Guia para Desenvolvimento de Software Seguro

Tribunal Regional do Trabalho da 4a Região

Versão	Data	Autor	Descrição
1.0	01/09/17	Alexandre Leite, Anderson Miranda, André Farias, Felipe Giotto, Felipe Levin, Michel Barreto, Stéfano Mór.	Versão inicial.
1.0.1	30/11/17	Stéfano Mór	Inclusão de verificação periódicas de listas de vulnerabilidades no *roadmap*.
1.0.2	24/04/18	Stéfano Mór	Unificação das seções “Determinação de Identidade e Nível de Acesso de Usuários” com “Autenticação de Usuários”.
2.0.0	19/01/23	Carlos Augusto Rivilino Lucas Pozatti Stéfano Mór	Atualização de diretivas com mudanças estruturais e passagem de foco para o desenvolvimento de aplicações *Web*.

O presente documento deve ser revisado ao menos anualmente.

Diretrizes para a revisão estão disponíveis em manual complementar.

APRESENTAÇÃO E OBJETIVOS

Em um cenário em que o Judiciário está cada vez mais digital, disponibilizando seus serviços por meio de aplicações *Web*, torna-se primordial que os sistemas e aplicações sejam desenvolvidos observando os princípios de segurança da informação, sob pena de possibilitarem a consecução de ataques cibernéticos, com impactos cada vez mais devastadores.

Este documento é o guia para desenvolvimento seguro de software no âmbito do Tribunal Regional do Trabalho da 4a Região (TRT4). Seu objetivo é servir como guia de boas práticas a serem adotadas por analistas e técnicos, desenvolvedores e instaladores de software, tornando o processo de concepção dos sistemas construídos dentro deste Tribunal mais seguro, confiável, auditável e estável. As orientações aqui contidas são direcionadas a todos os envolvidos no processo de desenvolvimento de software no âmbito do TRT4. Adotando-se as práticas seguras recomendadas neste guia, reduz-se consideravelmente os riscos de segurança relacionados às aplicações disponibilizadas na internet, tornando o ambiente do TRT4 mais seguro.

As diretrizes constantes no presente instrumento foram elaboradas originalmente por um Grupo de Trabalho para Desenvolvimento Seguro de Software (GTDevSeg), subordinado diretamente à Coordenadoria de Desenvolvimento de Sistemas (CDS) da Secretaria de Tecnologia da Informação e Comunicações (SETIC) do TRT4. A partir da versão 2.0.0 deste guia, a Coordenadoria de Segurança da Informação e Proteção de Dados (CSIPD), subordinada à SETIC, passou a trabalhar na revisão desse documento em parceria com a CDS. Consoante a esta colaboração, o foco do guia foi alterado para passar a privilegiar — de maneira não-exclusiva — os aspectos referentes ao desenvolvimento seguro de soluções e serviços em ambiente *Web*, baseando-se majoritariamente e transcrevendo diretivas do *OWASP Secure Coding Practices Quick Reference Guide*[1], com material adicional extraído do *OWASP Application Security Verification Standard 4.0*[2] e demais referências.

FORMATO

O documento é estruturado em torno de “diretrizes”, recomendações de boas práticas a serem seguidas em cada um dos tópicos listados. Seu formato é detalhado abaixo. Após, discute-se a estrutura do documento e disposição de seu conteúdo.

[1] https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf
[2] https://owasp.org/www-pdf-archive/OWASP_Application_Security_Verification_Standard_4.0-en.pdf

DIRETRIZES

Uma diretriz é escrita em forma imperativa ou imperativa negativa — "*Deve-se* [...]" ou "*Não se deve* [...]" e pode estar classificada, em função da necessidade de proteção aos dados e das obrigações imputadas ao programador ou analista, em dois níveis:

- **Mínimo**. São *deveres* a serem seguidos na construção de sistemas para que se obtenha um nível de segurança considerado indispensável.
- **Complementar**. São *medidas adicionais* pertinentes à construção de sistemas para que se aumente o nível de segurança do sistema desenvolvido, quando crítico.

  Os níveis são *cumulativos*, salvo em sobreposição de escopo.

A cada diretriz é atribuído um identificador único (ID). O ID é formado pela concatenação de uma letra indicando o nível da diretriz — "M" para para o mínimo, "C" para o complementar — e uma numeração no formato «número do seção».«número da subseção (quando houver)».«enumeração sequencial por nível da diretriz». Cada diretriz possui um ID único. No caso de diretrizes removidas, seu ID é preservado e não pode ser atribuído a outras diretrizes nas versões futuras do documento.

O conjunto de diretrizes de uma dada seção é apresentado em formato tabular, onde cada diretriz é listada com seu ID e definição. As diretrizes de nível mínimo vêm primeiro, seguidas pelas diretrizes de nível complementar. *Eg*., para uma hipotética seção 29,

ID	Diretriz
M29.1	Não se deve utilizar diretrizes sem o respectivo ID.
M29.2	Não se deve atribuir o mesmo ID a duas ou mais diretrizes distintas.
C29.1	Deve-se preservar o ID de diretrizes declaradas obsoletas.
C29.2	Não se deve aproveitar IDs de diretrizes declaradas obsoletas em novas diretrizes.

ESTRUTURA

Este guia é direcionado, sobretudo, a desenvolvedores de *software* inseridos na construção dos sistemas do TRT4. Considerando esse cenário, o guia foi projetado para facilitar a consulta expressa, mas sem omitir informações detalhadas.

O guia possui dois capítulos principais. O primeiro, "Diretrizes para Desenvolvimento de Software Seguro", contém diretrizes para o desenvolvimento de *software* cuja

implementação seja resiliente no escopo de sua segurança, com orientações que abordam práticas de programação, uso de criptografia, uso de senhas fortes, *etc*. O segundo, “Diretrizes para Desenvolvimento Seguro de Software” contém diretrizes que tornam o processo de desenvolvimento de software mais seguro e confiável, com boas práticas para versionamento, controle de acesso ao código fonte, testes de software, *etc.*

Além dos capítulos principais, um glossário está disposto ao final do documento e deve ser utilizado em caso de necessidade de detalhamento das seções iniciais.

