CENTRO ESTADUAL DE EDUCAÇÃO
TECNOLÓGICA PAULA SOUZA
Faculdade de Tecnologia de Jundiaí – Dep. Ary Fossen

Defesa Cibernética

PROJETO INTEGRADOR II
CENÁRIO:

Implementação de um Algoritmo Criptográfico

ALEX SANTOS DA SILVA
BRUNO NEEMIAS ALVES MOTA
LUCAS BEZERRA DA SILVA
MARCELA DE CASTRO SANTOS
MELQUISEDEC SOUZA ASSUNÇÃO

Jundiaí
2025

1. Introdução

1.1. Contextualização do tema

No cenário digital, onde a troca e o armazenamento de informações ocorrem em grandes quantidade, a criptografia e a segurança cibernética surgiram como pilares fundamentais para a proteção de dados e a manter a confiança em sistemas computacionais. A crescente evolução das ameaças cibernéticas, que incluem desde ataques de phishing e ransomware até invasões complexas a ativos e sistemas de elevado valor, exige uma abordagem proativa e robusta para garantir a integridade, confidencialidade e disponibilidade das informações(NIST, 2025). A criptografia, em sua essência, é a ciência de proteger informações por meio de códigos, transformando dados legíveis em formatos ilegíveis para qualquer pessoa que não possua a chave de decifração. Essa técnica é indispensável para garantir a privacidade em comunicações, a segurança em transações financeiras e a proteção de dados sensíveis armazenados em bancos de dados.

A segurança cibernética abrange um conjunto mais amplo de práticas, tecnologias e processos projetados para proteger redes, dispositivos, programas e dados de ataques,

danos ou acessos não autorizados. Ela não se limita apenas à criptografia, mas engloba também a gestão de identidades e acessos(IAM), a segurança de redes, a segurança de aplicações e a recuperação de desastres. Com a interconexão global e a dependência crescente de sistemas digitais, a segurança cibernética eficaz torna-se essencial em todos os setores da sociedade. Falhas na segurança podem resultar em perdas financeiras significativas, danos à reputação,
comprometimento da privacidade e até mesmo ameaças à segurança nacional (FORTINET, 2025).

Nesse contexto, a detecção e prevenção de acessos não autorizados tornam-se componentes críticos de qualquer estratégia de segurança cibernética. A detecção envolve a identificação de atividades suspeitas ou anômalas que possam indicar uma tentativa de invasão ou comprometimento do sistema. Isso é frequentemente realizado por meio de sistemas de detecção de intrusão (IDS), monitoramento de logs e análise de comportamento de usuários e entidades (UEBA). A prevenção, por sua vez, foca em implementar barreiras e controles para impedir que acessos não autorizados ocorram em primeiro lugar. Isso inclui o uso de autenticação multifator

(MFA), firewalls, sistemas de controle de acesso rigorosos e a aplicação de políticas de segurança robustas. A capacidade de detectar e prevenir proativamente tais acessos é vital para minimizar o impacto de incidentes de segurança, proteger ativos digitais e manter a continuidade das operações, especialmente em sistemas que gerenciam informações sensíveis, como um Sistema de Gestão de Biblioteca (FORENSE.IO, 2025).

1.2. Objetivos do Projeto

1.2.1 Objetivo Geral

Implementar um algoritmo criptográfico robusto e eficiente, integrado ao Sistema de Gestão de Biblioteca da Nexus Library desenvolvido no Projeto Integrador I, visando aprimorar a segurança e a proteção dos dados sensíveis dos usuários e do acervo, em conformidade com as melhores práticas de segurança da informação e requisitos legais.

1.2.2 Objetivos Específicos

Qualidade e Testes de Software: Descrever e aplicar técnicas de programação segura no desenvolvimento da integração do algoritmo criptográfico, focando na proteção dos dados de entrada dos usuários (e.g., credenciais) e no armazenamento seguro das informações no banco de dados (MySQL), utilizando criptografia para garantir confidencialidade e integridade. Desenvolver e executar planos de testes específicos para validar a eficácia do algoritmo criptográfico e a solidez das técnicas de programação segura implementadas, incluindo testes de penetração e análise de vulnerabilidades.

Legislação Aplicada à Tecnologia da Informação: Descrever como a implementação do algoritmo criptográfico e as práticas de segurança de dados adotadas no Sistema de Gestão de Biblioteca da Nexus Library estão em conformidade com a legislação vigente sobre segurança e proteção de dados (e.g., LGPD - Lei Geral de Proteção de Dados Pessoais no Brasil), abordando aspectos como consentimento, direitos dos titulares e responsabilidades da empresa. Realizar uma análise de impacto sobre a proteção de dados (DPIA) para o novo componente criptográfico.

Inteligência Artificial: Investigar e propor a utilização de uma técnica específica de Inteligência Artificial (e.g., aprendizado de máquina para otimização de parâmetros criptográficos, detecção de padrões de ataque ou anomalias) para auxiliar no desenvolvimento ou aprimoramento do algoritmo criptográfico ou de seus mecanismos de defesa. Descrever metodologicamente como a Inteligência Artificial pode auxiliar nas questões de proteção de dados e na realização de simulações de proteção contra ataques cibernéticos.

2. Revisão bibliográfica

2.1 Algoritmos Criptográficos

A segurança da informação em sistemas computacionais é fundamentalmente sustentada pela criptografia, que se divide em duas categorias principais, a criptografia simétrica e a criptografia assimétrica.

A criptografia simétrica ou chave privada, utiliza uma única chave para cifrar e decifrar os dados, sendo o Advanced Encryption Standard (AES), o algoritmo mais amplamente adotado e recomendado globalmente. Sua eficiência e robustez o tornam ideal para a criptografia de grandes volumes de dados, como logs de sistema e informações de acervo. Seu funcionamento se baseia com a origem (X) que usa o processo específico e a senha secreta para converter a mensagem original em um formato ilegível. O receptor (Y) utiliza o processo oposto com a mesma senha para restaurar a mensagem ao seu estado inicial. Um invasor (Z), mesmo conhecendo o processo, é incapaz de decifrar sem a senha.

O AES, em comparação ao seu antecessor o Data Encryption Standard (DES) é muito mais seguro, que mesmo um

supercomputador que realiza bilhões de testes de chaves, demoraria trilhões de anos para quebrar um AES-128.

Em contraste, a criptografia assimétrica, exemplificada pelo algoritmo RSA (Rivest–Shamir Adleman), utiliza um par de chaves (pública e privada). Embora mais lento que o AES, o RSA é indispensável para a troca segura de chaves simétricas e para a implementação de Assinaturas Digitais, garantindo a autenticidade e o não-repúdio das informações. Este modelo oferece benefícios, como dispensar o compartilhamento da chave pessoal, atenuando as chances de ser interceptada, e viabilizar uma troca de dados segura com qualquer indivíduo que detenha a chave geral do receptor. Seu funcionamento se base com o destinatário (X), gera um par de chaves, uma pública e outra privada. A chave pública é divulgada, permitindo que qualquer remetente, como (Y), cifre mensagens destinadas a (X). Apenas (X) utilizando sua chave privada, pode decifrar a mensagem. Mesmo que um intruso (Z), conheça a chave pública, ele não conseguirá decifrar a mensagem sem a chave privada. Nem mesmo (Y), que cifrou a mensagem, poderá decifrá-la posteriormente.

2.1.1 Criptografia de Logs de Auditoria (AES-256)

O AES-256 no modo de encadeamento de blocos (e.g., CBC ou GCM) gera logs de auditoria contêm informações sensíveis sobre as operações do sistema (e.g., IPs, RAs, horários de acesso). O AES-256 é um padrão de criptografia simétrica rápido e inquebrável por força bruta com a tecnologia atual, garantindo a confidencialidade dos dados de log em repouso, em conformidade com a LGPD.

