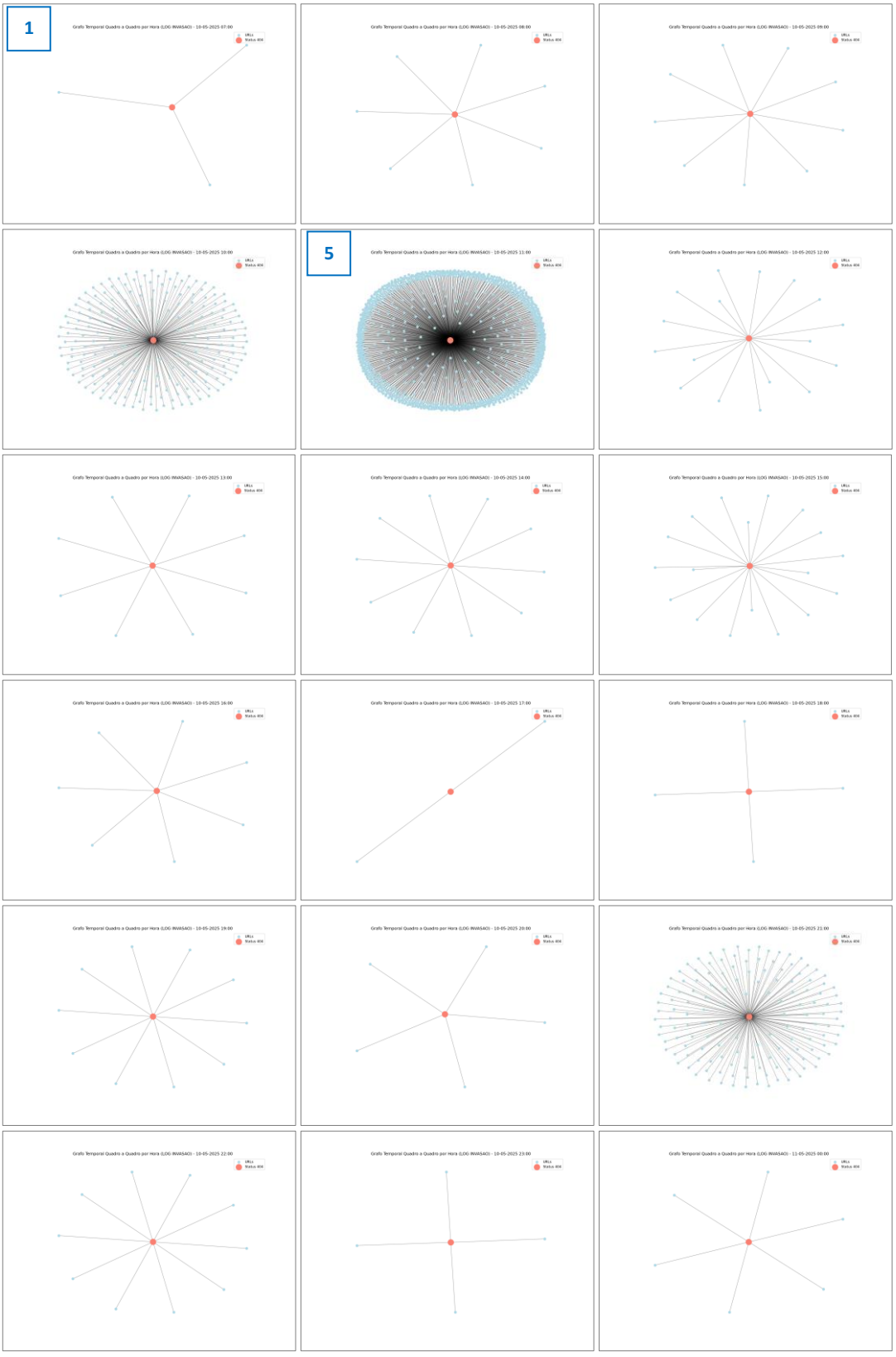
Para o arquivo de logs do dia da invasão, nos experimentos, foram registradas 4.851 linhas de log resultando em 49 intervalos de hora encontrados e, portanto, 49 quadros. Para os arquivos de logs do dia normal de acessos, foram registradas 21.894 linhas de log resultando em 25 intervalos, sendo 25 quadros (cada um representando um intervalo de hora).

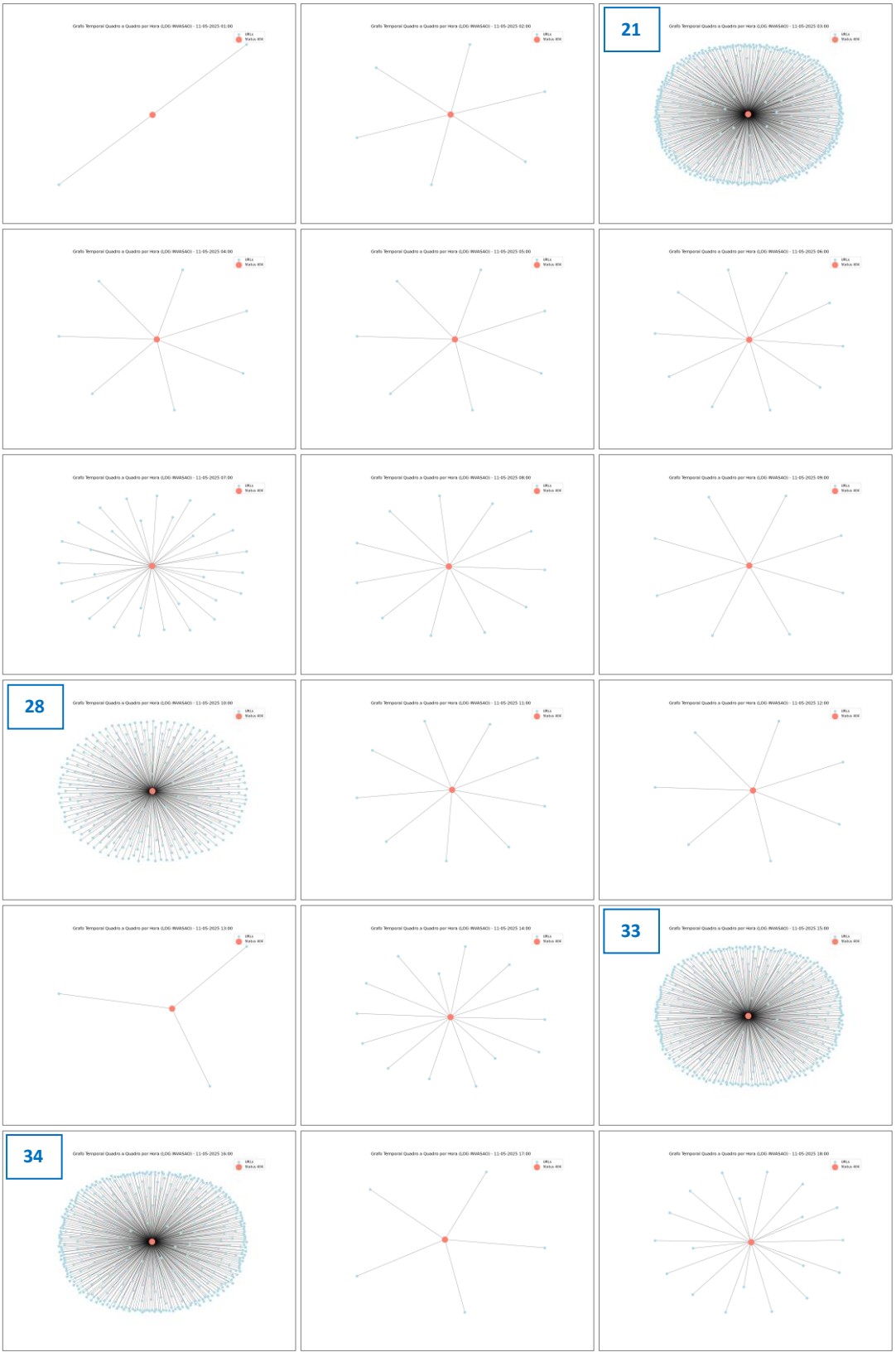
Nota-se, na Figura 19, um padrão visual claramente distinto nos quadros 5 (11/05/2025 11:00), 21 (11/05/2025 03:00), 28 (11/05/2025 10:00), 33 (11/05/2025 15:00) e 34 (11/05/2025 16:00), caracterizado por uma concentração elevada de conexões e um comportamento atípico em relação aos demais horários. Esse padrão contrasta com o observado na Figura 20, referente a um dia de log normal no servidor, em que não se verifica a mesma densidade de conexões ou estrutura visual em nenhum intervalo de hora, reforçando a diferença entre o dia da invasão e dias normais de operação.