REVISÕES

O processo de revisão deste guia, de periodicidade ao menos anual, deve ser efetuado seguindo diretrizes próprias, disponíveis em seu manual complementar para revisões.

DIRETRIZES PARA DESENVOLVIMENTO DE SOFTWARE SEGURO

Todas as seções que dizem respeito ao desenvolvimento de sistemas *Web* assumem que as operações são realizadas em ambientes municiados por uma infraestrutura segura *a priori*.

1 ARQUITETURA DE SOFTWARE

Diretrizes para auxílio na construção de um projeto arquitetural de *software* seguro, focando-se em práticas gerais de programação e no desenho de seus componentes.

ID	Diretriz
M1.1	Deve-se isolar no código da aplicação os trechos de código que contêm lógica privilegiada.
M1.2	Deve-se evitar erros de cálculo decorrentes da falta de entendimento da representação interna da linguagem de programação usada e de como é realizada a interação com os aspectos de cálculo numérico. *Eg*., reconhecer representação de sinal, valores do tipo “Not-A-Number” (NaN), valores especiais, etc.
M1.3	Deve-se proteger as variáveis e os recursos compartilhados contra acessos concorrentes inapropriados.
M1.4	Deve-se utilizar mecanismos de bloqueio que evitem a ocorrência de requisições simultâneas feitas à aplicação ou utilizar um mecanismo de sincronização para evitar condições de concorrência (*race conditions*).
M1.5	Deve-se aumentar os privilégios da aplicação para um patamar mais elevado o mais tardiamente possível em relação ao fluxo de execução que necessita dos privilégios adicionais e revogar esses privilégios adicionais assim que não forem mais necessários.

2 VALIDAÇÃO DOS DADOS DE ENTRADA

Diretrizes para validação de dados de entrada do usuário e do recebimento de dados de outros sistemas a ser realizada antes do processamento dos dados.

ID	Diretriz
M2.1	Deve-se efetuar a validação de dados de entrada de fontes não-confiáveis. Eg., dados inseridos por usuários, base de dados externas, fluxos de arquivos, etc.
M2.2	Deve-se especificar o conjunto de caracteres apropriado — *eg*., UTF-8 —, para todas as fontes de entrada de dados.
M2.3	Deve-se codificar os dados de entrada para um conjunto de caracteres comuns antes de sua validação (*canonicalize*).
M2.4	Deve-se rejeitar dados de entrada quando há falha no sistema de validação.
M2.5	Deve-se validar todos os dados provenientes de clientes antes do processamento, incluindo todos os parâmetros, campos de formulário, mecanismos de *postback* automáticos em código embutidos — *eg*., Javascript —, conteúdos das URLs e cabeçalhos HTTP — *eg*., os nomes e os valores dos *Cookies*.
M2.6	Deve-se validar os dados advindos de redirecionamentos.
M2.7	Deve-se validar o tipo dos dados de entrada recebidos contra os esperados.
M2.8	Deve-se validar o intervalo dos dados de entrada recebidos contra os esperados.
M2.9	Deve-se validar o tamanho dos dados de entrada recebidos contra os esperados.
M2.10	Deve-se validar o formato dos dados de entrada, limitando quais caracteres são permitidos e garantindo que os dados seguem um padrão esperado. *Eg*., CPF, endereço de *e-mail*, número de telefone, CEP.
M2.11	Deve-se validar que URLs de redirecionamento dinâmico — *eg*., URLs recebidas por parâmetros — estejam em uma lista de URLs permitidas pelo sistema antes de realizar o redirecionamento ou, alternativamente, deve-se mostrar um aviso de redirecionamento para conteúdo potencialmente não confiável.
M2.12	Deve-se codificar as rotinas de validação de dados de entrada de maneira centralizada na aplicação.
M2.13	Deve-se efetuar a descodificação UTF-8 caso o sistema suporte o conjunto de caracteres extendidos de UTF-8.

M2.14	Deve-se rejeitar requisições e respostas cujos valores de cabeçalho não contenham apenas caracteres ASCII.
M2.15	Deve-se implementar controles adicionais de segurança caso caracteres potencialmente perigosos — *eg*., <, >, ", ', %, (,), &, +, \, \', \" — precisem ser permitidos na entrada de dados da aplicação.
M2.16	Deve-se aplicar verificações padrão para os dados de entrada, checando: a existência de *bytes* nulos como %00; a existência de caracteres de nova linha como %0d, %0a, \r, \n; a existência de caracteres “ponto-ponto barra” como “../” ou “..\” e a existência de alteradores de caminhos. No caso de um conjunto de caracteres em UTF-8, o sistema deve utilizar representações alternativas como %c0%ae%c0%ae/.
C2.1	Deve-se efetuar a validação de dados de entrada de qualquer fonte.
C2.2	Não se deve aceitar dados de entrada cujos caracteres estejam fora de uma lista de caracteres ou expressões regulares permitidas.

3 CODIFICAÇÃO DE SAÍDA

Diretrizes sobre a adequação de formato dos dados de saída e o preparo desses dados para interações com outros sistemas.

ID	Diretriz
M3.1	Deve-se codificar todos os caracteres, a menos que sejam conhecidos por serem seguros para o interpretador de destino.
M3.2	Deve-se escapar todos os dados provenientes de fontes não confiáveis, considerando o contexto em que serão usados. *Eg*., construção de consultas SQL, XML, LDAP e telas em HTML.
M3.3	Deve-se tratar dados provenientes de fontes que não sejam confiáveis e que gerem comandos para o sistema operacional.

4 AUTENTICAÇÃO E GERENCIAMENTO DE CREDENCIAIS

Diretrizes para a verificação da identidade de usuários ao realizarem operações nos sistemas e para determinação de identidade do usuário e seu correspondente nível de acesso às informações.

Observação. Autenticação AD *versus* OAuth2.

- No OAuth2, ao contrário do AD, a autenticação é feita diretamente em uma página externa, via HTTPS, sendo que o sistema web *não tem acesso* às credenciais inseridas pelo usuário. Depois do login, a página externa retorna um *token* para a aplicação. Esse *token* garante que o usuário foi autenticado corretamente.
- No OAuth2 é exigida uma conexão ativa com a Internet. Em situações de contingência onde o usuário não tenha acesso à página externa, não será possível autenticar-se no sistema.