2.1.2 Armazenamento de Senhas (password_hash)

Funções de hashing seguras, como password_hash() do PHP (linguagem de script), que utiliza o algoritmo Bcrypt por padrão. As senhas jamais devem ser armazenadas de forma reversível. O uso de funções de hashing com salt e cost factor (como o Bcrypt) impede ataques de rainbow table e torna a quebra por força bruta inviável, mesmo em caso de comprometimento do banco de dados. Isso garante a irreversibilidade e a segurança das credenciais dos usuários.

2.1.3 Prevenção de SQL Injection (Prepared Statements)

A utilização de Prepared Statements e bind_param em todas as interações com o banco de dados (MySQL) previne ataques de SQL Injection, conforme as diretrizes da OWASP (Open Worldwide Application Security Project), esta é uma das vulnerabilidades mais comuns em aplicações web.

O uso de Prepared Statements separa o código SQL dos dados de entrada do usuário, garantindo que a entrada seja tratada sempre como dado e nunca como comando executável, prevenindo eficazmente a injeção de código malicioso.

O Quadro 1 apresenta as criptográfias e de segurança consideradas para o sistema Nexus Library.

Quadro 1 - Modelos considerados para serem desenvolvidos em um sistema de gerenciamento de bibliotecas

Tipo de Dado	**Mecanismo de Segurança**	**Algoritmo/Função**	**Propósito**
Senhas de Usuário	Hashing Irreversível	password_hash() (Bcrypt)	Confidencialidade e Irreversibilidade
Logs de Auditoria	Criptografia Simétrica	AES-256	Confidencialidade em Repouso
Interação com DB	Prevenção de Injeção	Prepared Statements/bind_param	Integridade e Prevenção de SQL Injection

2.2 Detecção e Prevenção de Ataques Cibernéticos

Com o avanço da tecnologia os usuários mal-intencionados estão cada vez mais sofisticados, atacando sistemas com mais força e táticas de disfarçar seus rastros. Para proteger desses intrusos existem vários tipos de defesa e umas delas são os Intrusion Detection System (IDS) e o Intrusion Protection System (IPS).

Para proteger desses intrusos existem vários tipos de defesa e umas delas são os Intrusion Detection System (IDS) e o Intrusion Protection System (IPS).

2.2.1 Intrusion Detection System (IDS)

O IDS é uma proteção no sistema que alertar o Administrador da rede que algo está errado no tráfego ou alguém está infiltrando no sistema. Existem tipos variados de IDS. Os mais comuns incluem:

- *Sistema de detecção de intrusão de rede (NIDS)*

Uma Solução NIDS é implantada dentro da rede em pontos estratégicos para monitorar a entrada e saída do tráfego de dados.

- *Sistema de detecção de intrusão de host (HIDS)*

Como o próprio nome já diz, o sistema instalado dentro da máquina específica, assim pode detectar uma ameaça interna, no caso que não vem de fora, ou pode descobrir ameaças do próprio host tentando infiltrar na rede.

- *Sistema de detecção de intrusão baseado em assinatura (SIDS)*

Uma solução SIDS monitora todos os pacotes na rede de uma organização e os compara com assinaturas de ataque em um banco de dados de ameaças conhecidas.

- *Sistema de detecção de intrusão baseado em anomalias (AIDS)*

Essa solução monitora o tráfego em uma rede e o comprar com uma linha de base predefinida considerada 'normal'. Ele detecta atividade e comportamento atípicos em toda a rede, incluindo largura de banda, dispositivos, portas e protocolos.

- *Sistema de detecção de intrusão de perímetro (PIDS)*

Uma solução de PIDS é colocada em uma rede para detectar tentativas de intrusão que ocorrem no perímetro das infraestruturas críticas das organizações.

- *Sistema de detecção de intrusão baseado em máquina virtual (VMIDS)*

Uma solução VMIDS detecta intrusões monitorando máquinas virtuais. Ele permite que as organizações monitorem o tráfego em todos os dispositivos e sistemas aos quais seus dispositivos estão conectados.

- *Sistema de detecção de intrusão baseado em pilha (SBIDS)*

O SBIDS é integrado ao protocolo de controle de transmissão/protocolo de internet (TCP/IP) de uma organização, que é usado como um protocolo de comunicação em redes privadas.

2.2.2 Intrusion Protection System (IPS)

Diferente do IDS que apenas detecta, os IPS são como uma barreira que protege o sistema, como o próprio nome diz. Eles analisam e barrem possíveis ameaças aos sistemas. A implantação de uma ferramenta de IPS permite que as organizações evitem ameaças avançadas, como ataque de negação de serviço (DoS), pishing, spam e ameça de vírus. Eles também podem ser usados em exercícios de revisão para ajudar as organizações a descobrirem vulnerabilidades em seu código e políticas. As formas de instalação dessas defesas são basicamente por softwares como: Snort ou Suricata que são códigos para essa funções de IPS/IDS.

2.3 Softwares de Segurança Cibernética

O aumento da dependência de sistemas digitais e a expansão do uso da internet têm ampliado exponencialmente a exposição das organizações a riscos cibernéticos, à medida que a tecnologia e sua integração na sociedade avançam, expondo ambientes corporativos e governamentais a possíveis ataques. Nesse cenário, os softwares de segurança cibernética desempenham papel essencial na proteção de dados, na

manutenção da integridade dos sistemas, na prevenção de ataques e na mitigação de riscos.

Os sistemas de segurança cibernética utilizam diferentes ferramentas, cada uma projetada para proteger uma camada específica dos ambientes de TI. Entre as principais categorias, destacam-se:

- *Antivírus*

Responsável por identificar, bloquear e remover softwares maliciosos, como vírus, trojans, worms e ransomware. Ele monitora continuamente o dispositivo para detectar comportamentos suspeitos e impedir a execução de códigos maliciosos.

- *Detecção e Resposta a Ameaças (EDR/XDR)*

Ferramentas que monitoram atividades anormais nos endpoints e utilizam inteligência artificial para identificar ataques sofisticados. Além de detectar, também automatizam respostas, como isolar máquinas comprometidas e bloquear processos maliciosos.

- *Firewall*

Atua como uma barreira entre a rede interna e a rede externa, controlando o tráfego que entra e sai com base em regras de segurança. Firewalls modernos (NGFW) inspecionam pacotes em profundidade, bloqueiam ataques e permitem segmentação de rede.

- *Detecção e Prevenção de Intrusões (IDS/IPS)*

O IDS monitora o tráfego de rede em busca de atividades suspeitas ou padrões de ataque. O IPS, além de detectar, também bloqueia automaticamente as tentativas de invasão. São essenciais para identificar ataques antes que causem danos.

- *Gestão de Identidade e Acesso (IAM)*

Controla quem pode acessar sistemas, aplicativos e recursos, garantindo autenticação adequada (como MFA) e permissões conforme o princípio de mínimo privilégio. Reduz riscos relacionados ao uso indevido de credenciais.

- *Gerenciamento de Vulnerabilidades*

Ferramentas que fazem varreduras em sistemas e redes para identificar falhas de segurança, configurações incorretas e

software desatualizado. Elas ajudam as organizações a corrigir vulnerabilidades antes que sejam exploradas por atacantes.

- *Pentest e Análise de Segurança*

Conjunto de ferramentas usadas para simular ataques reais e avaliar a resistência dos sistemas. Esses testes permitem identificar brechas, validar controles e reforçar a postura de segurança da organização.