Essa abordagem temporal com animação permite uma visualização da evolução dos acessos com erro 404 ao longo do tempo, evidenciando momentos específicos em que pode haver indícios de atividades suspeitas. Desta forma, a representação gráfica por hora com animação e interatividade pode auxiliar visualmente na detecção de comportamentos atípicos com potencial a ser incorporada como uma ferramenta de apoio à análise de incidentes de segurança em tempo real.

As animações quadro a quadro dos grafos temporais, correspondentes aos experimentos desta seção, foram disponibilizadas em uma versão interativa das animações, permitindo a visualização dinâmica ao longo do tempo dos logs analisados. Essas animações podem ser acessadas no seguinte endereço: https://www.ifsc.usp.br/~flavia/mba-ia (Lisboa, 2025). Além de complementar a análise, esse recurso com animações evidenciou a aplicabilidade prática da modelagem proposta, ao permitir observar de forma clara a evolução dos padrões de acesso e possíveis indícios de ataques, oferecendo subsídios para uma aplicação objetiva de análise em tempo real dos logs.

Figura 19 – Sequência de quadros horários representando grafos das requisições de acesso com erro 404 nos logs do dia de invasão




21
28
33
34

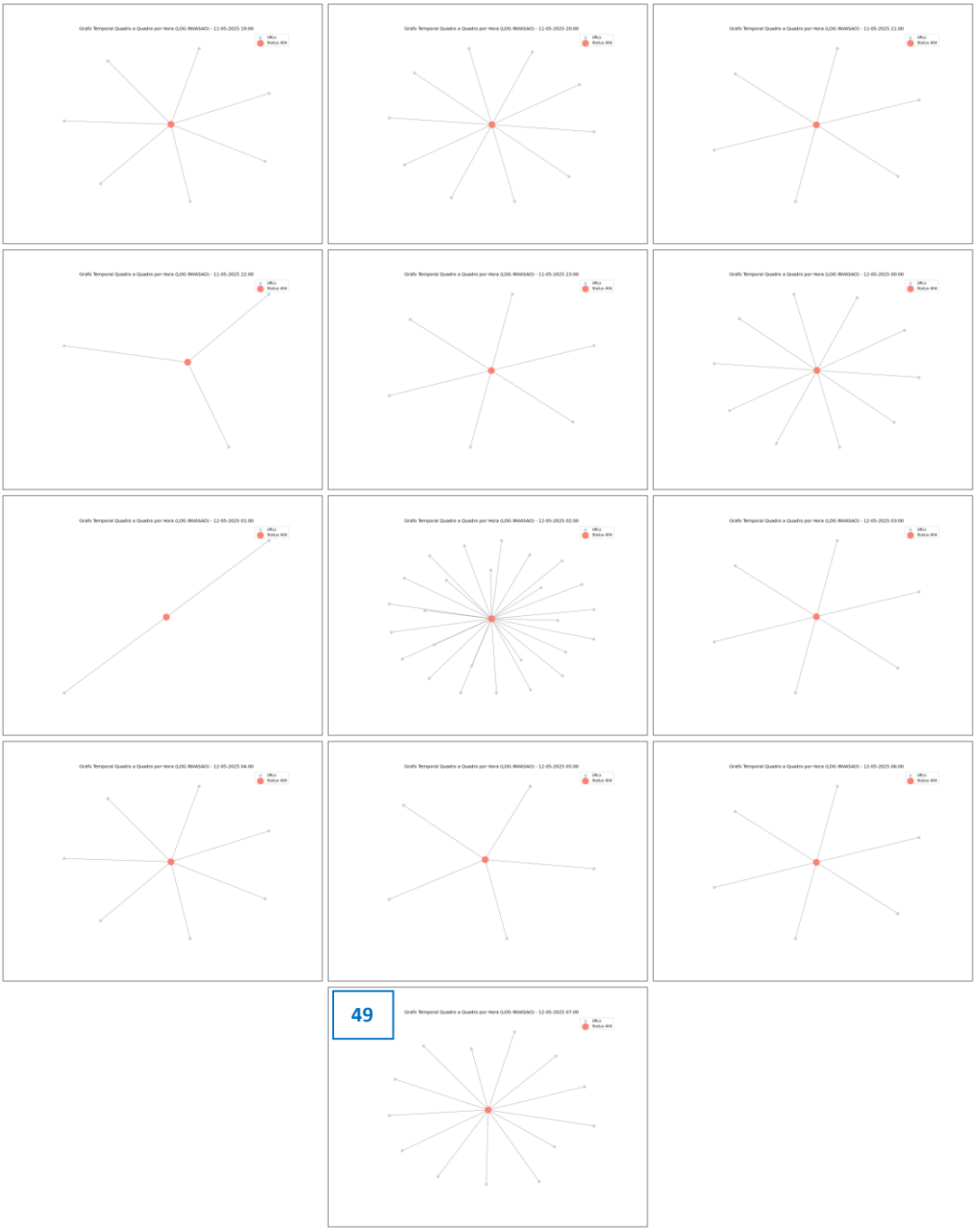
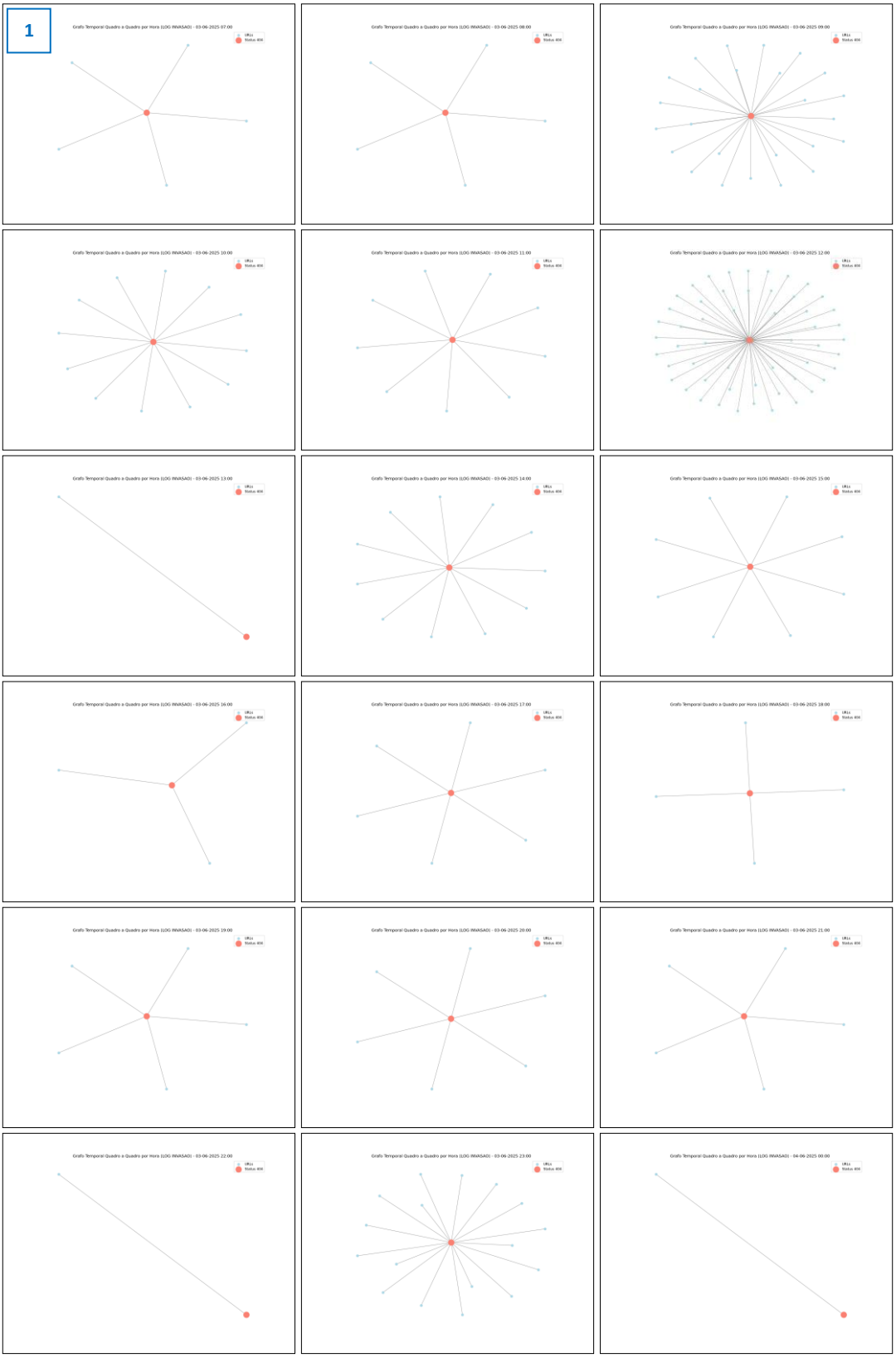
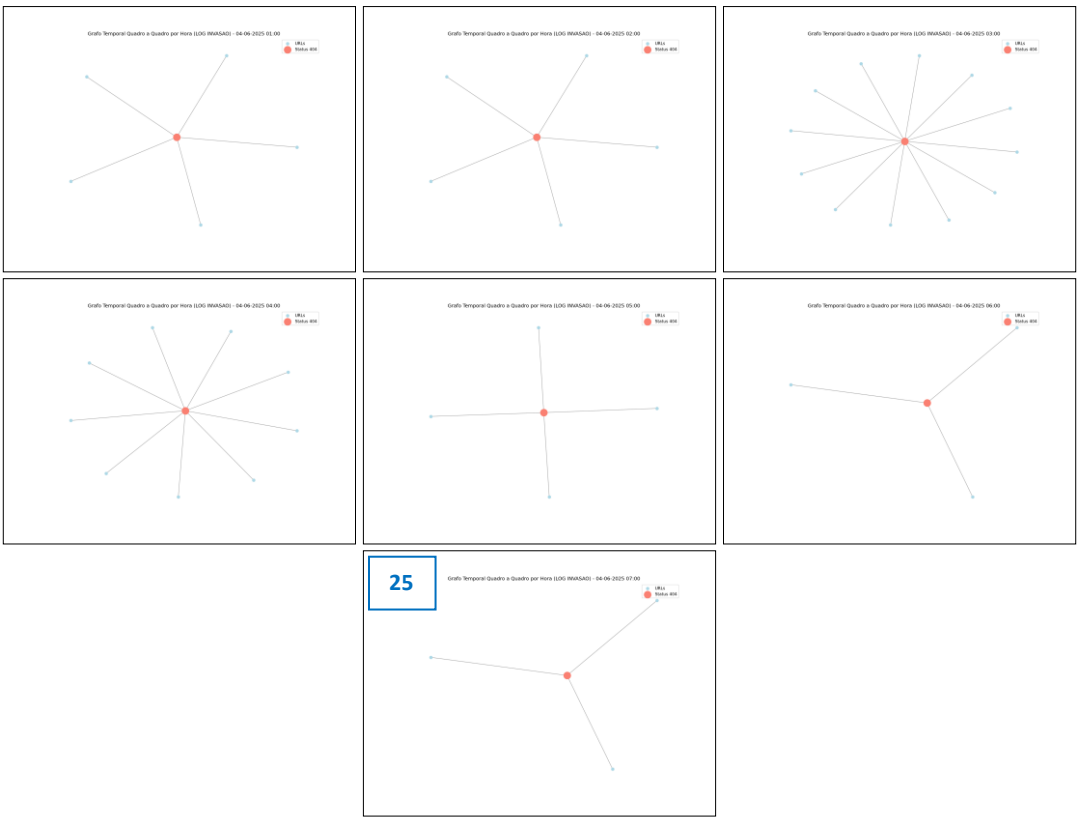

Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 11-05-2025 19:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 11-05-2025 20:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 11-05-2025 21:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 11-05-2025 22:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 11-05-2025 23:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 00:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 01:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 02:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 03:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 04:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 05:00
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 06:00
49
Grafo Temporal Quadro a Quadro por Hora (LOG-INVASAO) - 12-05-2025 07:00

Figura 20 – Sequência de quadros horários representando grafos das requisições de acesso com erro 404 em um dia normal de acessos ao servidor

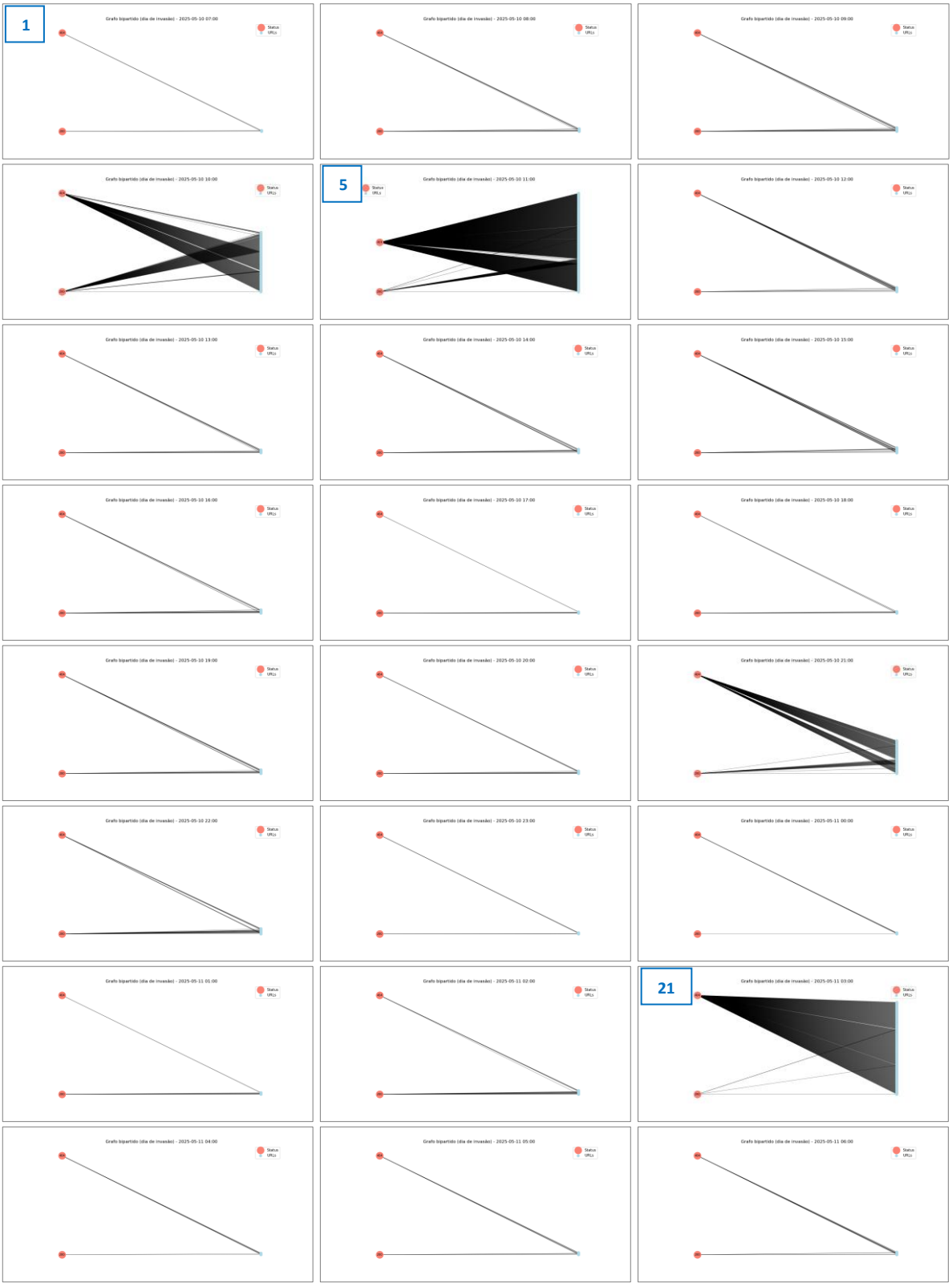


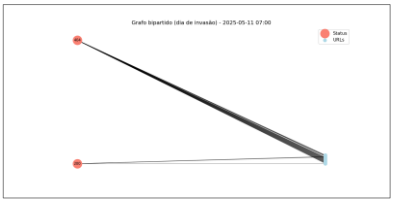


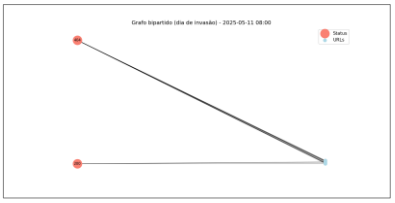
Ainda nas ilustrações para os experimentos com grafos temporais, a Figura 21 apresenta as sequências de imagens quadro a quadro em que cada quadro tem uma representação de grafos bipartidos dos códigos de resposta 200 e 404 com as URLs acessadas, registradas em um intervalo de uma hora dos logs do dia de invasão.

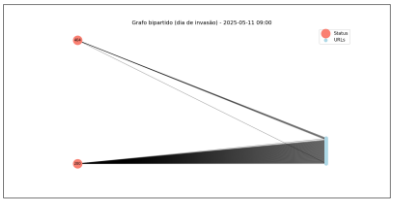
E novamente, agora nessa modelagem para grafos bipartidos, nota-se o mesmo padrão visual distinto já verificado nos grafos modelados anteriormente (Figura 19), também nos quadros 5 (11/05/2025 11:00), 21 (11/05/2025 03:00), 28 (11/05/2025 10:00), 33 (11/05/2025 15:00) e 34 (11/05/2025 16:00). Nesse modelo, a alta concentração de arestas para o código de resposta 404 corrobora ainda mais a evidência de acessos anômalos durante o período do ataque, contrastando com a baixa densidade de conexões associadas ao código 200, o que reforça a presença de um padrão de comportamento atípico característico de tentativas de exploração.

Figura 21 – Sequência de quadros horários representando grafos bipartidos IP x URL nos logs do dia de invasão