ID	Diretriz
M4.1	Não se deve armazenar senhas em texto plano sem utilizar um algoritmo de *hash* seguro com *salt*.
M4.2	Deve-se utilizar controle de usuário e senha nominais para determinar a identidade unívoca do usuário, vedando-se o uso de credenciais por múltiplos usuários.
M4.3	Deve-se utilizar autenticação via AD e/ou o *framework* OAuth2 sempre que possível para autenticar usuários internos.
M4.4	Deve-se utilizar grupos de *Active Directory* (AD) para determinar as políticas de acesso e roles de usuário.
M4.5	Deve-se dar ciência ao usuário das permissões e níveis de acesso que possui.
M4.6	Deve-se utilizar HTTPS em todas as telas do sistema.
M4.7	Deve-se requerer autenticação para todas as páginas e recursos, exceto para aqueles que são intencionalmente públicos.
M4.8	Deve-se estabelecer e utilizar serviços de autenticação padronizados e testados.
M4.9	Deve-se utilizar uma implementação centralizada para realizar os procedimentos de autenticação, disponibilizando bibliotecas que invoquem os serviços externos de autenticação.
M4.10	Deve-se separar a lógica de autenticação do recurso que está sendo requisitado e

	usar redirecionadores nos controladores de autenticação centralizados.
M4.11	Não se deve indicar qual parte dos dados de autenticação está incorreta nas mensagens de falha na autenticação. *Eg*., em vez de exibir mensagens como “Nome de usuário incorreto” ou “Senha incorreta”, utilizar apenas mensagens como “Usuário e/ou senha inválidos” para ambos os casos de erro.
M4.12	Deve-se utilizar autenticação para conexão a sistemas externos que envolvam tráfego de informação sensível ou acesso a funções.
M4.13	Deve-se cifrar e armazenar em um local protegido de um sistema confiável as credenciais de autenticação para acessar serviços externos à aplicação.
M4.14	Não se deve armazenar as credenciais de autenticação no código-fonte da aplicação.
M4.15	Não se deve armazenar as credenciais de autenticação ou qualquer outro dado sensível em imagem de container.
M4.16	Deve-se notificar o usuário quando a sua senha for alterada.
M4.17	Deve-se comunicar a data/hora da última utilização — bem ou mal sucedida — de uma conta de usuário no próximo acesso ao sistema.
M4.18	Deve-se modificar todas as senhas e os identificadores de usuários (IDs) que, por padrão, são definidas pelos fornecedores.
M4.19	Deve-se exigir nova autenticação dos usuários antes da realização de operações críticas.
M4.20	Deve-se garantir que código de terceiros utilizado para o processo de autenticação não contém código malicioso.
M4.21	Não se deve validar os dados de autenticação antes do final de todas as entradas de dados, especialmente nas implementações de autenticação sequencial.
M4.22	Deve-se utilizar apenas requisições POST para transmitir credenciais de autenticação.
M4.23	Deve-se exigir a mudança de senhas temporárias na próxima vez que o usuário realizar a autenticação no sistema.
M4.24	Deve-se desativar a funcionalidade de lembrar a senha nos campos de senha do navegador.
M4.25	Deve-se armazenar de forma segura os dados de usuários e de sistemas que

	utilizam cada senha fornecida.
M4.26	Não se deve utilizar as mesmas senhas para ambientes de desenvolvimento, homologação ou produção.
C4.1	Deve-se utilizar certificado digital para determinar a identidade do usuário.
C4.2	Deve-se realizar monitoramento para identificar ataques contra várias contas de usuários que utilizem a mesma senha.
C4.3	Deve-se utilizar autenticação de múltiplos fatores (utilizando simultaneamente *token*, senha, biometria etc.).
C4.4	Deve-se garantir que, uma vez autenticado, o usuário não possa acessar o sistema de outro endereço IP, a menos que se autentique novamente.

5 GERENCIAMENTO DE SENHAS

Diretrizes para geração, distribuição e uso de senhas em sistemas computacionais. Os fatores examinados para o uso de senhas em *software* desenvolvido de maneira segura são:

- **Geração e parametrização de senhas.** Explana critérios para a escolha de senhas cuja finalidade é dificultar sua quebra por ataques de força-bruta ou adivinhação. O principal parâmetro considerado é o comprimento (tamanho) da senha. Inclui procedimentos para testes de força de senhas.
- **Armazenamento e distribuição de senhas.** Explana métodos para armazenamento seguro de senhas geradas tanto no lado validado (usuário, programa cliente, *etc.*) quanto no lado validador (*software*, sistema autenticador, *etc*.). Inclui métodos para a transmissão segura de senhas via rede e parâmetros para os procedimentos de mudança de senhas.
- ***Interface.*** Explana medidas necessárias à *interface* para as tentativas de validação de senhas. Inclui parametrização para determinar a frequência de tentativas permitidas para validação de uma senha e a apresentação da senha parcialmente submetida para o usuário.

ID	Diretriz
M5.1	Não se deve utilizar senhas com menos de 12 caracteres.
M5.2	Deve-se utilizar pelo menos letras maiúsculas e minúsculas, junto a ao menos um tipo de caractere (dígito, símbolo), seguindo o estabelecido pela Portaria 4.772/2008 do TRT4.
M5.3	Não se deve usar palavras comumente utilizadas para senhas (ou variantes destas). *Eg*., nome do animal de estimação, membro da família ou pessoa significativa; datas de aniversário; nome do feriado favorito; algo relacionado ao time esportivo favorito e as palavras “senha” e “password”.
M5.4	Não se deve armazenar senhas em claro.
M5.5	Deve-se armazenar ao menos o *hash* criptográfico com *salt*.
M5.6	Não se deve usar um canal em claro para a transmissão da senha ou elemento correspondente.
M5.7	Não se deve utilizar método de conferência menos seguro que desafios baseados em *hash* ou o uso de *hashes* armazenados.

