Berghem

Smart Information Security

DESENVOLVIMENTO SEGURO

Guia de Consulta Rápida

+55 11 2391-5745

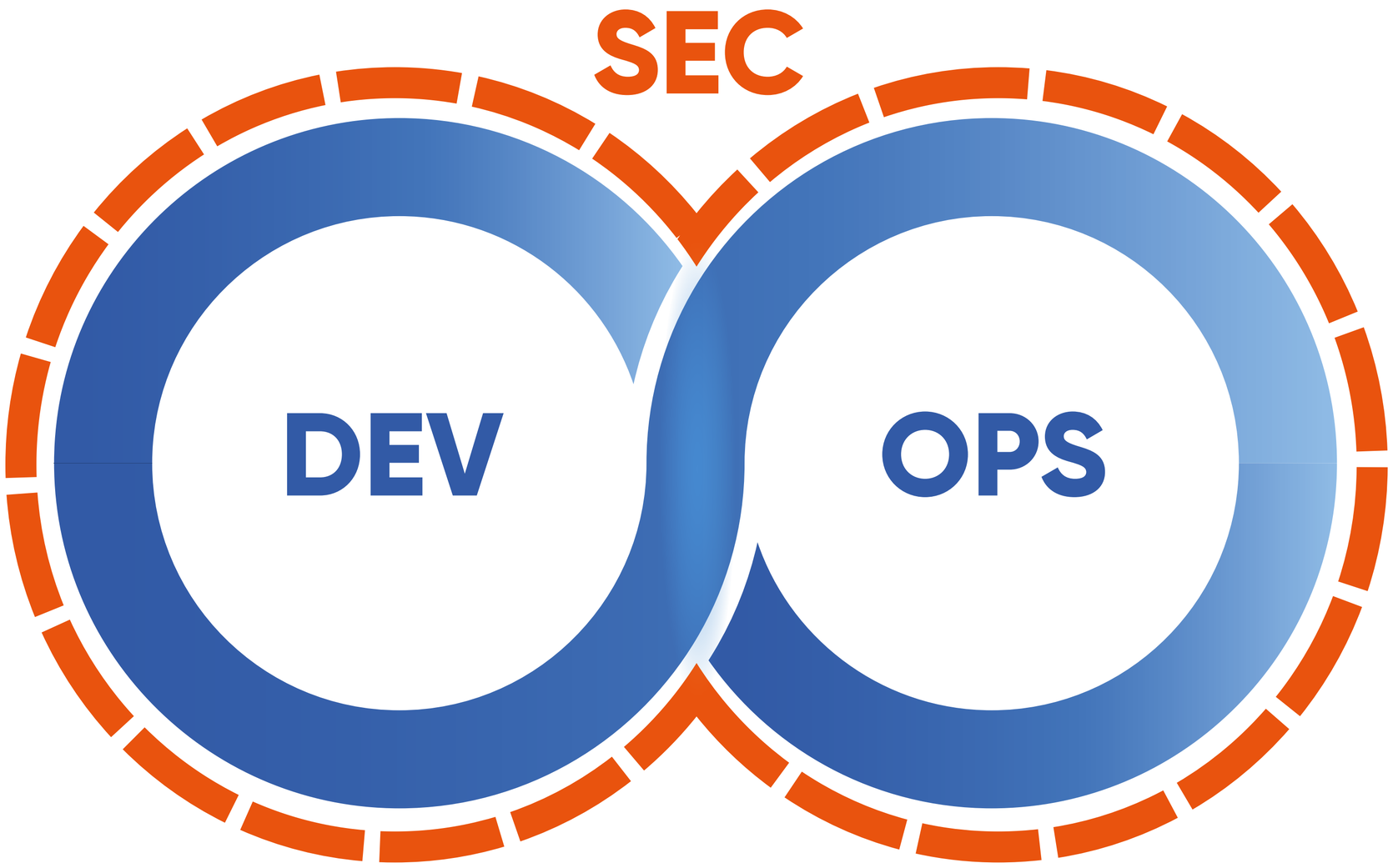
berghem.com.br

comercial@berghem.com.br

DESENVOLVIMENTO SEGURO

Guia de Consulta Rápida

Berghem
Smart Information Security



INTRODUÇÃO

Desenvolvimento Seguro (S-SDLC e DevSecOps) é um conjunto de práticas e processos com o objetivo de agregar segurança ao longo do processo de desenvolvimento de software, desde o design e arquitetura da aplicação, até a entrada em produção e sustentação.