- *Criptografia*

Tecnologia que protege a confidencialidade e integridade dos dados por meio da codificação de informações. É usada para proteger comunicações (como HTTPS), armazenar dados sensíveis e garantir autenticidade por meio de certificados digitais.

- *Backup e Recuperação*

Ferramentas responsáveis por copiar, armazenar e restaurar dados e sistemas em caso de falhas, ataques, perda acidental ou ransomware. Garantem a continuidade das operações e minimizam impactos de incidentes.

A seguir, são apresentadas as principais ferramentas de segurança cibernética, destacando softwares amplamente

utilizados no mercado, suas funções e a forma como contribuem para a proteção de ambientes corporativos. Serão abordadas soluções para proteção de endpoints, detecção e resposta a ameaças, firewalls, detecção e prevenção de intrusões, gestão de identidade, análise de vulnerabilidades, testes de segurança, criptografia e backup, bem como as melhores práticas para implementar software seguro de maneira eficiente e integrada:

- *Microsoft Defender (Antivírus / Defender)*

Microsoft Defender uma plataforma completa de segurança para terminais que combina proteção antivírus de próxima geração com capacidades avançadas de detecção e resposta (EDR). Por meio de um agente leve instalado nos dispositivos, ele monitora continuamente o comportamento de processos, arquivos e conexões, utilizando heurística e aprendizado de máquina para identificar ameaças novas e desconhecidas. Grande parte da inteligência de detecção vem da nuvem: eventos e telemetria são enviados para os servidores da Microsoft, onde são correlacionados com dados de inteligência de ameaças para identificar padrões perigosos. Quando uma ameaça é identificada, ele pode responder

automaticamente, por exemplo isolando o dispositivo ou interrompendo processos maliciosos. Além disso, o Defender oferece investigação automatizada e correção de incidentes, bem como gestão de vulnerabilidades ajudando a reduzir a superfície de ataque e reforçar a postura de segurança de maneira proativa(MICROSOFT, 2025).

- *CrowdStrike Falcon (Detecção e Resposta a Ameaças / EDR/XDR)*

É uma solução de detecção de ameaça e resposta endpoint (EDR) totalmente nativa em nuvem, baseada em um agente leve que coleta telemetria extremamente rica dos dispositivos (como processos, chamadas de biblioteca, conexões de rede). Esse agente envia os dados para a nuvem da CrowdStrike, onde modelos de machine learning e regras baseadas no comportamento (IOAs Indicators of Attack) analisam os eventos para detectar atividades suspeitas, incluindo movimentos laterais, exploração de memória e uso de credenciais. Além disso, o Falcon integra inteligência de adversários (“adversary intelligence”), o que permite contextualizar alertas com informações sobre TTPs (táticas, técnicas e procedimentos) de atacantes. Quando uma ameaça

é confirmada, a plataforma pode responder por meio de Real Time Response (RTR), permitindo execução remota de comandos, isolamento de máquinas e remoção de artefatos maliciosos. Também suporta automação de workflows (SOAR), facilitando a orquestração de respostas em larga escala. (CROWDSTRIKE, 2025).

- *Fortinet FortiGate (Firewall/NGFW)*

O FortiGate da Fortinet é um firewall de próxima geração (NGFW) que aplica inspeção profunda de pacotes (DPI – Deep Packet Inspection) para analisar não apenas os cabeçalhos de tráfego, mas também o conteúdo dos pacotes em busca de ameaças sofisticadas, como malware, botnets, canais de comando e controle (C2) e ataques de dia zero. Por meio do serviço FortiGuard, o FortiGate usa inteligência global de ameaças para manter suas regras de IPS (Intrusion Prevention System) sempre atualizadas, detectando e bloqueando ataques em tempo real. Além disso, o firewall pode aplicar políticas de controle de aplicações, filtragem de URLs/DNS, VPN e segmentação de rede, agregando múltiplas camadas de defesa para garantir visibilidade, desempenho e segurança centralizada na rede. (FORTINET, 2025).

- *Snort (Detecção e Prevenção de Intrusões – IDS/IPS)*

O Snort é uma solução open source de detecção e prevenção de intrusões baseada na captura de pacotes (libpcap), capaz de monitorar o tráfego de rede em tempo real e compará-lo com um conjunto de regras definidas pelo usuário. Cada regra do Snort estabelece condições (como IP de origem/destino, porta, conteúdo do pacote) para identificar tráfego malicioso ou suspeito. Dependendo da configuração, o Snort pode operar em três modos: sniffer (somente lê pacotes), logger (registra pacotes no disco) ou IPS (intercepta e bloqueia tráfego conforme as regras). Quando executado como IPS, ele pode descartar ou rejeitar pacotes perigosos, prevenindo ataques como injeção, varreduras ou comportamentos maliciosos. A flexibilidade de suas regras torna o Snort altamente adaptável a diferentes cenários de rede e ameaça. (SNORT, 2025).

- *Azure Active Directory (Gestão de Identidade e Acesso)*

O Azure Active Directory (parte da estrutura Microsoft Entra) é uma plataforma de identidade na nuvem que gerencia autenticação e autorização para usuários e dispositivos. Ele

permite definir políticas granulares de acesso com base em sinais diversos — como usuário, localização, dispositivo, risco — por meio do recurso Conditional Access. Por exemplo, pode-se exigir autenticação multifator (MFA) quando alguém tenta acessar recursos corporativos de fora da rede, enquanto acessos internos mais seguros podem ter requisitos mais brandos. Essa abordagem suporta a filosofia de Zero Trust, na qual não se assume confiança apenas porque alguém está autenticado. As políticas são declarativas ("se-então") e podem ser testadas em modo de relatório antes de serem habilitadas para minimizar impacto nos usuários. (MICROSOFT, 2025).

- *Nessus (Tenable) – Gerenciamento de Vulnerabilidades*

O Nessus é um scanner de vulnerabilidades altamente reconhecido que avalia a segurança dos ativos de uma organização ao identificar falhas de software, configurações incorretas e problemas de compliance. Utilizando um extenso banco de plugins atualizado com frequência, o Nessus realiza varreduras autenticadas (com credenciais) ou não, dependendo do acesso disponível, para obter uma visão completa da superfície de ataque. Ele realiza fases como

descoberta de hosts (ping, varredura de portas) e depois aplica testes de vulnerabilidade, gerando um relatório detalhado com os riscos encontrados, priorizados por métricas como CVSS e VPR (Vulnerability Priority Rating). A ferramenta permite auditoria de conformidade (ex: CIS), exportação de relatórios customizados e comparação entre varreduras para monitorar evolução das vulnerabilidades ao longo do tempo. (NESSUS, 2025).

- *Kali Linux (Pentest e Análise de Segurança)*

O Kali Linux é uma distribuição especializada em segurança ofensiva, projetada para testes de intrusão, análise de vulnerabilidade e auditoria de sistemas. Baseado em Debian, ele vem pré-configurado com centenas de ferramentas para diferentes fases de um pentest: desde a coleta de informações (reconhecimento), passando pela varredura e exploração de vulnerabilidades, até a pós-exploração e geração de relatórios. Ferramentas populares incluídas no Kali são o Metasploit, Nmap, Burp Suite, Wireshark, entre outras. O Kali permite que profissionais simulem ataques reais de forma controlada, identificando brechas que poderiam ser exploradas por

cibercriminosos, para que sejam corrigidas antes que se tornem um risco real. (KALI LINUX, 2025).