M5.8	Não se deve mostrar diretamente a senha quando esta necessita ser digitada pelo usuário — deve haver opção de habilitar e desabilitar a visualização da senha digitada até então.
M5.9	Não se deve elaborar senhas sem auxílio de *software* gerador de senhas aleatórias, configurado para atender aos parâmetros aqui estabelecidos.
M5.10	Não se deve utilizar senha que não tenha sido validada por um *software* testador de força de senhas.
M5.11	Não se deve utilizar periodicidade de troca de senhas superior a 6 meses.
M5.12	Não se deve enviar a senha antiga para o usuário, em claro ou não.
M5.13	Não se deve armazenar senha que não esteja criptografada seguindo o nível mínimo de criptografia estabelecido neste documento.
M5.14	Não se deve permitir uma taxa de tentativas de validação de senha superior a 5 tentativas por minuto.
M5.15	Deve-se bloquear a conta de usuário em caso de 5 erros de autenticação consecutivos e sua reabilitação deve depender de processo específico.
C5.1	Não se deve utilizar senhas com menos de 20 caracteres.
C5.2	Não se deve utilizar senha que não tenha sido validada por um software testador de força de senhas diferente do software gerador de senhas.
C5.3	Não se deve armazenar senha que não esteja criptografada seguindo o nível forte de criptografia estabelecido neste documento.
C5.4	Deve-se utilizar um método de prova com conhecimento zero[3] de senha.
C5.5	Deve-se exigir prova de origem da requisição. *Eg*., *captchas* para demonstrar que o usuário é humano; assinatura digital para provar que requisição veio do sistema permitido.

[3] Blum, Manuel; Feldman, Paul; Micali, Silvio. Non-Interactive Zero-Knowledge and Its Applications. Proceedings of the twentieth annual ACM symposium on Theory of computing (STOC 1988): 103–112.

6 GERENCIAMENTO DE SESSÕES

Diretrizes para o gerenciamento de sessões de usuário, visando garantir a autenticidade e o correto exercício de permissões do usuário enquanto durar sua sessão no sistema.

Nesse escopo específico, as diretrizes estão subdivididas de acordo com o momento da sessão (início, manutenção, término), além de tópico específico sobre controle da sessão.

6.1 Controle de Sessão

ID	Diretriz
M6.1.1	Deve-se utilizar controles de gerenciamento de sessão baseados no servidor ou em *frameworks*.
M6.1.2	Não se deve definir o domínio e o caminho para os *cookies* que contenham identificadores de sessão autenticados para um endereço externo ao site.
M6.1.3	Deve-se configurar o atributo "secure" para *cookies* transmitidos através de uma conexão TLS.
M6.1.4	Deve-se configurar os cookies com o atributo “HttpOnly”, a menos que seja explicitamente necessário ler ou definir os valores dos mesmos através de scripts do lado cliente da aplicação.
C6.1.1	Deve-se utilizar mecanismos complementares ao mecanismo padrão de gerenciamento de sessões para operações sensíveis do lado servidor. *Eg*., no caso de operações de gerenciamento de contas através da utilização de *tokens* aleatórios ou parâmetros associados à sessão.

6.2 Criação de Sessão

ID	Diretriz
M6.2.1	Não se deve permitir o estabelecimento de sessão caso a aplicação não consiga ter acesso às informações contidas na configuração de segurança.
M6.2.2	Não se deve reconhecer identificadores gerados por controles fora do servidor ou do *framework* de controle como válidos.
M6.2.3	Não se deve permitir *logins* persistentes (sem prazo de expiração).

6.3 Manutenção de Sessão

ID	Diretriz
M6.3.1	Não se deve reaproveitar uma sessão estabelecida antes do *login* em caso de nova autenticação.
M6.3.2	Não se deve reaproveitar um identificador de sessão quando houver uma nova autenticação.
M6.3.3	Não se deve expor os identificadores de sessão em URLs, mensagens de erro ou *logs*.
M6.3.4	Deve-se proteger os dados de sessão do lado servidor contra acessos não autorizados por outros usuários do mesmo servidor, inclusive durante a sessão vigente.
M6.3.5	Deve-se notificar o usuário clara e constantemente a respeito do tempo de encerramento de sessão.
C6.3.1	Não se deve permitir conexões simultâneas com o mesmo identificador de sessão.

6.4 Término de Sessão

ID	Diretriz
M6.4.1	Deve-se encerrar completamente a sessão ou conexão associada no *logout*.
M6.4.2	Deve-se disponibilizar a funcionalidade de *logout* em todas as páginas que requerem autenticação.
M6.4.3	Deve-se estabelecer um tempo de expiração da sessão que seja o mais curto possível, baseado no balanceamento dos riscos e requisitos funcionais do negócio.
C6.4.1	Deve-se realizar o encerramento da sessão periodicamente, mesmo quando ela estiver ativa.

7 CONTROLE DE ACESSOS

Diretrizes e definições para a realização do controle de acessos a recursos do sistema, artefatos de desenvolvimento e partes sensíveis da aplicação.

ID	Diretriz
M7.1	Deve-se restringir o acesso às URLs protegidas somente aos usuários autorizados.
M7.2	Deve-se restringir o acesso às funções protegidas, às referências diretas aos objetos, aos serviços e aos dados da aplicação somente aos usuários autorizados.
M7.3	Deve-se restringir o acesso aos atributos e dados do usuário, às informações de políticas dos mecanismos de controle de acesso e às configurações de segurança relevantes somente aos usuários autorizados.
M7.4	Deve-se restringir o acesso a arquivos somente aos usuários autorizados.
M7.5	Não se deve utilizar o campo “referer” do cabeçalho como forma de verificação principal.
M7.6	Deve-se fazer a revalidação periódica da autorização do usuário para garantir que os privilégios não foram modificados.
M7.7	Não se deve atribuir privilégios além do mínimo às contas de serviço ou contas de suporte a conexões provenientes ou destinadas a serviços externos.
M7.8	Deve-se utilizar um único componente em toda a aplicação *Web* para realizar o processo de verificação de autorização de acesso — isto inclui bibliotecas que invocam os serviços externos de autorização.
M7.9	Deve-se exigir nova autenticação caso os privilégios do usuário tenham sido modificados durante uma sessão.
M7.10	Deve-se prover suporte à desativação de contas e ao encerramento das sessões quando terminar a autorização do usuário.
M7.11	Deve-se garantir o controle de autorização em todas as requisições, inclusive em scripts do lado servidor, "includes" e requisições provenientes de tecnologias do lado cliente, como AJAX.
M7.12	Deve-se criar uma política de controle de acesso para documentar as regras de negócio da aplicação, tipos de dados e critérios ou processos de autorização para

