FACULDADE DE TECNOLOGIA SENAC GOIÁS
Segurança da Informação

Aldo Brito
Jordan Gladys
Liniker Lettierre
Rony Carneiro

POLÍTICA PARA O DESENVOLVIMENTO DE SOFTWARE SEGURO

Olegário Correa da Silva Neto

GOIÂNIA,
2016

Aldo Brito
Jordan Gladys
Liniker Lettierre
Rony Carneiro

POLÍTICA PARA O DESENVOLVIMENTO DE SOFTWARE SEGURO

Relatório apresentado como requisito parcial para obtenção de aprovação na disciplina de Segura em Programação, no Curso de Segurança da Informação, na Faculdade de Tecnologia Senac Goiás.

Olegário Correa da Silva Neto

GOIÂNIA,
2016

SUMÁRIO

1 INTRODUÇÃO

Objetivando a construção de uma política de segurança da informação em software, o seguinte trabalho tem como foco tópicos que envolvem assegurar o ambiente de desenvolvimento do projeto, assim como definição de direitos de acesso físico e lógico do software, seu desenvolvimento baseado em requisitos de segurança e bons costumes a serem seguidos, resultando na segurança do desenvolvimento do código crítico e em andamento, livre de erros, interrupções e com uma boa qualidade do serviço.

2 DESENVOLVIMENTO

Deve-se levar em consideração uma política de segurança em desenvolvimento de softwares, que se faz relevante pois objetiva evitar a ocorrência de desastres que atinjam o ambiente e garantir uma boa sustentação, para que o desenvolvimento e o ambiente onde ocorre a produção sejam bem-sucedido e seguro logicamente e fisicamente.

Para assegurar o ambiente de desenvolvimento se leva em consideração certos itens:

- Os acessos devem ser discutidos pela diretoria para inibir qualquer tipo de contato de não autorizados com o ambiente do projeto;
- Participantes do projeto devem assegurar o não compartilhamento de informações com terceiros, a não ser se permitido pela diretoria;
- Se faz necessário a boa alocação dos equipamentos usados para armazenar os dados, com clima e manutenção em boas condições e em dias;
- Todo tipo de sinistros ocorridos no momento deverá ser avaliado por uma equipe com integrantes escolhidos pelo diretor do projeto;
- Deverá haver uma equipe de analistas para verificar se o projeto vem cumprindo com as regras da programação segura;

Em relação a segurança do software em si, as principais vantagens para as organizações ao adotar um modelo de gestão de segurança de software são bastante relevantes, podendo citar:

- **Melhoria da segurança**: Identifica e elimina as vulnerabilidades (no código fonte) antes que elas se tornem uma ameaça no ambiente de produção e de missão crítica;
- **Redução de custos**: Identificar e eliminar as vulnerabilidades do software antes dele ir para o ambiente de produção proporcionam redução significativa nos custos de desenvolvimento, correções e respostas a incidentes;
- **Demonstra a conformidade**: Garante que o seu desenvolvedor atende aos padrões de codificação segura, validando os critérios de aceitação do software antes do seu desenvolvimento.

Além disso, algumas boas práticas de segurança que auxiliam no desenvolvimento de softwares, muitas delas com usos de ferramentas se fazem de importante implementação:

→ Gerenciamento de código fonte

- Deve-se fazer utilização do GitHub, um serviço de Web Hosting compartilhado para projetos que utilizam o controle de versão Git (sistema de controle de versão distribuído também um sistema de gerência de source code, enfatizando velocidade.

→Realização de testes

- Uso de teste desde pequenos trechos de código por vez até práticas que buscam uma avaliação geral do software.

→ Gerenciamento de correção de bugs

- O uso de uma ferramenta de Bug Tracking permite manter registro das falhas encontradas no sistema e facilita a comunicação entre os envolvidos na identificação e correção de bugs.

→ Utilização de processo de integração contínua

- Essa prática tem como objetivo garantir a qualidade no software desenvolvido, automatizando verificações no processo de trabalho da ferramenta, garantindo assim, que seja possível gerar uma nova versão com o mínimo de bugs.

→ Documentação do software e da arquitetura que o suporta

- Uma clara documentação da arquitetura e código fonte ajuda a aumentar a qualidade do software desenvolvido.

→ Utilização de padrões de código seguro e checklists

- Definição de padrões de códigos seguros e boas práticas de acordo com a linguagem adotada e ambiente definido pela organização se fazem necessárias. Faça Checklists para verificar as principais ações durante o desenvolvimento e revisão de segurança do software.

É necessário compreender que a definição dos cenários negativos, cuja realização é indesejada, resulta no requisito de segurança. Para obtermos bons resultados, utilizamos certos requisitos de segurança em diversos campos:

→ Âmbito de Confidencialidade

- Senha e outros campos de entrada sensíveis precisam de máscaras.
- Senhas não devem ser armazenadas as claras nos sistemas backend, e se forem devem passar por processo de hash com uma função pelo menos equivalente a uma forte função hash, como por exemplo o SHA-256.
- Transport Layer Security (TLS) como Secure Socket Layer (SSL) deve ser colocado em prática para proteger contra ameaças internas de Man in the Middle (MITM).
- O uso de protocolos inseguros como o FTP para transmissão de credenciais de contas em texto claro para fora da organização deve ser proibido.