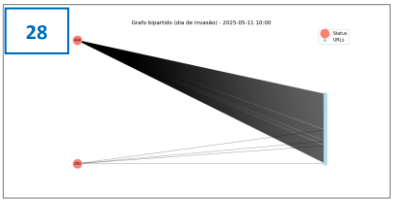


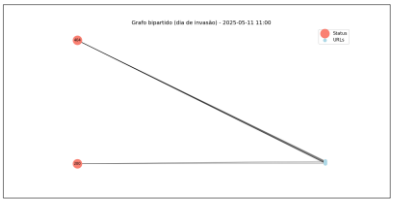

Grafo bipartido (dia de invasão) - 2025-05-11 07:00
Status
URLs

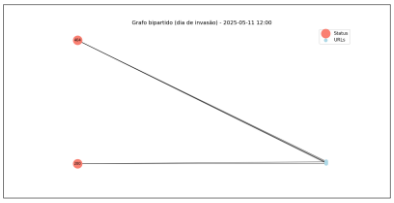

Grafo bipartido (dia de invasão) - 2025-05-11 08:00
Status
URLs

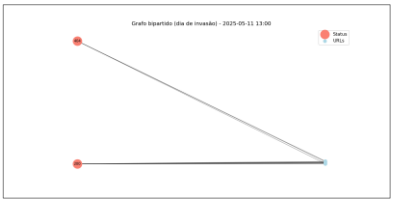

Grafo bipartido (dia de invasão) - 2025-05-11 09:00
Status
URLs

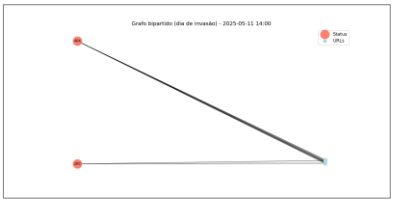
28


Grafo bipartido (dia de invasão) - 2025-05-11 10:00
Status
URLs

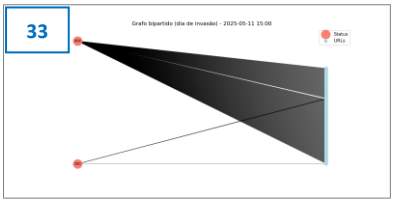

Grafo bipartido (dia de invasão) - 2025-05-11 11:00
Status
URLs


Grafo bipartido (dia de invasão) - 2025-05-11 12:00
Status
URLs

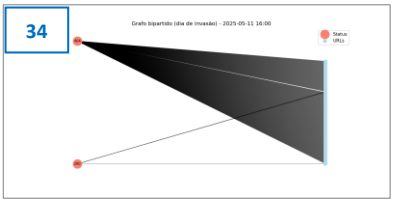

Grafo bipartido (dia de invasão) - 2025-05-11 13:00
Status
URLs

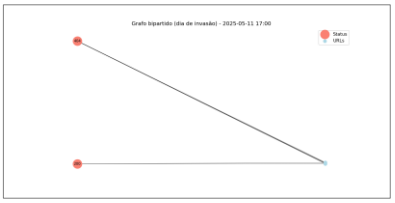

Grafo bipartido (dia de invasão) - 2025-05-11 14:00
Status
URLs

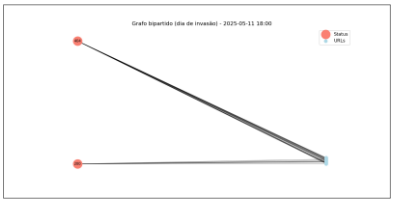
33

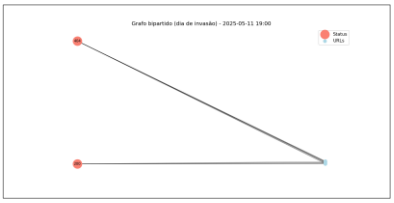

Grafo bipartido (dia de invasão) - 2025-05-11 15:00
Status
URLs

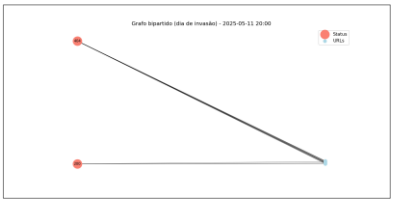
34

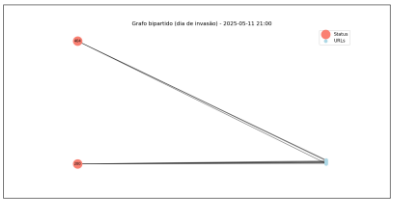

Grafo bipartido (dia de invasão) - 2025-05-11 16:00
Status
URLs

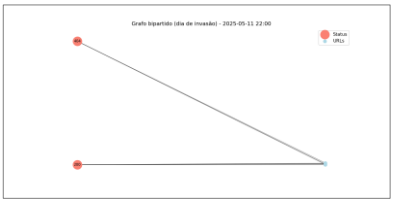

Grafo bipartido (dia de invasão) - 2025-05-11 17:00
Status
URLs

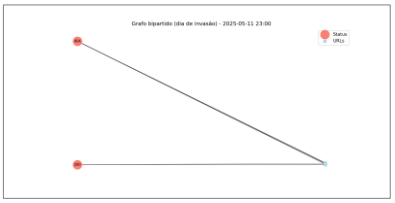

Grafo bipartido (dia de invasão) - 2025-05-11 18:00
Status
URLs

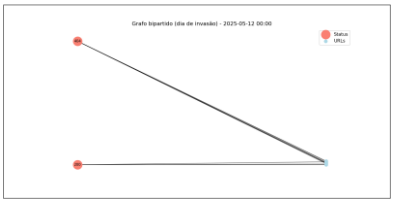

Grafo bipartido (dia de invasão) - 2025-05-11 19:00
Status
URLs

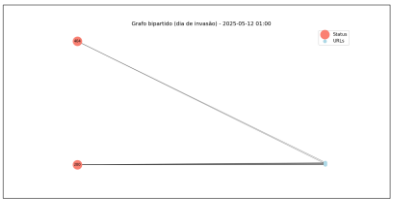

Grafo bipartido (dia de invasão) - 2025-05-11 20:00
Status
URLs

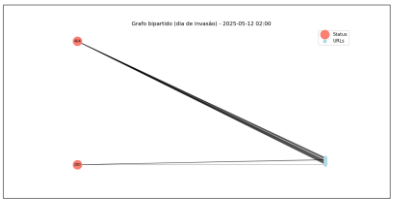

Grafo bipartido (dia de invasão) - 2025-05-11 21:00
Status
URLs