Hoje em dia, é impensável e inviável esperar meses para o lançamento de uma **nova *feature*.** Consequentemente, problemas que antes não eram tão alarmantes vêm ganhando cada vez mais destaque, dentro e fora das organizações.

A **segurança cibernética**, um desses problemas, é amplificado pela alta velocidade nas transformações do ambiente de negócios e de desenvolvimento de software.

Temos consciência do enorme desafio que é **equilibrar** os assuntos de segurança cibernética, inovação e novos negócios. Mas... >

A **Berghem - *Smart Information Security*** é uma **consultoria especializada em Desenvolvimento Seguro** e pode auxiliar sua empresa nesta jornada, desde a **avaliação** do cenário atual, até a **implementação** de processos de desenvolvimento seguro, bem como a **operação/execução** de determinadas etapas do processo.

Nossa equipe de consultores está à disposição para auxiliá-lo a elaborar as estratégias mais adequadas quanto à segurança, privacidade e conformidade de dados, aplicações, redes, infraestruturas e meios de pagamento.

...não é mais possível escolher entre segurança e negócios.

É preciso, em vez disso, incrementar a eficácia da **segurança cibernética** — tornando-a parte do time de **desenvolvimento de software** (por onde são lançados novos produtos) e, ao longo da jornada, estruturar processos, implantar novas tecnologias de modo a ter uma postura de **segurança cibernética assertiva.** ▪

DESENVOLVIMENTO SEGURO E LEGISLAÇÃO

A **segurança de aplicações** e o próprio processo de **desenvolvimento seguro** são partes fundamentais do processo de adequação a leis como a **LGPD** e o regulamento de proteção de dados da União Europeia (**GDPR** na sigla em inglês). Como a própria lei brasileira traz em seu art. 46, § 2º, as "medidas de que trata o caput deste artigo deverão ser observadas desde a fase de concepção do produto ou do serviço até a sua execução". O artigo, portanto, estabelece que as medidas descritas no caput, ou seja, as **"medidas de segurança, técnicas e administrativas** aptas a proteger os dados pessoais de acessos não autorizados e de situações acidentais ou ilícitas de destruição, perda, alteração, comunicação ou qualquer forma de tratamento inadequado ou ilícito" devem ser observadas no desenvolvimento do software ou da funcionalidade.

A SEGURANÇA CIBERNÉTICA É UMA JORNADA. DÊ O SEU PRIMEIRO PASSO AGORA, COM ESTE E-BOOK.