	que os acessos possam ser devidamente concedidos e controlados.
C7.1	Não se deve aplicar regras de controle de acesso representadas pela camada de apresentação divergentes das regras presentes no lado servidor.
C7.2	Deve-se implementar a auditoria das contas de usuário e assegurar a desativação de contas não utilizadas.
C7.4	Deve-se utilizar mecanismos de criptografia e verificação de integridade no lado servidor para detectar possíveis adulterações em dados armazenados no lado do cliente.
C7.5	Deve-se limitar o número de transações que um único usuário ou dispositivo pode executar em determinado período de tempo.

8 PRÁTICAS DE CRIPTOGRAFIA

Diretrizes para a configuração e utilização de algoritmos de criptografia e *hash* visando prover confidencialidade a dados.

ID	Diretriz
M8.1	Deve-se criptografar dados sigilosos e sensíveis.
M8.2	Deve-se utilizar um método criptográfico que siga o princípio de Kerckhoffs[4]; o método de encriptação e seus parâmetros devem ser públicos e estar documentados e somente a chave criptográfica deve ser mantida em sigilo.
M8.3	Não se deve utilizar um cifrador que admita um método conhecido para quebra da chave criptográfica melhor do que a força bruta, baseada em tentativa e erro.
M8.4	Não se deve utilizar o modo de cifrador de bloco *Electronic Codebook* (ECB) ou modos menos seguros.
M8.5	Não se deve utilizar um tamanho da chave menor que 128 *bits* (cifrador simétrico) ou 1024 *bits* (cifrador assimétrico).
M8.6	Não se deve utilizar função de *hash* sem algum tipo de *salt*.
M8.7	Não se deve utilizar módulos de criptografia incompatíveis com a FIPS 140-2[5] ou com um padrão equivalente.
M8.8	Não se deve utilizar algoritmos considerados obsoletos para criptografia e *hash* criptográfico. *Eg*., MD5, SHA1, DES/3DES, RC2, RC4, MD4.
M8.9	Não se deve distribuir chaves criptográficas sem a utilização de uma infraestrutura de chave pública e, portanto, sem a utilização de um cifrador assimétrico.
C8.1	Não se deve utilizar um tamanho da chave menor que 256 *bits* (cifrador simétrico) ou 4096 *bits* (cifrador assimétrico).
C8.2	Deve-se utilizar módulos criptográficos com geradores de números pseudo-aleatórios de alta aleatoriedade para a geração de todos os números, nomes de arquivos, GUIDs e *strings* aleatórias.

[4] Shannon, Claude (4 October 1949). "Communication Theory of Secrecy Systems". Bell System Technical Journal. 28: 662. Retrieved 20 June 2014.
[5] NIST: FIPS PUB 140-2: SECURITY REQUIREMENTS FOR CRYPTOGRAPHIC MODULES

C8.3	Deve-se utilizar *hashes* criptográficos sempre que possível, sobretudo nos seguintes casos: verificação da integridade de dados; armazenamento e verificação de senhas; provimento de identificador único para objetos em um sistema e geração de números pseudo-aleatórios.

9 TRATAMENTO E REGISTRO DE ERROS

Diretrizes para a manutenção de *logs* para posterior auditoria, rastreamento e consulta de incidentes ligados à segurança dos sistemas. Cada sistema possui uma criticidade diferente no que se refere à restrição de acesso a dados, não-repúdio e histórico de operações realizadas no banco de dados. Por esse motivo, essa seção não *define* quais informações devem ser auditadas, mas sim *sugere* possíveis itens que podem ser auditados, rastreados ou *logados*. Estes itens, então, devem ser avaliados pelos gestores do produto.

Exemplos de eventos que podem ser registrados:

- operações de *login* e *logout*;
- tentativas inválidas de *login*;
- acessos a determinadas telas ou seções do sistema;
- acesso a informações com alguma restrição (*eg*, documentos sigilosos, processos em segredo de justiça, dados pessoais ou bancários);
- operações de inclusão, alteração ou exclusão de registros no banco de dados;
- alteração de perfil de acesso (para sistemas que possuem acesso com diferentes perfis)
- criação e exclusão de usuário e
- execução de jobs e tarefas automatizadas.

Exemplos de informações que podem ser armazenadas, relativas a cada evento:

- data e hora (com indicação do fuso horário);
- usuário que efetuou a operação;
- endereço IP da origem do evento;
- tela (URL) do sistema de onde a operação foi realizada;
- identificador da instância (para sistemas *clusterizados*);
- para operações de inserção, alteração ou exclusão, o tipo da operação, nome da tabela que foi manipulada, ID do registro e, se for o caso, valores anterior e atual de cada campo;
- parâmetros informados pelo usuário (*eg*., parâmetros GET ou POST), tomando cuidado de não armazenar dados sensíveis, como senhas;
- tempo de resposta do sistema;
- para execução de *jobs* e tarefas automatizadas, armazenar o resultado da operação; falha, sucesso, cancelada, etc.

Exemplos de forma de captura dos dados para auditorias:

- alterações aplicadas no banco de dados podem ser auditadas via *triggers*[6];
- auditar as alterações a partir da própria aplicação[7] — algumas informações poderão não ser registradas (*eg*, operações SQL realizadas por fora da aplicação).
- Em sistemas *Web* desenvolvidos em Java, um *filtro*[8] pode interceptar as requisições feitas à aplicação.