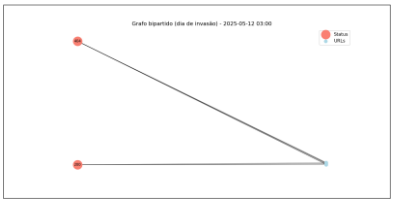

Grafo bipartido (dia de invasão) - 2025-05-11 22:00
Status
URLs

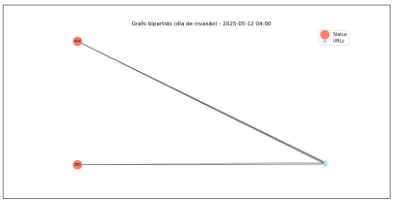

Grafo bipartido (dia de invasão) - 2025-05-11 23:00
Status
URLs

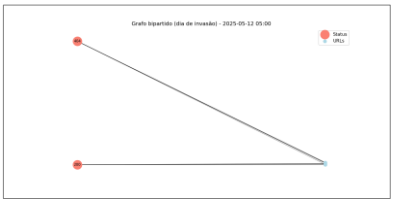

Grafo bipartido (dia de invasão) - 2025-05-12 00:00
Status
URLs

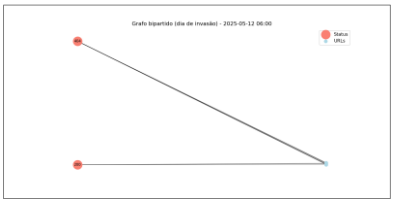

Grafo bipartido (dia de invasão) - 2025-05-12 01:00
Status
URLs


Grafo bipartido (dia de invasão) - 2025-05-12 02:00
Status
URLs

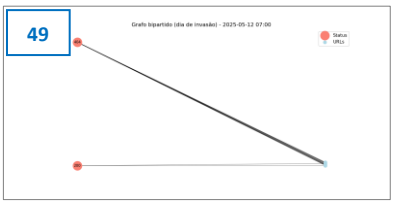

Grafo bipartido (dia de invasão) - 2025-05-12 03:00
Status
URLs


Grafo bipartido (dia de invasão) - 2025-05-12 04:00
Status
URLs


Grafo bipartido (dia de invasão) - 2025-05-12 05:00
Status
URLs


Grafo bipartido (dia de invasão) - 2025-05-12 06:00
Status
URLs

49


Grafo bipartido (dia de invasão) - 2025-05-12 07:00
Status
URLs

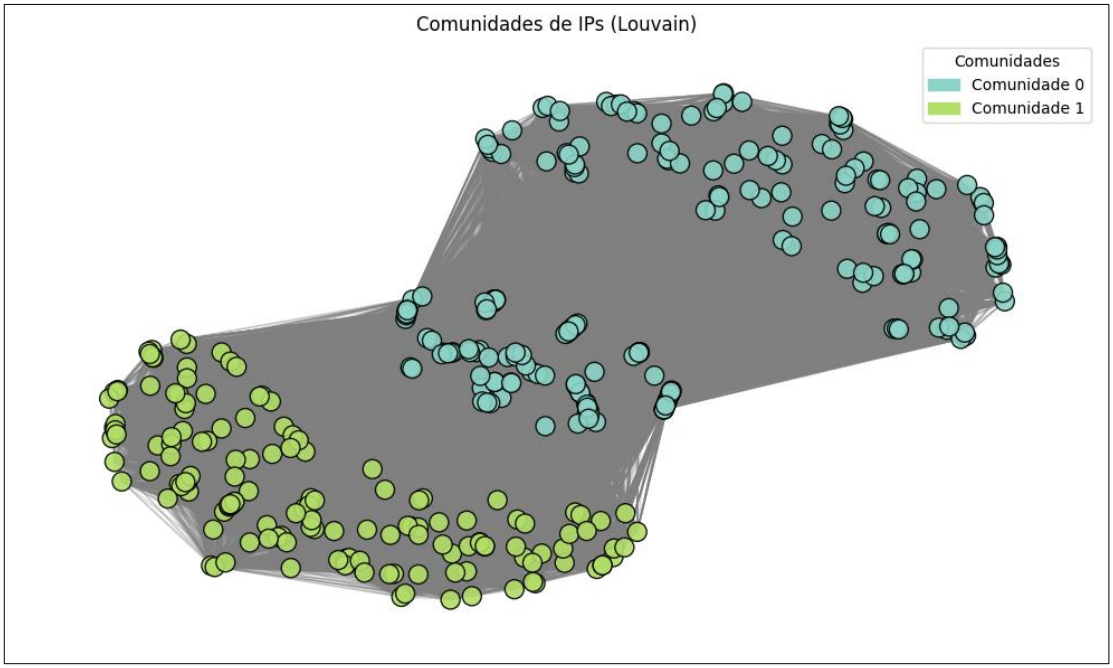
5.4 EXPERIMENTAÇÃO 4: Grafos com Detecção de Comunidades

O objetivo deste experimento foi identificar padrões de comportamento entre os IPs a partir das respostas do servidor, utilizando a detecção de comunidades. A análise foi conduzida sobre o log da invasão por SQLi registrada no Servidor A1.

Para a construção do grafo de comunidades, inicialmente foi criado um grafo bipartido IP ↔ código de resposta HTTP, considerando apenas os códigos HTTP 200 (acessos bem-sucedidos) e HTTP 404 (recursos inexistentes). Essa projeção resultou em um grafo homogêneo contendo apenas IPs, conectados entre si quando compartilhavam respostas semelhantes do servidor. Em seguida, aplicou-se o algoritmo Louvain para identificar comunidades densamente conectadas, ou seja, grupos de IPs com padrões similares de comportamento.