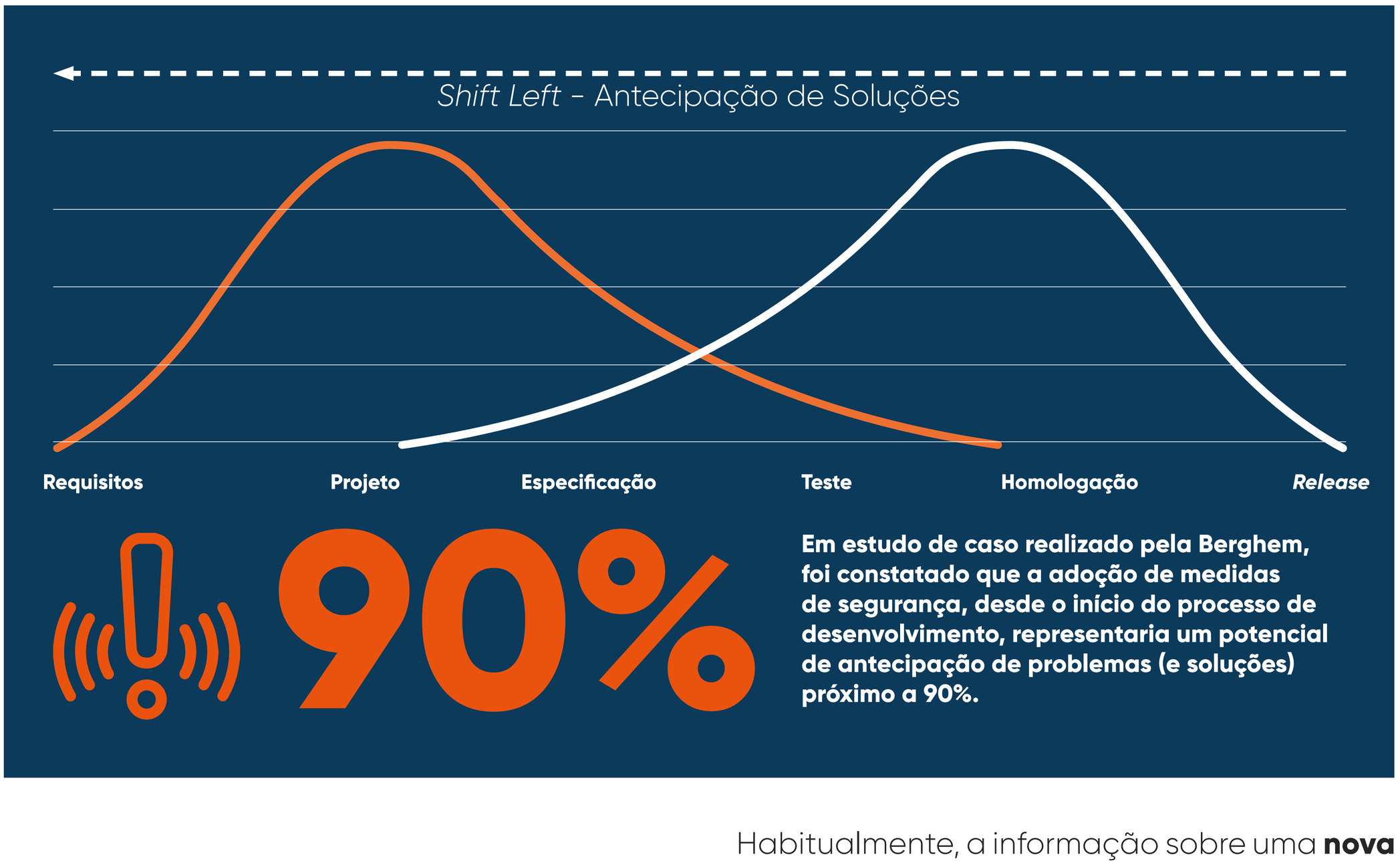
SEGURANÇA INTEGRADA

Times de desenvolvimento e segurança devem caminhar juntos; desenvolvedores, perceber o valor das orientações e recomendações da equipe de segurança. E o time de segurança deve compreender os objetivos de negócios e as necessidades do desenvolvimento.

As abordagens que possuem, atualmente, mais repercussão são o **DevSecOps** para a metodologia de desenvolvimento ágil e o **S-SDLC** (*Secure Software Development Lifecycle*) para a de desenvolvimento em modelo cascata.

Ambos os cenários partem do princípio de trazer a segurança cibernética cada vez mais para o começo do processo de desenvolvimento.

Quanto mais cedo as equipes se preocupam com os **requisitos de segurança**, mais problemas podem ser evitados, poupando horas e horas de trabalho e dinheiro com as tão indesejadas **"refações".**



90%

Em estudo de caso realizado pela Berghem, foi constatado que a adoção de medidas de segurança, desde o início do processo de desenvolvimento, representaria um potencial de antecipação de problemas (e soluções) próximo a 90%.