- *OpenSSL (Criptografia)*

O OpenSSL é uma biblioteca criptográfica de código aberto amplamente usada para proteger comunicações e dados. Ele oferece suporte a protocolos como TLS/SSL, além de permitir a geração, manipulação e armazenamento de chaves assimétricas (RSA, ECC) e simétricas (AES, etc.). Com o OpenSSL, aplicações podem estabelecer canais seguros via HTTPS, garantir a integridade dos dados transmitidos e assinar certificados digitais para autenticação. Também é usado para criptografar dados em repouso, gerenciar certificados e realizar operações de hash. Sua flexibilidade e robustez fazem dele um componente central em servidores, clientes web, e muitas outras soluções seguras. (OPENSSL, 2025).

- *Veeam Backup & Replication (Backup e Recuperação)*

O Veeam Backup & Replication é uma solução de proteção de dados projetada para ambientes virtuais, físicos e em nuvem. Ele realiza backups baseados em imagem, especialmente eficiente para máquinas virtuais, usando

snapshots para capturar todo o estado do sistema. Esses dados são armazenados em repositórios seguros com compressão e deduplicação para otimizar espaço. Além disso, o Veeam permite testar a recuperação automaticamente: ele pode restaurar VMs diretamente a partir dos backups para verificar a integridade e disponibilidade dos dados. Em incidentes, como ataques de ransomware ou falhas críticas, a ferramenta possibilita restaurar rapidamente dados e sistemas, minimizando a perda e garantindo a continuidade das operações. (VEEAM, 2025)

- *Práticas na implementação de software seguro*

A implementação de software seguro envolve um conjunto de práticas que visam reduzir vulnerabilidades, proteger dados sensíveis e garantir que sistemas e aplicações possam resistir a ataques. Esse processo deve ser contínuo, integrando segurança ao ciclo de desenvolvimento, implantação e operação. Entre as melhores práticas, destaca-se o uso de ferramentas destinadas a fortalecer diferentes camadas da arquitetura de TI, desde o endpoint até a infraestrutura de rede e identidade.

Uma das medidas essenciais é garantir a proteção dos dispositivos que executam o software, prevenindo que malwares comprometam o ambiente. Soluções como o Microsoft Defender Antivírus e o CrowdStrike Falcon exercem papel fundamental nesse contexto: elas monitoram processos, analisam comportamento e detectam ameaças que poderiam explorar falhas no software, como execução de código malicioso ou escalonamento de privilégios. Uma empresa pode configurar o Defender para bloquear automaticamente a execução de scripts desconhecidos utilizados por atacantes para explorar aplicações web vulneráveis. Já o CrowdStrike Falcon pode isolar uma máquina que apresente comportamentos suspeitos, impedindo a propagação lateral e resguardando o ambiente onde o software está hospedado.

Outra prática indispensável é o controle de acesso adequado, softwares seguros devem implementar autenticação forte, autorização granular e princípios como “mínimo privilégio”. O Azure Active Directory ajuda a aplicar essas práticas por meio de políticas de Acesso Condicional e autenticação multifator (MFA). Por exemplo, uma aplicação corporativa pode exigir que usuários só acessem determinadas

funcionalidades se estiverem utilizando dispositivos confiáveis ou autenticados via MFA, reduzindo o risco de comprometimento por credenciais roubadas.

A proteção da rede também é central para o desenvolvimento seguro. Firewalls de próxima geração, como o Fortinet FortiGate, oferecem inspeção profunda de pacotes que permite identificar e bloquear tráfego malicioso que tenta explorar vulnerabilidades em softwares expostos à internet. Em um cenário real, uma API utilizada por um aplicativo pode ser protegida por regras que filtram portas não autorizadas, bloqueiam tentativas de força bruta e detectam ataques como SQL Injection. Complementando essa defesa, sistemas IDS/IPS como o Snort identificam padrões de ataque direcionados a aplicações, gerando alertas para equipe de segurança ajustar configurações e corrigir falhas exploráveis.

A implementação de software seguro também requer análise proativa de vulnerabilidades. Ferramentas como o Nessus (Tenable) ajudam nesse processo ao identificar falhas no ambiente onde o software está sendo executado, incluindo bibliotecas desatualizadas, portas abertas indevidas e configurações incorretas. Um exemplo prático é uma varredura

periódica com o Nessus para identificar serviços que não deveriam estar habilitados no servidor de aplicação, prevenindo superfícies de ataque desnecessárias.

Além disso, o desenvolvimento seguro inclui testes contínuos, tanto de forma automatizada quanto ofensiva. O uso do Kali Linux é essencial para auditorias de segurança, pois permite realizar testes de intrusão antes da publicação do software. Um analista pode usar ferramentas como Metasploit ou Burp Suite dentro do Kali para descobrir vulnerabilidades de autenticação ou falhas de validação de entrada antes que atacantes as explorem. Esse ciclo de teste e correção fortalece a robustez do aplicativo antes de sua entrada em produção.

Outro pilar fundamental é a proteção de dados por criptografia. O OpenSSL é amplamente utilizado para implementar HTTPS e proteger comunicações entre cliente e servidor, garantindo confidencialidade e integridade das informações. No contexto prático, um software corporativo pode gerar certificados TLS usando OpenSSL, configurando canais seguros entre APIs e bancos de dados, impedindo que informações sensíveis transitem em texto puro.

Por fim, mesmo com práticas rígidas, incidentes podem acontecer por isso, a última camada de segurança é a continuidade de negócios e recuperação de dados, ferramentas como Veeam Backup & Replication garantem que, caso um ataque comprometa aplicações, a organização tenha condições de restaurar rapidamente sistemas e dados, reduzindo impactos operacionais. A configuração de backups diários e réplicas em nuvem, podem restaurar uma versão íntegra da aplicação após um ataque de ransomware.

Em resumo, a implementação de software seguro depende da combinação de práticas de desenvolvimento, configurações adequadas e uso estratégico de ferramentas especializadas. Cada solução desempenha um papel complementar e, juntas, contribuem para construir um ambiente resiliente, confiável e preparado para enfrentar os desafios crescentes da cibersegurança moderna.

3. Metodologia

Diante do crescimento da tecnologia, os desafios da segurança de sistemas se sofisticaram e um sistema de gestão de bibliotecas que coleta, armazena e processa dados pessoais e sensíveis se tornou um alvo potencial para ataques cibernéticos, a metodologia aqui proposta visa delinear, de forma sistemática, os critérios de seleção, as práticas de desenvolvimento e os controles necessários para conceber um sistema de gestão de bibliotecas alinhado aos princípios de qualidade de software e às exigências normativas da Lei Geral de Proteção de Dados Pessoais no Brasil, e além disso incorpora-se o uso de técnicas de uso da inteligência artificial tanto para apoiar testes de segurança e o desenvolvimento de algoritmos, de acordo com abordagens apresentadas por Sommerville em Software Engineering em 2016.

A alta variabilidade de ferramentas de tecnologia decorre, segundo Ian Sommerville e Barry Boehm, da complexidade e da velocidade de evolução dos sistemas de software e como isso gera a necessidade de diferentes ferramentas e abordagens. Portando o Quadro 2 apresenta a proposta metodológica para desenvolvimento do sistema

Nexus Library e de implementação de algoritmo criptográfico, considerando os princípios de qualidade de software, exigências normativas brasileiras e o uso de inteligência artificial para aprimorar os fatores de segurança.