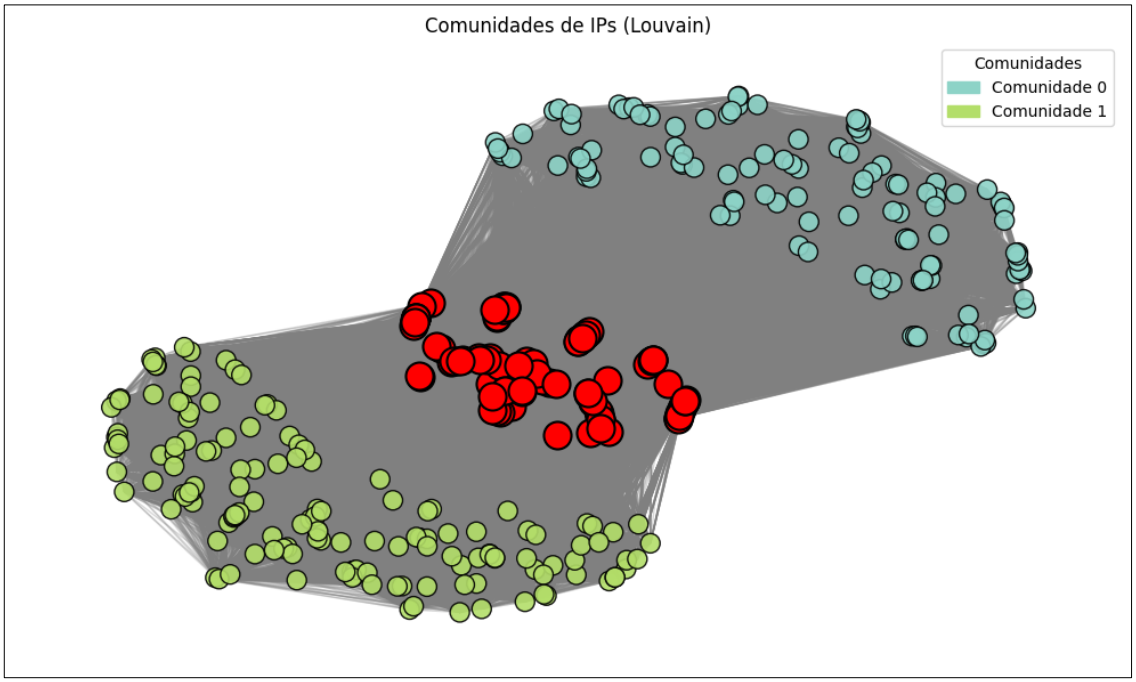
A Figura 22 ilustra essas comunidades, evidenciando dois grupos principais de IPs. Cada ponto do grafo representa um IP, e as cores distinguem as comunidades detectadas. As numerosas conexões internas (arestas em cinza) reforçam a coesão de comportamento dentro de cada grupo, o que pode estar relacionado a diferentes perfis de atividade: uma comunidade possivelmente associada a acessos legítimos (respostas 200) e outra caracterizada por acessos que geraram grande volume de erros 404, sugerindo comportamentos de varredura.

Figura 22 – Grafo de comunidades de IP usando Louvain do log do dia da invasão, com filtro para códigos de resposta HTTP 200 e 404



Além da segmentação por comunidades, foi realizada uma análise de centralidade para identificar IPs com maior influência na estrutura da rede. Utilizando-se da métrica de centralidade de intermediação (*betweenness centrality*) pode-se evidenciar os nós que atuam como pontes entre diferentes grupos. Os IPs com valores acima do percentil 90 dessa métrica foram destacados em vermelho no grafo (Figura 23). Especificamente, no caso da invasão por SQLi analisada, os nós destacados em vermelho podem representar IPs que transitaram entre comportamentos suspeitos (múltiplas tentativas com resposta 404) e ações bem-sucedidas (com resposta 200), sendo, portanto, candidatos a fontes efetivas da exploração. Esses IPs podem incluir tanto agentes automatizados em varredura quanto aqueles que realizaram de fato a injeção SQL com sucesso.

Figura 23 – Grafo de comunidades de IP usando Louvain do log do dia da invasão, com nós de maior centralidade destacados em vermelho



Complementando a análise dos grafos, o histograma apresentado na Figura 24 apresenta a distribuição dos códigos de resposta HTTP por comunidade do grafo da Figura 22. Nota-se que a comunidade 0 concentra um número elevado de erros 404, o que corrobora a hipótese de comportamento anômalo, enquanto a comunidade 1 apresenta predominantemente respostas 200, associadas a acessos bem-sucedidos. Na Figura 25, o histograma mostra o grau de centralidade média por comunidade. Observa-se que a comunidade 0 possui um grau de centralidade superior, o que indica que os IPs desse grupo estão mais interconectados, sugerindo

padrões de acesso mais intensos e possivelmente automatizados, enquanto a comunidade 1 apresenta uma centralidade ligeiramente menor, compatível com acessos mais dispersos e típicos de tráfego legítimo.

Figura 24 – Histograma de ocorrências para o código de resposta HTTP por comunidade de IPs, aplicando Louvain (grafo representado na Figura 22)

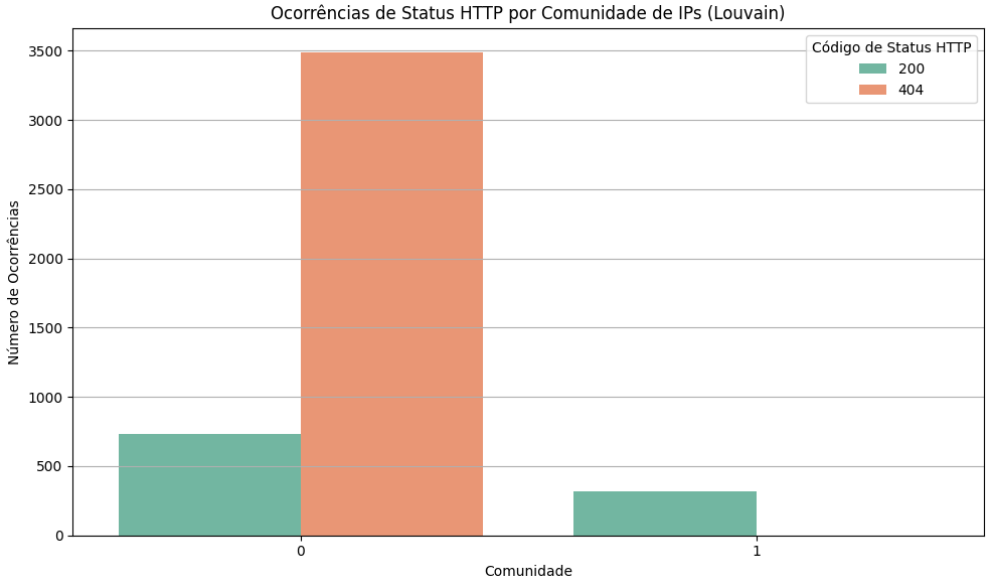
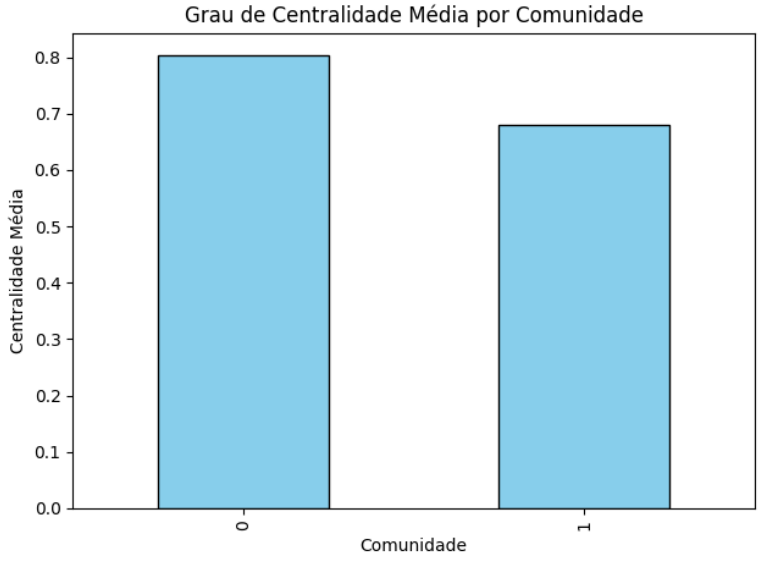


Figura 25 – Histograma da métrica do grau de centralidade média por comunidade de IPs, aplicando Louvain (grafo representado na Figura 22)



Esses resultados indicam como a detecção de comunidades, associada à análise de métricas como centralidade e distribuição de códigos HTTP, pode auxiliar na diferenciação entre comportamentos normais e padrões potencialmente maliciosos nos logs de servidores web.