Habitualmente, a informação sobre uma **nova funcionalidade** chega às equipes de segurança apenas quando já estão prontas para o release, ao passo que o time de desenvolvimento está focado numa próxima entrega — e de repente precisa **corrigir problemas** de segurança da entrega anterior, gerando desentendimentos, bloqueios e riscos (soa familiar?) >

COMO EQUILIBRAR ESSA BALANÇA

É preciso ter uma **interface de comunicação e colaboração eficiente que busque unificar as diferentes necessidades (negócio, prazos, desenvolvimento e segurança)** em todo o processo de desenvolvimento, visando garantir a entrega de um sistema seguro para os usuários.

Abordagens em linha com a figura do ***Security Champions***, em que um desenvolvedor é eleito para ser a referência de segurança cibernética dentro do time, têm se mostrado eficazes nesse contexto.

Além disso, as empresas devem investir cada vez mais em **treinamentos sobre segurança cibernética** para os desenvolvedores. Inúmeras vulnerabilidades podem ser evitadas apenas **aplicando arquiteturas de referências, boas práticas para codificação e configuração de ambientes e servidores.**

MODELAGEM DE AMEAÇAS

Uma boa estratégia é procurar identificar as principais ameaças e vulnerabilidades relacionadas à aplicação, bem como estabelecer os controles de segurança elementares, antes de iniciar o desenvolvimento.

A **modelagem de ameaças**, por exemplo, é uma ótima ferramenta para introduzir o assunto de segurança cibernética às equipes de tecnologia.

Quando os desenvolvedores entendem que suas **escolhas** (desde linguagem de programação, frameworks, IDEs, padrões de projeto, controle e protocolos de segurança e outros) podem trazer **riscos de segurança cibernética** para o projeto, eles passam a tomar **decisões mais conscientes.**

A **modelagem de ameaças** é uma aliada para a construção desse entendimento.

Antes mesmo de adotar modelos mais famosos e robustos disponíveis no mercado (como **STRIDE, Kill Chains, MITRE ATT&CK ou Attack Trees**), os times de desenvolvimento podem colher alguns dos benefícios da modelagem de ameaças ao se perguntar:

- **Quais são as vulnerabilidades cibernéticas mais conhecidas das linguagens de programação utilizadas? E dos frameworks?**
- **Qual é a versão de frameworks e bibliotecas que estamos usando? Todos os desenvolvedores estão na mesma versão? A versão está atualizada? Existe um processo de atualização definido?**

Além disso, podem ser realizadas outras **investigações** em conjunto com as equipes de segurança cibernética, infraestrutura e gerência de projetos.

- **Se eu fosse um cibercriminoso, qual estratégia eu usaria para ter acesso ao sistema?**
- **Quais são as informações confidenciais dele? Como são armazenadas e protegidas?**
- **Qual funcionalidade apresenta maior risco de segurança cibernética?**
- **Existe integração com algum outro sistema que apresenta risco? Qual risco? Esse fornecedor também está preocupado com segurança cibernética?**

As **respostas** para essas perguntas podem:

- **Revelar, em número e grau, os riscos associados;**
- **Permitir que ameaças sejam antecipadas;**
- **E, dessa forma, evitar vulnerabilidades na aplicação;**

GESTÃO DE VULNERABILIDADES

Toda vulnerabilidade possui um ciclo de vida. Ela é identificada, classificada, remediada e documentada como lição aprendida.

A **gestão de vulnerabilidades**, além de ser considerada uma boa prática de **segurança cibernética**, também é relevante na adequação das regulamentações que envolvem privacidade e proteção de dados pessoais, a exemplo da **LGPD e do GDPR**, e outras que envolvem dados financeiros, como **PCI-DSS** e **BACEN 4.893**.

Você deve dispor de um **processo** para receber essas informações, e, logo em seguida, alguém ou algum sistema deve **categorizá-las**.