Quadro 2 - Frameworks de Desenvolvimento Seguro

Referência	Metodologia	Descrição
Sommerville (2016)	Software Development Life Cycle (SDLC) & Clean Code	Utilização de um ciclo de desenvolvimento estruturado (como o proposto, baseado em SDLC) e aplicação de Técnicas de Programação Segura para proteção de dados de entrada e armazenamento (e.g., uso de ORM Django para prevenir SQL Injection).
NIST (2025) e ISO/IEC 27002	Gestão de Segurança da Informação e Padrões Criptográficos	Utilização de padrões criptográficos robustos (como Bcrypt para hashing de senhas) e implementação de controles como Controle de Acesso Baseado em Perfis (RBAC) (usuário, bibliotecário, administrador) e Autenticação Multifator (MFA) para restringir o acesso a recursos e dados .
LGPD (Brasil, 2028)	Análise de Impacto à Proteção de Dados (DPIA)	Metodologia de avaliação proativa para identificar e mitigar os riscos à privacidade resultantes da implementação do novo componente criptográfico, garantindo conformidade legal e adesão aos princípios da legislação aplicável.
OWASP Foundation	OWASP Top 10 / OWASP ZAP	Framework de Testes de Segurança e validação de código. O uso do OWASP ZAP é focado na realização de análise de vulnerabilidades e testes de penetração automatizados para

		identificar falhas comuns (e.g., XSS, CSRF, SQL Injection) no ambiente Web (Django/Apache) .
Aprendizado de Máquina (IA)	Detecção de Anomalias (UEBA)	Investigação e proposta de uma técnica de IA para monitorar logs de atividade (LogAtividade) e padrões de acesso, com o objetivo de identificar comportamentos anômalos que possam indicar tentativas de ataque ou comprometimento, auxiliando na otimização de parâmetros de defesa.

3.1 Planejamento do Projeto

O gerenciamento do projeto seguiu as diretrizes do ciclo de vida de desenvolvimento de sistemas (SDLC - Systems Development Life Cycle), adaptado para incluir etapas rigorosas de Security by Design. Para garantir a organização e o cumprimento dos prazos, o cronograma foi estruturado em fases distintas, abrangendo desde a análise de requisitos de segurança até a validação final dos algoritmos criptográficos e módulos de Inteligência Artificial.

3.1.1 Cronograma de Atividades

O projeto foi delineado no período de 6 meses de execução. O Quadro 3 detalha as principais etapas, prazos e entregáveis.

Quadro 3 – Cronograma de Execução do Projeto

Fase	Atividade Principal	Período	Entregável
1. Planejamento	Levantamento de requisitos de segurança e definição do escopo criptográfico.	Mês 0 ao mês 1	Documento de Requisitos de Segurança.
2. Pesquisa	Estudo sobre algoritmos (AES/SHA-256), LGPD e técnicas de IA aplicadas à segurança.	Mês 1 ao mês 3	Referencial Teórico e escolha das tecnologias e relatório do projeto.
3. Desenvolvimento	Codificação do algoritmo criptográfico e integração com o banco de dados MySQL.	Mês 3 ao mês 4	Módulo de Criptografia funcional.
4. Testes	Execução de testes de penetração, análise de vulnerabilidades e validação de input.	Mês 4 ao mês 5	Relatório de Testes de Segurança.
5. Documentação	Análise de Impacto, escrita do artigo e	Mês 5 ao mês 6	Artigo final e Relatório final

	preparação da apresentação.		

3.1.2 Divisão de Tarefas

A equipe, composta por cinco integrantes, foi organizada de forma colaborativa, atribuindo-se responsabilidades de acordo com as competências técnicas individuais e os objetivos específicos do projeto (Qualidade de Software, Legislação e Inteligência Artificial). A distribuição das funções ocorreu da seguinte maneira:

- *Alex Santos da Silva e Bruno Neemias Alves Mota*

Responsáveis pelo desenvolvimento do backend e implementação do algoritmo criptográfico. Focaram nas técnicas de programação segura e integração com o banco de dados.

- *Lucas Bezerra da Silva*

Encarregado da execução dos testes de segurança (Testes de Penetração e Análise de Vulnerabilidades) e validação da robustez do código.

- *Marcela de Castro Santos*

Responsável pela análise de conformidade legal (LGPD), elaboração da Análise de Impacto à Proteção de Dados (DPIA) e documentação técnica.

- *Melquisedec Souza Assunção:*

Responsável pela pesquisa e prototipagem do componente de Inteligência Artificial voltado para a detecção de anomalias e otimização dos parâmetros de segurança.

3.2 Ferramentas e Tecnologias

A seleção das ferramentas e tecnologias priorizou soluções de código aberto (open source), amplamente reconhecidas pela comunidade de segurança da informação e compatíveis com a arquitetura do sistema Nexus Library. A seguir, são listadas as ferramentas utilizadas e suas respectivas justificativas técnicas.

3.2.1. Linguagem de Programação: PHP 8.3

Escolhida devido ao seu amplo suporte nativo para bibliotecas de criptografia robustas (como OpenSSL e Sodium) e funções de hashing seguras (password_hash, password_verify). Além disso, o PHP possui integração facilitada com bancos de dados e frameworks modernos que permitem implementar práticas de programação segura e auditoria de código. Sua sintaxe acessível e a grande comunidade tornam a manutenção e evolução do sistema mais ágil, garantindo confiabilidade em aplicações web de qualquer porte.

3.2.2. Sistema Gerenciador de Banco de Dados (SGBD): MySQL 8.0

Justifica-se a escolha, pois para garantir a persistência dos dados do sistema Nexus Library. O MySQL oferece recursos nativos de controle de acesso e suporte a conexões criptografadas (SSL/TLS), essenciais para os objetivos de confidencialidade do projeto.

3.2.3. Bibliotecas de Criptografia

O bcrypt (Provos, 1999) é um método de criptografia do tipo hash para senhas baseado no Blowfish. Foi criado por Niels Provos e David Mazières e apresentado na conferência da USENIX em 1999. Foi especificamente escolhido para o algoritmo de hashing de senhas (conversão de senhas), pois ele permite trabalho ajustável, adaptativo, conhecido como “sal e custo”, o que oferece proteção eficaz contra ataques de força bruta e rainbow tables.

3.2.4. Ambiente de Desenvolvimento Integrado (IDE): Visual Studio Code (VS Code)

Adotado pela flexibilidade e suporte a extensões de análise estática de código (SAST), que auxiliam na identificação precoce de vulnerabilidades de segurança durante a escrita do código.

3.2.5. Controle de Versão: Git e GitHub

Essencial para o gerenciamento do código-fonte, permitindo o trabalho colaborativo, rastreabilidade das alterações e versionamento seguro do projeto.

3.2.6. Ferramentas de Teste de Segurança: OWASP ZAP (Zed Attack Proxy)

O Open Web Application Security Project (OWASP) é uma iniciativa colaborativa internacional que atua como uma comunidade online aberta. Seu objetivo é criar e disseminar metodologias, ferramentas, tecnologias e orientações essenciais para o desenvolvimento de aplicações web seguras. O projeto, composto por indivíduos e empresas, busca padronizar as abordagens de segurança no desenvolvimento web, compartilhando ativamente o conhecimento na área.

Foi escolhido, pois é uma das ferramentas de segurança mais populares e ativamente mantidas pelo OWASP, por centenas de voluntários internacionais, é o OWASP ZAP (ZAP). Esta ferramenta gratuita é fundamental para testar a segurança de aplicações web, sendo útil tanto para testadores de

penetração experientes quanto para iniciantes. É capaz de analisar aplicações web e identificar uma ampla gama de problemas de segurança, incluindo:

- Injeção de SQL
- Cross Site Scripting (XSS)
- Desserialização insegura
- Autenticação corrompida
- Controle de acesso quebrado
- Exposição de dados sensíveis
- Configuração de segurança incorreta
- Componentes com vulnerabilidades conhecidas
- Cabeçalhos de segurança ausentes.