6 RESULTADOS E CONCLUSÕES

As investigações e análises desenvolvidas neste trabalho buscaram explorar o potencial da modelagem por grafos para a interpretação de padrões presentes nos logs de acesso de servidores Web Apache. A seguir, são discutidos os principais resultados obtidos e como eles se relacionam com os objetivos e hipóteses apresentados ao longo do trabalho.

6.1 Análise dos Resultados

Os grafos bipartidos, tripartidos e temporais construídos a partir dos logs – tanto de dias normais quanto do log de invasão por SQLi – revelaram padrões estruturais que não apenas distinguem cenários de ataque de situações normais, como também podem ser generalizados para outros tipos de ataques a servidores web. Observou-se que, em casos de invasão, há uma etapa inicial caracterizada pelo envio de um grande volume de requisições ao servidor com o objetivo de identificar vulnerabilidades. Esse comportamento gera padrões incomuns nos logs, como acessos repetitivos a URLs inexistentes (muitos erros de código 404) ou tentativas sucessivas de exploração de recursos específicos, os quais se refletem visualmente nos grafos construídos e analisados.

O estudo de caso desenvolvido concentrou-se na análise de um ataque de injeção de SQL, cuja “assinatura” gráfica se caracteriza por um intenso volume de requisições sucessivas e pela ocorrência de numerosos erros 404, resultantes de tentativas automatizadas de varredura de URLs em busca de vulnerabilidades. É importante destacar que diferentes tipos de ataques tendem a produzir assinaturas estruturais distintas nos grafos — por exemplo, um ataque de negação de serviço distribuído (DDoS) ou a exploração de uma vulnerabilidade específica poderiam gerar padrões topológicos e temporais diversos. Dessa forma, ainda que nos experimentos tenha sido utilizado um caso de SQLi como base, a mesma abordagem pode ser aplicada para outros tipos de ataque, como varredura de diretórios, exploração de vulnerabilidades conhecidas e até mesmo ataques de negação de serviço. Nesse sentido, trabalhos futuros poderiam explorar a construção de uma biblioteca de assinaturas gráficas associadas a diferentes categorias de ameaças, contribuindo para o desenvolvimento de sistemas de detecção generalizáveis.

Outra discussão relevante diz respeito à comparação entre as ferramentas de análise exploratória tradicional, comumente aplicadas a dados de logs, e a análise baseada em grafos. As visualizações exploratórias por histogramas e *heatmaps* mostraram-se úteis para fornecer

uma visão quantitativa dos logs, como a frequência de códigos HTTP ou a distribuição de requisições ao longo do tempo. No entanto, essas técnicas não representam relações entre os elementos. A análise utilizando grafos, por outro lado, adiciona uma camada relacional fundamental. Em um histograma, a principal informação apresentada é a de quantidade – ou seja, quantas vezes cada código de resposta HTTP ocorreu. Trata-se, basicamente, de uma informação quantitativa isolada. Já na representação por grafos, é possível acessar informações adicionais, pois eles permitem representar relações. É possível, ainda, visualizar a associação entre um determinado código de resposta e qual URL foi acessada que gerou esse código (indicando um erro ou um acesso bem-sucedido). Essas relações carregam informações importantes que ajudam a entender padrões de comportamento nos acessos. Por exemplo, enquanto um histograma pode mostrar que ocorreram 300 erros com o código 404, um grafo pode evidenciar que 150 desses erros vieram de uma mesma categoria de URLs. Isso pode indicar uma tentativa automatizada de varredura por diretórios, como parte de um ataque, algo que não seria perceptível apenas com a visualização do histograma. Por outro lado, se os erros 404 estiverem todos relacionados a uma única URL, isso pode ser resultado da exclusão de um arquivo do servidor e não de um ataque. Essa distinção é mais facilmente perceptível nos grafos, uma vez que eles mostram claramente a conexão entre os erros e os recursos acessados.

Além disso, a análise da topologia dos grafos permite detectar comunidades, nós centrais, caminhos recorrentes e padrões estruturais complexos. Os grafos também possibilitam agregar e analisar diversas relações entre atributos dos logs, como IP x URL, IP x código HTTP, URL x método HTTP, entre outros. A modelagem por grafos k-partidos, em especial, oferece uma estrutura relacional mais informativa que vai além da contagem bruta de eventos, permitindo análises qualitativas e quantitativas mais profundas. Dessa forma, os grafos complementam os histogramas, fornecendo uma camada adicional de interpretação e detecção de padrões que seria difícil ou impossível de obter apenas com métodos estatísticos convencionais.

Outro resultado relevante foi obtido com os experimentos envolvendo grafos de similaridade (G1 – limite de similaridade e G2 – k-NN), que reforçaram a aplicabilidade da técnica para identificar dias com comportamento anômalo. O modelo G1, com limite de similaridade por cosseno > 0,6, destacou apenas os dias fortemente correlacionados ao log de invasão, formando agrupamentos coesos e específicos. Já o modelo G2, baseado em k-NN, gerou redes mais conectadas, úteis para análise exploratória, evidenciando dias com menor similaridade, mas ainda próximos ao log de ataque. Esses resultados mostram que os dois

modelos podem ser complementares: G1 para identificar padrões mais específicos e G2 para expandir a análise em cenários com baixa similaridade geral.

Além dessas abordagens, os grafos temporais com animação também se mostraram uma ferramenta relevante para a análise dinâmica dos logs. Essa técnica possibilitou visualizar a evolução dos acessos ao longo do tempo, destacando picos de requisições com erros, como o 404, que podem indicar comportamentos suspeitos, tais como varreduras automatizadas ou etapas iniciais de ataques de negação de serviço. As animações quadro a quadro permitem observar de maneira progressiva como esses eventos se distribuem no tempo, facilitando a identificação de momentos críticos que poderiam demandar atenção imediata. Além de complementar a análise, a utilização de animações temporais reforça a aplicabilidade prática da modelagem proposta, ao permitir acompanhar de forma clara a evolução de padrões de acesso e possíveis indícios de ataques, fornecendo subsídios objetivos para a tomada de decisão.

6.2 Conclusões

Os modelos temporais destacaram-se como a abordagem mais eficaz entre as técnicas avaliadas, por permitirem acompanhar a evolução das interações nos logs e identificar variações anômalas ao longo do tempo. Além de sua capacidade descritiva, esses modelos oferecem potencial para aplicações em tempo real, nas quais métricas fixas ou comparações entre grafos sucessivos possam ser utilizadas para detectar dissimilaridades em relação ao comportamento considerado normal e, assim, emitir alertas automáticos diante de indícios de atividades suspeitas.

Por fim, destaca-se que a aplicação de modelagem por grafos aos logs de acesso demonstrou ser simples, visual e de baixo custo computacional, diferentemente de muitas ferramentas de mercado que exigem infraestrutura robusta para análise. Essa característica não apenas torna a abordagem acessível, como também já a torna útil para atividades práticas de administração de servidores web, oferecendo uma forma intuitiva de detectar padrões anômalos e comportamentos suspeitos. Além disso, uma perspectiva futura promissora é a adaptação dessa metodologia para monitoramento em tempo real. A implementação de um dashboard integrado, com atualizações periódicas dos grafos temporais e alertas automáticos para padrões anômalos, é tecnicamente viável e poderia ampliar significativamente a capacidade de resposta frente a incidentes de segurança, consolidando ainda mais a relevância da abordagem aqui proposta.

Dessa forma, os resultados obtidos reforçam que a modelagem por grafos aplicada à análise de logs de servidores web não apenas complementa as técnicas tradicionais de exploração de dados, como também oferece uma abordagem relacional capaz de revelar padrões relevantes e de fácil interpretação visual. Embora ainda existam possibilidades de aprimoramento e evolução para aplicações em tempo real, os experimentos realizados demonstram que a proposta é viável e pode servir como base para futuras pesquisas e desenvolvimentos práticos na área de monitoramento e segurança de servidores.