- Arquivos de log não devem armazenar qualquer informação sensível como definido pelo negócio, de modo que seja compreensível por seres humanos.

→ Âmbito de Integridade

- Os formulários de entrada e query strings precisam ser validadas frente a um conjunto de entradas aceitáveis, antes do software os aceitar para processá-los.
- O software a ser publicado deve ser disponibilizado juntamente com o checksum e a função hash usada para computar o checksum, de modo que o interessado possa validar sua precisão e completude.

→ Âmbito de Disponibilidade

- O software deve estar preparado para atender máxima capacidade inicial de 100 usuários simultâneos, aumentando com de acordo com a demanda.
- O software e seus dados devem ser replicados por todos os centros de dados para balancear carga e redundância.
- A funcionalidade de missão crítica no software deve ser restaurada a operação normal no prazo de 1 hora de descontinuidade, o da missão essencial no software no prazo de 4 horas da interrupção, e o de missão suporte no software no prazo de 24 horas.

→ Âmbito de Autenticação

- O software será implantado apenas na Intranet e o usuário autenticado deve fornecer suas credenciais de novo para acessar a aplicação, uma vez já autenticado na rede.
- Na política de autenticação se faz necessário dois/ou autenticações com múltiplos fatores para todos os softwares de processamento financeiro.

→ Âmbito de Âmbito de Autorização

- O acesso a arquivos secretos sensíveis deve ser restrito somente a usuários com níveis de permissão Secret.
- Os usuários não devem ser demandados a enviar suas credenciais sempre, uma vez que ele já tenha se autenticado.
- Todos os usuários autenticados herdarão a permissão de leitura somente que são parte do papel do usuário convidado enquanto os usuários autenticados por padrão terão permissão de leitura e escrita como usuário regular. Apenas os usuários com acesso administrativo, terão todos os direitos dos usuários regulares.

→ Âmbito de Auditoria e Logs

- As tentativas de logon deverão ser registradas junto com o timestamp e endereço de IP de origem da requisição.
- Logs de auditoria deverão ser adicionados de novos registros e nunca sobrescritos.
- Logs de auditoria deverão ser mantidos de modo seguro por um período de 3 anos.

- Cada uma das atividades do usuário deverá ser rastreada de modo único.
- As sessões devem ser suspensas quando o usuário solicita o log off ou fecha a janela do browser.
- Identificadores de sessão para identificar a sessão de usuários não devem ser passados em claro ou se adivinhar facilmente.

→ Âmbito de Erros e Gerenciamento de Exceção

- Os erros e exceções devem ser manipulados a partir de blocos try, catch e finally.
- Mensagens de erro que são mostradas ao usuário revelarão somente a informação necessária, sem mostrar detalhes internos do sistema na mensagem.
- Detalhes de exceções de segurança devem ser auditados e monitorados em certo período.
- Deverão ser encriptados os dados sensíveis do arquivo de configuração da aplicação web, como strings de conexão.
- Senhas e chaves de criptografia não devem ser registradas no código fonte do software.
- A inicialização e a liberação de variáveis globais devem ser monitoradas com cuidado.
- Eventos de inicialização e interrupção de sessão devem incluir proteções na informação de configuração, uma contramedida contra ameaças de vazamento.

3 CONCLUSÃO

Tendo em vista os aspectos observados, vê-se que para que o software a ser desenvolvido, e o ambiente de desenvolvimento estejam seguros, faz-se necessário que a alocação dos equipamentos usados no projeto seja assegurada contra sinistros que possam ocorrer e ocasionar perdas, boa manutenção e climatização.

Também deve-se discutir tanto o acesso do pessoal que será autorizado no ambiente de desenvolvimento quanto o nível de autenticidade dado a cada indivíduo. Realização de testes, gerência de código e bugs e utilização de padrões de código seguro devem ser implementadas para que o desenvolvimento do código tenha qualidade de serviço, resultando na redução considerável dos custos, maior segurança e amostra de conformidade com os critérios para que o código seja aceito no seu pré-desenvolvimento.

Para compreender cenários negativos, implementação de requisitos de segurança faz-se necessários no Âmbito de confidencialidade, disponibilidade, na autenticação, autorização, campo de erros e gerência de exceções, reduzindo chances de acessos indevidos aos dados do software, e erros que possam ocasionar interrupção da missão crítica e indisponibilidade do mesmo.

4 REFERÊNCIAS BIBLIOGRÁFICAS

http://blog.andradesoto.com.br/archives/2014/05/20/desenvolvimento-de-software-seguro-para-as-organizacoes/
http://www.sirc.unifra.br/artigos2010/7.pdf
http://micreiros.com/boas-praticas-para-desenvolvimento-de-softwares-seguros/
http://softwareseguro.blogspot.com.br/