Além das vulnerabilidades em **sistemas próprios**, é preciso também identificar vulnerabilidades atreladas à sua **infraestrutura de rede** e ativos computacionais.

A classificação é uma atividade chave da gestão de vulnerabilidades, pois é parte do processo de gestão de risco e permite a sua priorização.

Portanto, é necessário ter um bom método ou sistema de **gestão de vulnerabilidades** para gerenciar o processo de tratamento conforme o nível e **criticidade e relativo risco**. >

ALGUMAS PERGUNTAS QUE PODEM AJUDAR NESSE PROCESSO SÃO:

- **Qual é a criticidade da vulnerabilidade? (dica: vale a pena usar calculadoras de criticidade para responder à pergunta, como o *Common Vulnerability Scoring System Calculator - CVSS*);**
- **Essa vulnerabilidade foi localizada em uma funcionalidade de uso frequente pelos usuários ou uma funcionalidade de pouco uso?**
- **Esse código é fácil ou complexo de resolver?**
- **Qual o impacto e o dano se um cibercriminoso explorar essa vulnerabilidade?**
- **Meu time atual é capaz de corrigir essa vulnerabilidade ou vou precisar de ajuda externa e especializada?**

Após a classificação, é importante possuir um processo para **correção e documentação dessas vulnerabilidades**. Tal processo permite gerenciar o fluxo de correção de vulnerabilidades conforme o nível de risco, controlando o estado de tratamento de cada apontamento, mantendo sua capacidade de desenvolver novos produtos. ■

APRENDIZADO COM OS ERROS

Os erros do passado podem se tornar um valioso ativo da empresa, ao gerar uma exclusiva base de conhecimento e integrá-lo à modelagem de ameaças e as atividades de capacitação das equipes.

Errar rápido, corrigir rápido e melhorar rápido. Em sintonia com as metodologias ágeis, os erros são considerados ativos valiosos de **lições aprendidas**. Conheça a seguir 3 maneiras de **aprender com os erros** do passado e **melhorar continuamente**.

ENGENHARIA REVERSA + 5 POR QUÊS

Se algo não aconteceu como esperado, que tal realizar uma **engenharia reversa** para entender o ocorrido?

Para isso, pode ser utilizada a técnica dos **5 por quês**. Essa técnica consiste basicamente em perguntar "por quê?" 5 vezes ao problema original, até chegar à **causa raiz**, a partir de uma engenharia reversa direcionada. Ao fazer isso, você e sua equipe previnem que erros derivados da mesma causa se repitam.

SPRINT REVIEW

Reuniões de retrospectivas são ótimos momentos para refletir sobre os acontecimentos recentes e encontrar soluções para problemas. Que tal adicionar um pouco de **segurança** cibernética? Convide alguém da equipe de segurança para participar, mesmo que nas primeiras reuniões a pessoa se mantenha como ouvinte.

BLAMELESS POST-MORTEM

Esse documento, também conhecido como **relatório pós-incidente**, direciona as equipes na busca de solução para os acontecimentos sem culpar as pessoas pelos problemas, mas, sim, encontrar as **falhas** em processos e sistemas para correção.

É essencial entender o que deu errado e como corrigir o erro no no futuro próximo. Por isso, é preferível manter essas informações organizadas e catalogadas em uma **base de conhecimento** ou **wiki**.

Essas informações são riquíssimas para melhoria dos processos e evitar a repetição de erros já cometidos.■

SISTEMAS LEGADOS

É essencial ter cuidado redobrado com a segurança dos sistemas legados – especialmente quanto a integrações, comunicação, regras de negócios e controles de segurança.

Não podemos ignorar que os **sistemas legados** ainda colaboram imensamente com o funcionamento de muitas aplicações que utilizamos hoje.

O equívoco em segurança cibernética não é sobre utilizarmos os sistemas legados, mas esquecermo-nos que eles existem.