ID	Diretriz
M9.1	Deve-se definir no documento de especificação de requisitos do sistema quais informações deverão ser registradas, o local e o tempo mínimo de armazenamento dos dados da auditoria.
M9.2	Não se deve expor informações sensíveis nas respostas aos erros, inclusive detalhes de sistema, identificadores de sessão ou informação da conta do usuário.
M9.3	Deve-se registrar em *log* todas as falhas de conexão TLS com o *backend*.
M9.4	Deve-se registrar em *log* todas as falhas que ocorreram nos módulos de criptografia.
M9.5	Deve-se registrar em *log* todas as exceções lançadas pelo sistema.
M9.6	Deve-se registrar em *log* todo o uso de funções administrativas, inclusive as mudanças realizadas nas configurações de segurança.
M9.7	Deve-se registrar em *log* todas as falhas de controle de acesso.
M9.8	Não se deve armazenar informações sensíveis nos registros de *logs*, como detalhes desnecessários do sistema, identificadores de sessão e senhas.
M9.9	Deve-se restringir o acesso aos *logs* apenas para pessoal autorizado.
M9.10	Não se deve permitir que as entradas de *log* que incluam dados não-confiáveis sejam executadas como código-fonte na interface de visualização de *logs*.
M9.11	Deve-se prover controles de *log* com suporte a casos de sucesso e a casos de falha relacionados aos eventos de segurança.
M9.12	Deve-se negar o acesso por padrão no tratamento de erros lógicos associados com os controles de segurança.

[6] Para aplicações que utilizam PostgreSQL, há uma proposta de rotina de auditoria de DML em https://wiki.postgresql.org/wiki/Audit_trigger_91plus .
[7] Para aplicações que utilizam Hibernate, é possível utilizar "Envers" https://docs.jboss.org/envers/docs/ ou outro *event listener*.
[8] Mais informações sobre filtros em http://www.oracle.com/technetwork/java/filters-137243.html .

M9.13	Deve-se utilizar mensagens de erro genéricas.
M9.14	Não se deve usar mecanismos de tratamento de erros que mostram informações de depuração (*debug*) ou informações da pilha de exceção.
M9.15	Deve-se definir no documento de especificação de requisitos do sistema quais são as políticas de retenção — tempo mínimo de armazenamento dos dados de auditoria — e de revisão dos *logs* — *ie*., procedimentos para revisar os logs, analisando se não há indícios de operações indevidas no sistema.
M9.16	Deve-se registrar em *log* as tentativas de conexão com tokens de sessão inválidos ou expirados.
M9.17	Deve-se registrar em *log* todos os eventos suspeitos de adulteração, inclusive alterações inesperadas no estado dos dados.
M9.18	Deve-se registrar em *log* todas as tentativas de autenticação, especialmente as que falharam.
M9.19	Deve-se utilizar uma rotina centralizada para realizar todas as operações de *log*.
M9.20	Deve-se utilizar páginas de erro personalizadas.
C9.1	Deve-se utilizar uma função de *hash* criptográfica para validar a integridade dos registros de *log*.
C9.2	Deve-se registrar em *log* todas as falhas de validação de entrada de dados.

10 PROTEÇÃO DE DADOS

Diretrizes que tratam do armazenamento de informações com grau de sigilo e de sua disponibilização. A seção define taxonomia para classificação de dados e descreve procedimentos para o armazenamento seguro dessa informação em *bancos de dados*. Também é detalhado o gerenciamento de permissões de acesso e distribuição de senhas a serem adotadas para operacionalização dessas estruturas.

No escopo deste documento os dados serão classificados, quanto ao seu *sigilo*, como:

- **Abertos.** Dados públicos (informação pública, conforme Resolução Administrativa TRT4 nº 01/2017), cujo conteúdo pode ou deve ser divulgado ao público externo.
- **Fechados.** Dados cujo acesso é restrito a um grupo específico de pessoas (informação ultrassecreta, secreta ou restrita, conforme Resolução Administrativa TRT4 nº 01/2017).

10.1 Procedimentos e Meios para Armazenamento de Dados Abertos

ID	Diretriz
M10.1.1	Deve-se utilizar meio de armazenamento que possua acesso para escrita restrito por senha.

10.2 Procedimentos e Meios para Armazenamento de Dados Fechados

ID	Diretriz
M10.2.1	Deve-se utilizar meio de armazenamento que possua acesso para leitura e escrita restrito por senha.
C10.2.1	Deve-se armazenar dados criptografados.

10.3 Permissões para Acesso a Informações em Bancos de Dados

ID	Diretriz
M10.3.1	Não se deve disponibilizar às aplicações acesso a algum banco de dados utilizando login de usuário com permissões de usuário *root* ou equivalente.
M10.3.2	Não se deve disponibilizar às aplicações acesso a algum banco de dados utilizando login de usuário com permissões para execução de comandos em *Data*

	Definition Language (DDL).
M10.3.3	Não se deve disponibilizar às aplicações acesso a algum banco de dados utilizando login de usuário com permissões além das estritamente necessárias ao seu funcionamento.
C10.3.1	Deve-se estabelecer correspondência um-para-um entre cada usuário de uma dada aplicação e do banco de dados.

10.4 Tratamento de Dados em Aplicações

ID	Diretriz
M10.4.1	Deve-se prover à aplicação a faculdade de remover dados sensíveis quando estes não forem mais necessários.
M10.4.2	Deve-se desativar a cache realizada no lado cliente das páginas que contenham informações sensíveis.
M10.4.3	Não se deve incluir informações sensíveis nos parâmetros de requisição HTTP GET.
M10.4.5	Não se deve publicar documentação do sistema que possa revelar informações importantes para potenciais atacantes.
M10.4.6	Deve-se implementar uma política de privilégio mínimo, restringindo os usuários apenas às funcionalidades, dados e informações do sistema que são necessárias para executarem suas tarefas.
M10.4.7	Deve-se proteger contra acesso não autorizado todas as cópias temporárias ou registradas em cache que contenham dados sensíveis e estejam armazenadas no servidor
M10.4.8	Deve-se criptografar informações altamente sensíveis quando armazenadas.
M10.4.9	Deve-se proteger o código-fonte presente no servidor para que não seja acessado por algum usuário sem permissão.
M10.4.10	Não se deve armazenar senhas, *strings* de conexão ou outras informações confidenciais em texto claro/legível ou em qualquer forma criptograficamente insegura no lado cliente.
M10.4.11	Deve-se remover comentários do código de produção que podem ser acessados pelos usuários.