REFERÊNCIAS

AGGARWAL, C.C.; ZHAI, C. A survey of text clustering algorithms. In: **Mining text data**. Boston, MA: Springer US, 2012. p. 77-128.

AKOGLU, L.; TONG, H.; KOUTRA, D. Graph based anomaly detection and description: a survey. **Data mining and knowledge discovery**, v. 29, p. 626-688, 2015.

Apache Logging, **Documentação oficial do Apache Logging**. https://httpd.apache.org/docs/current/logs.html. Acesso em: 15/07/2025.

BAHASSI, H. et al. Toward an exhaustive review on machine learning for cybersecurity. **Procedia Computer Science**, v.203, p.583-587, 2022.

BARANWAL, A.K. Approaches to detect SQL injection and XSS in web applications. **EECE 571b**, **Term Survey paper**, 2012.

BHANAGE, D.A.; PAWAR, A.V; KOTECHA, K. IT infrastructure anomaly detection and failure handling: a systematic literature review focusing on datasets, log preprocessing, machine & deep learning approaches and automated tool. **IEEE Access**, v.9, p.156392-156421, 2021.

BLONDEL, V.D. et al. Fast unfolding of communities in large networks. **Journal of statistical mechanics: theory and experiment**, v. 2008, n. 10, p. P10008, 2008.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. **ACM Computing Surveys (CSUR)**, v. 41, n. 3, p. 1-58, 2009.

CHORAŚ, M.; KOZIK, R. Machine learning techniques applied to detect cyber attacks on web applications. **Logic Journal of IGPL**, v. 23, n. 1, p. 45-56, 2015.

COMIN, C.H. et al. Complex systems: features, similarity and connectivity. **Physics Reports**, v. 861, p. 1-41, 2020.

DIESTEL, R. **Graph theory (6th ed.)**. Graduate Texts in Mathematics, vol. 173. Springer Nature, 2025.

FORTUNATO, S. Community detection in graphs. **Physics reports**, v. 486, n. 3-5, p. 75-174, 2010.

HE, P. et al. An evaluation study on log parsing and its use in log mining. In: **2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)**. IEEE, 2016. p. 654-661.

HE, P. et al. Towards automated log parsing for large-scale log data analysis. **IEEE Transactions on Dependable and Secure Computing.** v. 15, n. 6, p. 931-944, 2017.

HE, P. et al. Characterizing the natural language descriptions in software logging statements. In: **Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering**. 2018, p. 178-189.

HE, S. et al. Experience report: system log analysis for anomaly detection. In: **IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)**. Ottawa, Canada: IEEE, 2016. p. 207-218.

HOANG, X.D. Detecting common web attacks based on machine learning using web log. In: **International Conference on Engineering Research and Applications (ICERA 2020**). Springer International Publishing, 2021. p. 311-318.

HOLME, P.; SARAMÄKI, J. Temporal networks. **Physics reports**, v. 519, n. 3, p. 97-125, 2012.

HUSSEIN, S.A.; RÉPÁS, S.R. Anomaly detection in log files based on machine learning techniques. **Journal of Electrical Systems**, v.20, n.3, p. 1299-1311, 2024.

JACOMY, M. et al. ForceAtlas2: a continuous graph layout algorithm for handy network visualization designed for the Gephi software. **PloS one**, v. 9, n. 6, p. e98679, 2014.

JYOTHY, P. et al. Threat hunting in system using log analysis. In: **International Conference on Knowledge Engineering and Communication Systems (ICKECS)**. Chikkaballapur, India: IEEE, 2024. p. 1-6.

KILINCER, I.F.; ERTAM, F.; SENGUR, A. Machine learning methods for cyber security intrusion detection: Datasets and comparative study. **Computer Networks**, v.188, p.107840, 2021.

LANDAUER, M. et al. Dynamic log file analysis: An unsupervised cluster evolution approach for anomaly detection. C**omputers & Security**, v. 79, p. 94-116, 2018.

LISBOA, F.O.S.S. **Experimentação TCC do MBA em IA e Big Data (Turma 4): Modelando Grafos com LOGS de Acesso Web Apache**. São Carlos: IFSC/USP, 2025. Disponível em: https://www.ifsc.usp.br/~flavia/mba-ia. Acesso em: 15/09/2025.

LogPAI, **Log Parsing, Analysis, and Intelligence**, https://logpai.com. Acesso em: 02/02/2025.

MARCACINI, R. M. **Notebook: Similaridade entre Documentos com TF-IDF e n-gramas**. Notas de aula do curso MBA em IA e Big Data – Turma 4 – Curso 11: Inteligência Analítica usando Mineração de Textos. São Carlos: IFSC/USP, 2025. Não publicado.

MPMT, **Novo ataque: site do TSE está fora do ar; hacker alega autoria**. https://www.mpmt.mp.br/portalcao/news/1217/157327/novo-ataque-site-do-tse-esta-fora-do-ar-hacker-alega-autoria/14. Acesso em: 24/03/2025.

NEWMAN, M.E.J. Properties of highly clustered networks. **Physical Review E**, v. 68, n. 2, p. 026121, 2003.

NEWMAN, M.E.J. **Networks**. Oxford university press, 2018.

OGlobo Tecnologia, **Elon Musk diz que X é alvo de um enorme ciberataque e diz suspeitar que a origem é ucraniana**. https://oglobo.globo.com/economia/tecnologia/noticia/

2025/03/10/musk-diz-que-x-e-alvo-de-um-enorme-ciberataque.ghtml. Acesso em: 10/03/2025.

Olhar Digital, **USP passa por instabilidades em seu site, mas nega ataque hacker**. https://olhardigital.com.br/2025/03/11/seguranca/usp-passa-por-instabilidades-em-seu-site-mas-nega-ataque-hacker, 2025a. Acesso em: 12/03/2025.

_____, **Site do aeroporto de Guarulhos sai do ar após ataque hacker**. https://olhardigital.com.br/2025/03/15/seguranca/site-do-aeroporto-de-guarulhos-sai-do-ar-apos-suposto-ataque-hacker, 2025b. Acesso em: 15/03/2025.

OpenAI. **ChatGPT**, https://chatgpt.com. 2025.

OWASP Foundation, **OWASP Top Ten Web Application Security Risks**, https://owasp.org/www-project-top-ten. Acesso em: 22/07/2025.

SAKET, S.K.; PANDAY, A.; SINGH, G.N. A Review on Graph-Theoretic Techniques for Web Log Data. **Journal of Artificial Intelligence and Computer Science**, v. 1, n. 1, 2024.

Sematext Blog, **10 Best Apache Log Analysers: Free & Paid Tools (2023 Comparison)**. https://sematext.com/blog/apache-log-analyzers. Acesso em: 15/07/2025.

Security Report, **Força Aérea Brasileira confirma ataque DDoS**. https://securityleaders.com.br/forca-aerea-brasileira-confirma-ataque-ddos. Acesso em: 20/03/2025.

SEYYAR, M.B; ÇATAK, F.Ö.; GÜL, E. Detection of attack-targeted scans from the Apache HTTP Server access logs. **Applied computing and informatics**, v. 14, n. 1, p.28-36, 2018.

SHAUKAT, K. et al. A survey on machine learning techniques for cyber security in the last decade. **IEEE Access**, v.8, p.222310-222354, 2020.

VAARANDI, R.; PIHELGAS, M. Using security logs for collecting and reporting technical security metrics. In: **2014 IEEE Military Communications Conference**. IEEE, 2014, p. 294-299.

VAARANDI, R.; PIHELGAS, M. Logcluster: a data clustering and pattern mining algorithm for event logs. In: **2015 11th International Conference on Network and Service Management (CNSM)**. IEEE, 2015. p. 1-7.

W3C Standards, **Common Log File Format**, https://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format. Acesso em: 26/01/2025.

ZHU, J. et al. Loghub: a large collection of system log datasets for AI-driven log analytics. In: **2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)**. IEEE, 2023. p. 355-366.