Além disso, às vezes adotam **tecnologias antigas** e não propriamente adequadas e seguras para as aplicações modernas. Assim, o que era seguro antes pode não ser tão seguro após a integração em um ambiente moderno.

Múltiplas equipes de desenvolvimento possuem **pouca familiaridade** com esses sistemas — e, na verdade, para desempenhar seu trabalho, precisam apenas fazer **chamadas e integração com os sistemas legados.**

Pelo conceito de **"não mexer no que está funcionando"**, os sistemas legados são considerados eficientes e consolidados. E isso pode **agravar riscos desnecessários** em segurança cibernética. >

INVENTÁRIO DE ATIVOS

O primeiro passo para reverter essa situação é iniciar um **inventário de ativos.** Nesse inventário, inclua os **sistemas legados**, sua forma de funcionamento, datas de manutenção, riscos cibernéticos conhecidos e com quais outros sistemas ele está integrado e, **sobretudo, qual forma de segregação e integração com os novos sistemas está presente.**

Após esse levantamento, convoque as suas equipes de tecnologia e apresente as informações para responder as seguintes perguntas.

Quais desses sistemas apresentam maior risco cibernético para a empresa? Por quê?

É possível manter esses sistemas ou uma melhor estratégia seria realizar uma migração completa?

Como faremos para realizar a manutenção dos sistemas legados?

Com quais APIs e de qual forma esses sistemas estão integrados?

Existe algum projeto para integração desse sistema com outros?

Qualquer descuido poderá ocasionar uma vulnerabilidade a ser explorada.

A revisão dos sistemas legados também é importante para efeito de conformidade com a LGPD e o GDPR. Essas regulamentações consideram sistemas legados e aplicativos modernos sob o mesmo nível de conformidade.

SECURITY & PRIVACY BY DESIGN *& BY DEFAULT*

Implemente medidas técnicas e organizacionais nas fases iniciais do desenvolvimento de aplicações, reduza custos operacionais e esteja em conformidade com a LGPD e o GDPR.

Security & Privacy by* Design *& by Default — A segurança e privacidade por design e por padrão é uma abordagem que exige pensar em **segurança cibernética e privacidade desde o início de um projeto**, como funcionalidade intrínseca, padrão, recorrente do sistema.

A **LGPD** traz a noção de que medidas de segurança devem ser implementadas desde a concepção do produto ao *deployment* em produção. Deve-se, portanto, ser implementado um processo de desenvolvimento seguro.

No mesmo caminho, surgiu o ***Privacy by* Design,** abordagem que prevê a privacidade durante todo o processo de desenvolvimento da aplicação. >

CONHEÇA OS 10 PRINCÍPIOS DO *SECURITY BY* DESIGN.

1. **Estabelecer medidas** para minimizar a superfície de ataque;
2. **Definir padrões** de desenvolvimento seguro;
3. **Aplicar o princípio** do menor privilégio;
4. **Assimilar o princípio** de defesa em profundidade;
5. **Implementar a manipulação** segura de erros (falhar com segurança);
6. **Entender o conceito** de *Zero Trust*;
7. **Utilizar perfis de acesso** por função (ou RBAC – *Role Based Access Control*);
8. **Evitar** a segurança por obscuridade;
9. **Manter a simplicidade** e eficiência dos processos de segurança;
10. **Aplicar a segurança** cibernética na manutenção dos sistemas.

O *PRIVACY BY DESIGN* POSSUI 7 PRINCÍPIOS FUNDAMENTAIS EM SUA CONCEPÇÃO:

1. **Proativo,** não reativo; preventivo, não corretivo — de modo a evitar incidentes de violação à privacidade;

2. **Privacidade** como configuração padrão. As configurações padrão de determinado sistema devem ser ajustadas desde o início para preservar a privacidade do usuário;

3. **Privacidade** incorporada ao design, incluindo a arquitetura e modelos de negócio;

4. **Funcionalidade total.** Soma positiva, não soma-zero – ou seja, com benefícios e resultados para todos;

5. **Segurança de ponta a ponta.** Proteção completa incorporada ao ciclo de vida da informação;

6. **Visibilidade e transparência.** Manter a aplicação aberta (de fato) a consultas de dados pessoais e práticas de privacidade pelos usuários.