M10.4.12	Deve-se excluir todas as cópias temporárias ou registradas em cache que contenham dados sensíveis e estejam armazenadas no servidor logo que não forem mais necessários.
M10.4.13	Deve-se desativar a funcionalidade de auto-completar nos formulários que contenham informações sensíveis, inclusive no formulário de autenticação.

11 SEGURANÇA NAS COMUNICAÇÕES

Diretrizes que tratam da transmissão segura de dados sensíveis entre sistemas, de modo a salvaguardar a integridade, autenticidade e demais atributos pertinentes ao uso dos dados comunicados.

ID	Diretriz
M11.1	Deve-se utilizar criptografia na transmissão de todas as informações sensíveis.
M11.2	Deve-se empregar canais de comunicação com controle de duplicação e perda de informações/mensagens.
M11.3	Deve-se empregar canal de comunicação que provenha controle de integridade dos dados transmitidos. *Eg*, HTTPS.
M11.4	Deve-se empregar canal de comunicação que provenha confidencialidade dos dados transmitidos. *Eg*, HTTPS, VPNs.
M11.5	Não se deve utilizar certificados TLS inválidos, com nome de domínio incorreto ou expirados.
M11.6	Não se deve fornecer uma conexão insegura quando ocorrer alguma falha nas conexões TLS.
M11.7	Deve-se utilizar TLS para conexões com sistemas externos que envolvam funções ou informações sensíveis.
M11.8	Deve-se empregar um canal de comunicação com controle de autenticação. *Eg*, HTTPS, certificados digitais gerados por autoridades confiáveis, VPNs.
M11.9	Deve-se armazenar de maneira segura os dados a serem transmitidos em ambas as extremidades da comunicação.
M11.10	Deve-se especificar a codificação dos caracteres para todas as conexões.
M11.11	Deve-se filtrar os parâmetros que contenham informações sensíveis, provenientes do “HTTP referer”, nos links para sites externos.
C11.1	Deve-se empregar canal de comunicação que provenha garantia de não-repúdio dos dados transmitidos. *Eg*, certificados digitais emitidos por entidades confiáveis.
C11.2	Deve-se utilizar logs confiáveis das informações transmitidas, com confirmação de entrega e recepção das mensagens. *Eg*., *WS-ReliableMessaging* para SOAP WS.
C11.3	Deve-se utilizar um padrão único de implementação TLS, configurado de modo apropriado.

12 CONFIGURAÇÕES DE SISTEMA

Diretrizes para a instalação, configuração e gerenciamento de ambientes de desenvolvimento de sistemas.

12.1 Configurações de Plataforma e Tratamento de Requisições

ID	Diretriz
M12.1.1	Deve-se restringir para o mínimo possível os privilégios do servidor *Web*, dos processos e das contas de serviços.
M12.1.2	Deve-se remover código de teste ou qualquer funcionalidade desnecessária para o ambiente de produção antes da instalação do sistema no servidor de produção.
M12.1.3	Deve-se definir quais métodos de requisição (*eg*., HTTP, GET ou POST) a aplicação irá suportar e se serão tratados de modo diferenciado nas diversas páginas da aplicação.
M12.1.4	Não se deve manter informações desnecessárias presentes nos cabeçalhos de resposta HTTP e que podem estar relacionadas com o sistema operacional, versão do servidor *Web* e *frameworks* de aplicação.
M12.1.5	Deve-se isolar o ambiente de desenvolvimento da rede de produção e conceder acesso somente para grupos de desenvolvimento e testes.
M12.1.6	Deve-se garantir que os servidores, *frameworks* e componentes do sistema estão executando a última versão aprovada, com as atualizações de segurança mais recentes e que sejam compatíveis com as necessidades do sistema.
M12.1.7	Deve-se desativar a listagem de diretórios do servidor *Web*.
M12.1.8	Deve-se configurar o arquivo *robots.txt* adequadamente de forma a prevenir a divulgação da estrutura de diretórios e impedir que robôs de busca façam indexação de arquivos que não devem ser indexados.
M12.1.9	Deve-se desativar as extensões HTTP desnecessárias. *Eg*.,WebDAV.
C12.1.1	Deve-se remover todas as funcionalidades e arquivos desnecessários.
C12.1.2	Deve-se certificar de que, no caso do servidor processar tanto requisições HTTP 1.0 como HTTP 1.1, ambas as versões estarão configuradas de modo semelhante.
C12.1.3	Deve-se implementar um sistema de gestão de ativos para manter o registro dos componentes e programas.

12.2 Cabeçalhos de Segurança HTTP

ID	Diretriz
M12.2.1	Deve-se configurar o cabeçalho “Content-Type” em todas as respostas do servidor. O conteúdo da resposta deve corresponder ao “Content-Type” configurado.
M12.2.2	Deve-se configurar o cabeçalho “Content-Security-Policy” (CSP) nas respostas HTTP para ajudar a reduzir o impacto de ataques XSS que usam injeção de HTML, DOM, JSON ou JavaScript.
M12.2.3	Deve-se configurar o cabeçalho “X-Content-Type-Options: nosniff” em todas as respostas do servidor.
M12.2.4	Deve-se configurar nas respostas do servidor o cabeçalho “X-Frame-Options” e/ou o cabeçalho “Content-Security-Policy:frame-ancestor” para prevenir a possibilidade da aplicação ser embutida em iframes de sites de terceiros não autorizados.
C12.2.1	Deve-se configurar o cabeçalho “Strict-Transport-Security” em todas as respostas e para todos os subdomínios, tal como “Strict-Transport-Security: max-age=15724800; includeSubdomains”.
C12.2.2	Deve-se configurar o cabeçalho “Referrer-Policy” para evitar o vazamento de informações presentes na URL para *websites* não confiáveis por meio do cabeçalho “Referer”.

12.3 Acesso ao Código-Fonte

Quanto ao sigilo do código-fonte dos sistemas desenvolvidos, devem ser, por padrão, de livre acesso aos servidores da SETIC. As demais situações deverão ser analisadas, projeto a projeto, pela chefia.

ID	Diretriz
M12.3.1	Deve-se utilizar um sistema controle de versões para provimento de rastreabilidade de modificações e controle de acesso ao código-fonte. *Eg*, SVN[9], Git[10], Mercurial[11].