3.2.7. Análise de Dados: Pandas e Scikit-learn

O pandas é uma biblioteca de software de código aberto construída em Python para análise e manipulação de dados. A biblioteca pandas fornece estruturas de dados projetadas especificamente para lidar com conjuntos de dados tabulares com uma API Python.

Já o Scikit-Learn é uma biblioteca de Machine Learning muito conhecida e utilizada do Python, dentre as diversas existentes, elaborada em código aberto e desenvolvida para suportar e possibilitar o treino de diversas técnicas de estatística e Machine Learning, para aprendizagem supervisionada e não supervisionada.

Portanto, focando em desempenho, compatibilidade e mitigar falhas de segurança na biblioteca a escolha se deu, pois é amplamente utilizada e revisada.

3.2.8. Ferramenta de Inteligência Artificial

O Microsoft Defender na Nuvem envia telemetria para a nuvem onde os dados são correlacionados com inteligência de ameaças para identificar padrões perigosos e o CrowdStrike Falcon usa ML e regras baseadas em comportamento (IOAs - Indicators of Attack) para detectar atividades suspeitas, como exploração de memória e movimentos laterais. Escolhidos, pois oferecem capacidade preditiva e são ferramentas de fácil implementação.

4. Desenvolvimento

4.1 Implementação do Algoritmo Criptográfico

A implementação do algoritmo criptográfico e das técnicas de programação segura foi focada em três pilares essenciais para a segurança da aplicação web: o armazenamento seguro de credenciais, a proteção de dados sensíveis em logs de auditoria e a prevenção de vulnerabilidades de injeção.

4.1.1 Armazenamento Seguro de Credenciais

O armazenamento de senhas de usuários foi implementado utilizando a função password_hash() do PHP, conforme a Figura 1, que emprega o algoritmo Bcrypt com um salt gerado aleatoriamente e um fator de custo ajustável. Este método garante que as senhas não possam ser recuperadas em texto simples, mesmo em caso de acesso não autorizado ao banco de dados.

Figura 1 - Evidência de Código (PHP – cadastrar_usuario.php)

O código demonstra duas práticas essenciais de segurança no registro de usuários:

- *Hash seguro de senhas (Bcrypt):*

A senha fornecida pelo usuário não é armazenada em texto puro, mas convertida em um hash irreversível com password_hash(), garantindo confidencialidade mesmo em caso de vazamento da base.

- *Consulta segura com Prepared Statement:*

A inserção no banco é feita com prepare() e bind_param(), evitando SQL Injection e assegurando integridade dos dados.

autenticar.php

```php
<?php
session_start();
include 'db.php';
include 'funcoes.php';

// Verifica se os dados foram enviados via POST
if ($_SERVER['REQUEST_METHOD'] !== 'POST' || !isset($_POST['email']) || !isset($_POST['senha'])) {
  echo "<script>alert('Acesso inválido.'); window.location.href='login.php';</script>";
  exit;
}

// Captura os dados do formulário
$email = $_POST['email'];
$senha = $_POST['senha'];

// Consulta o usuário no banco
$sql = "SELECT * FROM usuario WHERE email = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("s", $email);
$stmt->execute();
$resultado = $stmt->get_result();
$usuario = $resultado->fetch_assoc();

// Verifica se o usuário existe e compara a senha
if ($usuario && password_verify($_POST['senha'], $usuario['senha'])) {
  $_SESSION['usuario'] = $usuario;
  registrar_log_atividades($conn, 'Login', 'Usuário autenticado com sucesso', $usuario['idUsuario']);
  header("Location: dashboard.php");
  exit;
} else {
  $mensagem = $usuario ? 'Tentativa de login com senha incorreta' : 'Tentativa de login com email não cadastrado';
  registrar_log_atividades($conn, 'Login', $mensagem, null);
  echo "<script>alert('Usuário ou senha inválidos!'); window.location.href='login.php';</script>";
}
```

Figura 2 - Evidência de Código (PHP – autenticar.php)

O código acima demonstra a verificação de login no sistema. A senha digitada pelo usuário é comparada com o hash armazenado no banco utilizando password_verify(), garantindo segurança e irreversibilidade. Além disso, a consulta SQL é protegida contra injeção com prepare() e bind_param(), e todas as tentativas de login são registradas na tabela logatividade com criptografia AES.

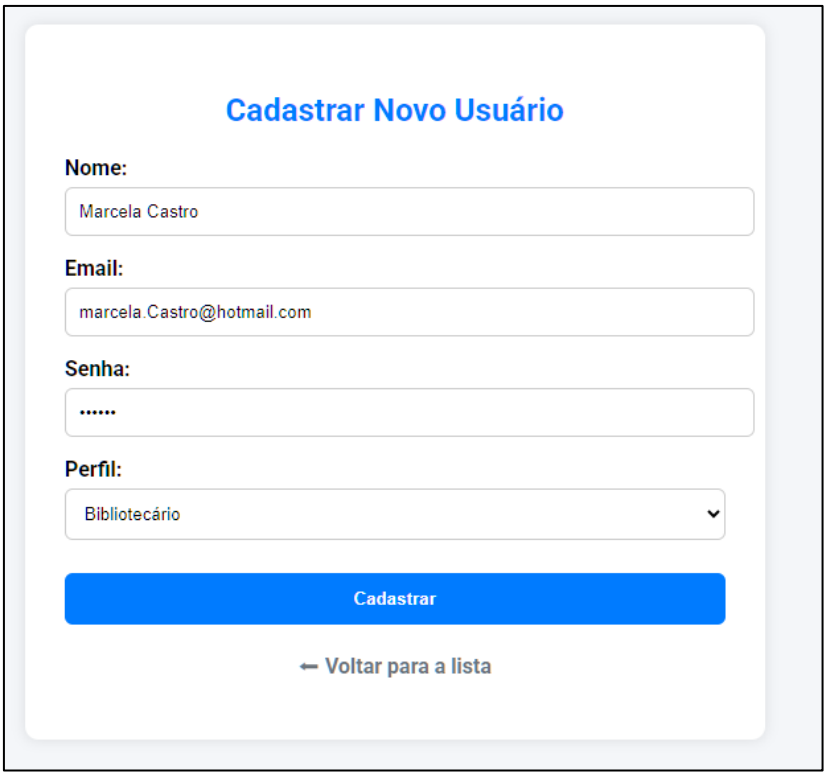


Figura 3 - Evidência de Sistema (cadastrar_usuario.php)

A evidência apresentada demonstra o preenchimento do formulário de Cadastro de Novo Usuário. Tal operação é restrita exclusivamente ao perfil de administrador do sistema, garantindo que apenas usuários autorizados possam realizar a inclusão de novos registros.