7. **Respeito pela privacidade** do usuário, cujos interesses devem estar em primeiro plano.

Os **desenvolvedores** precisam ter esses conceitos bem consolidados, havendo a necessidade de treinamento com foco em **privacidade e proteção de dados**, frente às exigências de leis como a LGPD e GDPR, e convidando um **consultor especializado** no assunto sempre que necessário. ■

TESTES E RETESTES

Com essas recomendações, as aplicações tendem a chegar mais seguras à fase de homologação e produção. Ainda assim, avaliações de vulnerabilidades e testes de segurança são indispensáveis, bem como as respectivas correções e retestes.

Se você realmente quer saber como está a sua **segurança cibernética**, os especialistas recomendam que você faça um **teste de penetração.**

Teste de penetração, também conhecido como **pentest** ou **hacking ético**, é uma forma avançada e ofensiva de **teste de segurança** projetada para fornecer uma análise técnica profunda da vulnerabilidade de um sistema ou ambiente.

Essa técnica vai além da avaliação básica de riscos e testes automatizados, sendo fundamental contar com a **experiência** de profissionais de segurança qualificados e que seguem um **processo de teste** rigoroso para simulação de ataques reais.

O teste de penetração é **extremamente valioso** para as empresas. Isso porque procuram simular e repetir formas de ataques adotadas por fraudadores e criminosos buscando encontrar fragilidade e portas vulneráveis que **permitam comprometer a segurança da aplicação.** >

FORMAS DE TESTAR

Black Box

Teste cego

Gray Box

Com conhecimento parcial e/ou login

White Box

Com conhecimento completo e acesso ao código-fonte

Tal **visão externa é essencial** para oferecer uma **análise independente** e **experiente** em relação a testes feitos por scanners e/ou pela equipe interna da empresa.

Um bom trabalho de ***hacking* ético** deve contemplar **testes para vulnerabilidades conhecidas** e o uso de **métodos** populares de *hacking*, mas também executar **fraudes especializadas**, de acordo com o negócio do cliente (ex.: meios de pagamento, varejo, saúde etc.) — inclusive por meio da verificação dos **controles** de segurança e **antifraude** implementados.

O resultado é um **relatório com as vulnerabilidades identificadas e mitigações recomendadas**. Inclusive, esse tipo de relatório costuma ser solicitado em procedimentos de **auditorias externas** de fornecedores e **conformidade com regulamentações.**

Também é importante ressaltar a **periodicidade** para realização desses testes. Idealmente, eles devem estar atrelados à **agenda de novos releases** e funcionalidades da aplicação, bem como a **modificações significativas ou a requisitos normativos.** ■

FRAMEWORKS DE SEGURANÇA

Indicamos alguns ***frameworks*** de desenvolvimento e **cibersegurança** que compilam **boas práticas** e anos de **pesquisas** de instituições dedicadas.

É necessário, no entanto, que você leve em consideração também os aspectos do seu negócio e mercado de atuação para desenvolver um processo assertivo para sua realidade. ■

SAMM (*Software Assurance Maturity Model*)

Programa de Gerenciamento de Patches e Vulnerabilidades do NIST (*National Institute of Standards and Technology*)

Guia de Gerenciamento de Vulnerabilidades do OWASP (*Open Web Application Security Project*)

CHEGAMOS AO FIM...

...ou melhor, ao começo da sua nova jornada no desenvolvimento seguro!

Com este material, esperamos **conscientizar** equipes de negócios e de desenvolvimento sobre a **importância do desenvolvimento seguro**, a partir dos principais pontos de atenção.

Forte abraço do nosso time!

Berghem

Smart Information Security

+55 11 2391-5745

berghem.com.br

comercial@berghem.com.br