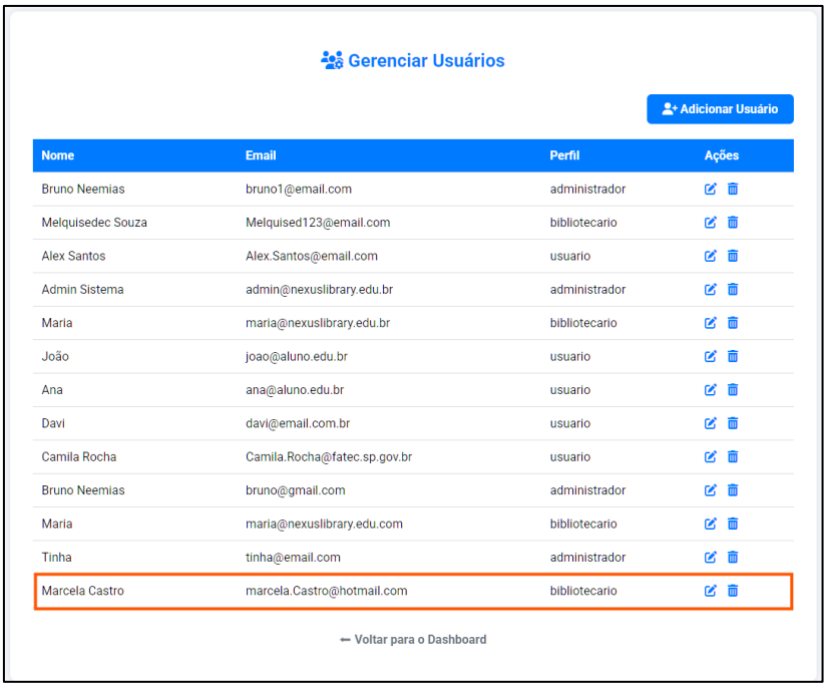


Figura 4 - Evidência de Sistema (cadastrar_usuario.php)

A evidência apresentada demonstra a interface de Gerenciamento de Usuários do sistema. Nela é possível visualizar a lista de usuários cadastrados, incluindo informações como Nome, E-mail e Perfil (administrador ou bibliotecário). A tela também disponibiliza ações administrativas, como edição e exclusão de registros, além da opção de adicionar novos usuários. O acesso a essa funcionalidade é restrito a perfis com

privilégios administrativos, garantindo o controle e a segurança na gestão de contas.

14	Maria	maria@nexuslibrary.edu.com	mMT9gA2g8IQTyADpGrv9skl0OUV1K1kvaHgwbDEyU0VYZnd3U0...
15	Tinha	tinha@email.com	$2y$10$q0FKnSOx4hYLnFJ9/srKkeyux9KVPJNfE9YwjJ0mr.H...
16	Marcela Castro	marcela.Castro@hotmail.com	$2y$10$OFKKiQTVOzo9qJYGWzUc8urBTuSSkA87KBH7/9eYv8m...

Figura 5 - Evidência do sistema da tabela de usuários

A evidência mostra que, após o cadastro da usuária Marcela Castro, o campo correspondente à senha não aparece em texto legível, mas sim como uma sequência longa de caracteres aleatórios. Isso confirma que o sistema está aplicando corretamente o processo de criptografia/hashing na senha, garantindo que ela não seja armazenada em formato puro no banco de dados.

4.1.2 Criptografia de Logs de Auditoria (AES-256)

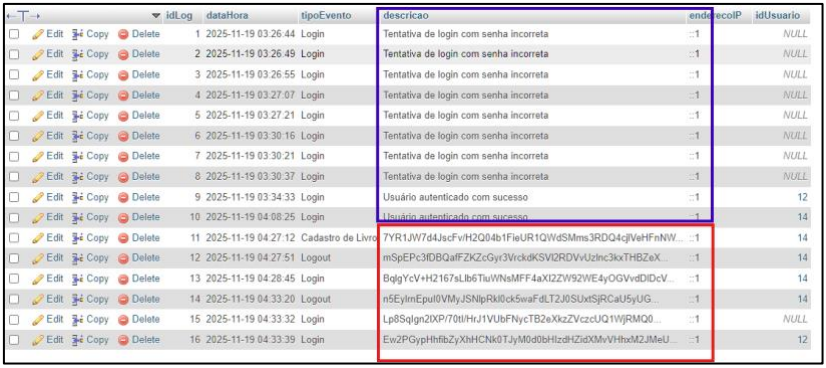
Para garantir a confidencialidade dos dados de auditoria, como tentativas de acesso e operações críticas, o sistema registra logs criptografados utilizando o algoritmo AES-256 no modo GCM (Galois/Counter Mode), por meio da biblioteca OpenSSL do PHP. O uso do GCM garante tanto a

confidencialidade quanto a autenticidade dos dados (via Authentication Tag). Conforme Evidência abaixo.

funcoes.php

```php
<?php
define('AES_KEY', 'sua-chave-secreta-32bytes');

function criptografarAES($texto) {
  $iv = openssl_random_pseudo_bytes(16);
  $cifra = openssl_encrypt($texto, 'AES-256-CBC', AES_KEY, 0, $iv);
  return base64_encode($iv . $cifra);
}

function descriptografarAES($textoCriptografado) {
  $dados = base64_decode($textoCriptografado);
  $iv = substr($dados, 0, 16);
  $cifra = substr($dados, 16);
  return openssl_decrypt($cifra, 'AES-256-CBC', AES_KEY, 0, $iv);
}

function registrar_log_atividades($conn, $tipoEvento, $descricao, $idUsuario = null) {
  $enderecoIP = $_SERVER['REMOTE_ADDR'];
  $descricaoCriptografada = criptografarAES($descricao);

  if ($idUsuario === null) {
    $stmt = $conn->prepare("
      INSERT INTO logatividade (tipoEvento, descricao, enderecoIP, dataHora)
      VALUES (?, ?, ?, NOW())
    ");
    $stmt->bind_param("sss", $tipoEvento, $descricaoCriptografada, $enderecoIP);
  } else {
    $stmt = $conn->prepare("
      INSERT INTO logatividade (tipoEvento, descricao, enderecoIP, idUsuario, dataHora)
      VALUES (?, ?, ?, ?, NOW())
    ");
    $stmt->bind_param("sssi", $tipoEvento, $descricaoCriptografada, $enderecoIP, $idUsuario);
  }

  $stmt->execute();
  $stmt->close();
}

?>
```

Figura 6 - Evidência de Código (PHP – funcoes.php)



	idLog	dataHora	tipoEvento	descricao	endereco IP	idUsuario
Edit Copy Delete	1	2025-11-19 03:26:44	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	2	2025-11-19 03:26:49	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	3	2025-11-19 03:26:55	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	4	2025-11-19 03:27:07	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	5	2025-11-19 03:27:21	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	6	2025-11-19 03:30:16	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	7	2025-11-19 03:30:21	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	8	2025-11-19 03:30:37	Login	Tentativa de login com senha incorreta	::1	NULL
Edit Copy Delete	9	2025-11-19 03:34:33	Login	Usuário autenticado com sucesso	::1	12
Edit Copy Delete	10	2025-11-19 04:08:25	Login	Usuário autenticado com sucesso	::1	14
Edit Copy Delete	11	2025-11-19 04:27:12	Cadastro de Livro	7YR1JW7d4JscFv/H2Q04b1FleUR1QWdSMms3RDQ4cjlVeHFnNW...	::1	14
Edit Copy Delete	12	2025-11-19 04:27:51	Logout	mSpEPc3fDBQafFZKZcGyr3VrckdKSVl2RDVvUzlnc3kxTHBZeX...	::1	14
Edit Copy Delete	13	2025-11-19 04:28:45	Login	BqlgYcV+H2167sLlb6TluWNsMFF4aXl2ZW92WE4yOGVvdDlDcV...	::1	14
Edit Copy Delete	14	2025-11-19 04:33:20	Logout	n5EyIrnEpuI0VMyJSNlpRkl0ck5waFdLT2J0SUxtSjRCaU5yUG...	::1	14
Edit Copy Delete	15	2025-11-19 04:33:32	Login	Lp8SqIgn2lXP/70tl/HrJ1VUbFNycTB2eXkzZVczcUQ1WjRMQ0...	::1	NULL
Edit Copy Delete	16	2025-11-19 04:33:39	Login	Ew2PGypHhfibZyXhHCNk0TJyM0d0bHIzdHZidXMvVHhxM2JMeU...	::1	12

Figura 7 - Evidência de Sistema (tabela logatividade)

A evidência apresentada demonstra o comportamento da tabela de auditoria antes e depois da implementação da criptografia AES.

Marcação Azul → registros gravados em texto simples, permitindo a leitura direta da descrição dos eventos.

Marcação Vermelha → registros gravados após a implementação da criptografia AES-256, onde a descrição aparece como uma sequência de caracteres cifrados, garantindo a confidencialidade das informações de auditoria.

4.1.3 Prevenção de SQL Injection (Prepared Statements)

Todas as interações com o banco de dados foram refatoradas para utilizar Prepared Statements (declarações preparadas) com a extensão MySQLi. Esta técnica garante que o código SQL seja compilado separadamente dos dados de entrada, eliminando a possibilidade de um atacante alterar a lógica da consulta por meio de caracteres especiais.

```php
$stmt = $conn->prepare("SELECT * FROM Usuario WHERE idUsuario = ?");
$stmt->bind_param("i", $id);
$stmt->execute();
$result = $stmt->get_result();
$usuario = $result->fetch_assoc();

if ($_SERVER["REQUEST_METHOD"] === "POST") {
  $nome = $_POST['nome'];
  $email = $_POST['email'];
  $perfil = $_POST['perfil'];

  $stmt = $conn->prepare("UPDATE Usuario SET nome=?, email=?, perfil=? WHERE idUsuario=?");
  $stmt->bind_param("sssi", $nome, $email, $perfil, $id);

  if ($stmt->execute()) {
    header("Location: usuarios.php");
    registrar_log_atividades($conn, "Edição de Perfil", "Usuário alterou seus dados", $idUsuario);
    exit;
  } else {
    $erro = "Erro ao atualizar: " . $stmt->error;
  }
}
?>
```

Figura 8 - Evidência de Código (PHP - Consulta Segura)

O trecho acima evidencia a implementação de consultas seguras no sistema Nexus Library. A utilização de prepare() e bind_param() garante que os dados recebidos dos formulários sejam tratados como parâmetros, impedindo que comandos

maliciosos sejam interpretados como parte da consulta SQL. Essa prática elimina vulnerabilidades de SQL Injection e reforça a integridade do sistema

5. Discussão

5.1 Avaliação dos Resultados

A implementação das medidas de segurança no sistema Nexus Library atingiu com sucesso os objetivos propostos. A adoção de password_hash() para senhas, AES-256 para logs e Prepared Statements para consultas ao banco de dados elevou significativamente a postura de segurança da aplicação, alinhando-a com as melhores práticas de desenvolvimento seguro e os requisitos da Lei Geral de Proteção de Dados (LGPD). A confidencialidade das credenciais dos usuários e dos registros de auditoria foi garantida, e a integridade do banco de dados foi protegida contra ataques de SQL Injection, mitigando riscos críticos de segurança da informação.

5.2 Desafios Encontrados

O principal desafio técnico enfrentado durante o projeto foi o gerenciamento seguro da chave criptográfica utilizada no algoritmo AES-256. Armazenar a chave de forma segura, sem que ela estivesse diretamente no código-fonte ou em um arquivo de configuração de fácil acesso, exigiu a

implementação de uma estratégia de armazenamento em variáveis de ambiente no servidor, isoladas do diretório da aplicação. Outro desafio foi a refatoração completa do código legado para a adoção de Prepared Statements, um processo que demandou tempo e testes rigorosos para garantir que nenhuma funcionalidade existente fosse comprometida.

5.3 Melhorias Futuras

Embora o projeto tenha alcançado seus objetivos principais, diversas melhorias podem ser implementadas em trabalhos futuros para fortalecer ainda mais a segurança do sistema.

- *Implementação de Criptografia Assimétrica (RSA):*

Utilizar o algoritmo RSA para a troca segura da chave simétrica (AES) entre diferentes componentes do sistema ou para a comunicação segura com APIs externas.

- *Assinatura Digital de Documentos:*

Implementar assinaturas digitais para garantir a autenticidade e a integridade de documentos importantes

gerados pelo sistema, como relatórios de acervo ou comprovantes de empréstimo.

- *Criptografia de Dados em Relatórios:*

Aplicar criptografia aos dados sensíveis contidos em relatórios exportados (e.g., PDFs, planilhas), garantindo que, mesmo fora do sistema, a informação permaneça protegida.

6. Referências bibliográficas

BRASIL. Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). Diário Oficial da União, Brasília, DF, 15 ago. 2018.

CROWDSTRIKE. CrowdStrike Falcon: Endpoint Protection Platform. Disponível em: https://www.crowdstrike.com. Acesso em: 15 nov. 2025.

FORENSE.IO. O que é: Acesso Não Autorizado - Glossário Forense. Disponível em: https://forense.io/glossario/o-que-e-acesso-nao-autorizado-glossario-forense/. Acesso em: 11 out. 2025.

FORTINET. FortiGate Next-Generation Firewall. Disponível em: https://www.fortinet.com/products/next-generation-firewall. Acesso em: 15 nov. 2025.

FORTINET. O que é segurança cibernética? Diferentes tipos de...Disponível em: https://www.fortinet.com/br/resources/cyberglossary/what-is-cybersecurity. Acesso em: 08 out. 2025.

FORTINET. O que é um sistema de intrusão? Disponível em: https://www.fortinet.com/br/resources/cyberglossary/intrusion-detection-system. Acesso em: 15 nov. 2025.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO) ; INTERNATIONAL ELECTROTECHNICAL COMMISSION (IEC). ISO/IEC 27002:2022: Information security, cybersecurity and privacy protection — Information security controls. Genebra, 2022.

KALI LINUX. Kali Linux Documentation and Tools. Disponível em: https://www.kali.org. Acesso em: 15 nov. 2025.

MICROSOFT. Microsoft Defender for Endpoint – Documentatio n. Disponível em: https://learn.microsoft.com/microsoft-365/security/defender-endpoint. Acesso em: 15 nov. 2025.

MICROSOFT. Microsoft Entra ID (Azure Active Directory). Disponível em: https://learn.microsoft.com/azure/active-directory. Acesso em: 15 nov. 2025.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST). Gaithersburg, MD: NIST, 2020.

NESSUS/TENABLE. Nessus Vulnerability Scanner – Documentat ion. Disponível em: https://www.tenable.com/products/nessu s. Acesso em: 15 nov. 2025.

NIST. NIST Cryptographic Standards and Guidelines. Disponível em: https://csrc.nist.gov/projects/cryptographic-standards-and-guidelines. Acesso em: 13 out. 2025.

OPENSSL SOFTWARE FOUNDATION. OpenSSL Cryptography and SSL/TLS Toolkit. Disponível em: https://www.openssl.org. Acesso em: 15 nov. 2025.

OWASP FOUNDATION. OWASP Top 10 – The Most Critical Web Application Security Risks. 2021.

PREY PROJECT. Criptografia simétrica e assimétrica: guia comp leto para criptografia. Disponível em: https://preyproject.com /es/blog/tipos-de-cifrado-simetrico-o-asimetrico-rsa-o-aes. Acesso em: 11 nov. 2025.

PROVOS, N.; MAZIÈRES, D. A future-adaptable password scheme. In: USENIX Annual Technical Conference, Freenix Track, 1999, Monterey, CA. Proceedings... Berkeley, CA, USA: USENIX Association, 1999. p. 119-124.

SNORT. Snort – Network Intrusion Detection & Prevention System. Disponível em: https://www.snort.org. Acesso em: 15 nov. 2025.

SOMMERVILLE, I. Software engineering. 10. ed. Boston: Pearson, 2016.

VEEAM. Veeam Backup & Replication – Documentation. Disponível em: https://www.veeam.com. Acesso em: 15 nov. 2025.