[9] Fonte: https://subversion.apache.org .
[10] Fonte: https://git-scm.com/ .
[11] Fonte: https://www.mercurial-scm.org/ .

12.4 Separação de Ambientes

ID	Diretriz
M12.4.1	Deve-se utilizar bancos de dados distintos para cada ambiente.
M12.4.2	Deve-se utilizar servidores de aplicação/*Web* distintos para cada ambiente.
M12.4.3	Deve-se prover acesso ao ambiente de desenvolvimento/testes/homologação apenas aos integrantes da equipe de desenvolvimento e aos interessados no projeto (*stakeholders*).
M12.4.4	Deve-se prover um instalador expresso para a instalação do ambiente necessário para a execução de uma dada aplicação.
C12.4.1	Não se deve fornecer as senhas de acesso ao ambiente de produção aos desenvolvedores.
C12.4.2	Deve-se realizar testes periódicos para assegurar a segurança do ambiente de desenvolvimento/testes/homologação.

13 SEGURANÇA EM BANCOS DE DADOS

Diretrizes para reforço de práticas seguras na interação entre aplicações e bancos de dados.

ID	Diretriz
M13.1	Deve-se usar consultas parametrizadas fortemente tipadas.
M13.2	Deve-se certificar de que as variáveis são fortemente tipadas.
M13.3	Deve-se utilizar validação de entrada e codificação de saída; se houver falha, o comando não deverá ser executado no banco de dados.
M13.4	Deve-se realizar a codificação (*escaping*) de meta caracteres em instruções SQL.
M13.5	Não se deve incluir *strings* de conexão no código da aplicação.
M13.6	Deve-se eliminar o conteúdo desnecessário incluído por padrão pelo fornecedor. *Eg*., esquemas e bancos de dados de exemplo.
M13.7	Deve-se desativar todas as contas criadas por padrão e que não sejam necessárias para suportar os requisitos de negócio.
M13.8	Deve-se atribuir à aplicação o menor nível possível de privilégios ao acessar o banco de dados.
M13.9	Deve-se, sempre que possível, usar procedimentos armazenados (*stored procedures*) para abstrair o acesso aos dados e permitir a remoção de permissões das tabelas no banco de dados.
M13.10	Não se deve criar SQLs concatenando parâmetros textuais de origem não-segura, como parâmetros preenchidos pelo usuário ou mesmo armazenados no banco de dados.
M13.11	Deve-se utilizar tratamento especial para consultas que não podem ser parametrizadas, como *escapes* ou codificação em hexadecimal[12].
C13.1	Deve-se encerrar a conexão com o banco de dados assim que possível.

[12] A forma de utilização de *PreparedStatements* varia para cada linguagem. Mais detalhes podem ser encontrados em https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet.

14 GERENCIAMENTO DE ARQUIVOS

Diretrizes para edição, distribuição, armazenamento e concessão de permissões de arquivos.

ID	Diretriz
M14.1	Não se deve repassar dados fornecidos pelos usuários diretamente a uma função de inclusão dinâmica.
M14.2	Não se deve salvar arquivos no mesmo diretório de contexto da aplicação *Web*.
M14.3	Deve-se prevenir ou restringir o carregamento de qualquer arquivo que possa ser interpretado e/ou executado pelo servidor *Web*.
M14.4	Deve-se desativar privilégios de execução nos diretórios de armazenamento de arquivos.
M14.5	Não se deve passar caminhos de diretórios ou de arquivos em requisições.
M14.6	Não se deve enviar o caminho absoluto do arquivo para o lado cliente de uma aplicação ou para o usuário.
M14.7	Deve-se certificar de que os arquivos da aplicação e os recursos estão definidos somente com o atributo de leitura.
M14.8	Deve-se requerer autenticação antes de se permitir que seja feito o carregamento de arquivos.
M14.9	Deve-se validar se os arquivos enviados são do tipo esperado através da validação dos cabeçalhos.
M14.10	Deve-se usar uma lista branca (*whitelist*) de nomes e de tipos de arquivos permitidos ao referenciar arquivos.
C14.1	Deve-se limitar os tipos de arquivos que podem ser enviados para aceitar somente os necessários ao propósito do negócio.
C14.2	Deve-se implantar o carregamento seguro de arquivos nos ambientes UNIX por meio da montagem do diretório de destino como uma unidade lógica.
C14.3	Deve-se verificar os arquivos que os usuários submeterem através do mecanismo de carregamento em busca de vírus e *malwares*.

15 GERENCIAMENTO DE MEMÓRIA

Diretrizes para controle de memória envolvendo alocação, desalocação, estabelecimento de tamanho dedicado, tratamento de *buffers* e estabelecimento de limites.

ID	Diretriz
M15.1	Deve-se verificar se o *buffer* é, de fato, tão grande quanto o especificado.
M15.2	Deve-se verificar os limites do *buffer* caso as chamadas à função sejam realizadas em ciclos.
M15.3	Deve-se verificar se não há nenhum risco de ocorrer gravação de dados além do espaço reservado em *buffer*.
M15.4	Deve-se liberar a memória alocada de modo apropriado após concluir a sub-rotina (função/método) e em todos os pontos de saída.
M15.5	Deve-se truncar todas as *strings* de entrada para um tamanho razoável antes de passá-las para as funções de cópia e concatenação.
C15.1	Deve-se utilizar pilhas não-executáveis, quando disponíveis.

DIRETRIZES PARA DESENVOLVIMENTO SEGURO DE SOFTWARE

16 PREVENÇÃO, REAÇÃO E MITIGAÇÃO DE FALHAS DE SEGURANÇA

Diretrizes para a realização de procedimentos que garantam uma reação adequada à ocorrência de falhas de segurança. Detalha-se o emprego de *backups*, testes e tratamento de ocorrências.

16.1 Backups

A adequação às diretrizes de *backup* depende, muitas vezes, de políticas e atuação da área de infraestrutura, mas são importantes aspectos a serem considerados e monitorados no desenvolvimento de aplicações seguras.

ID	Diretriz
M16.1.1	Deve-se incluir no plano de projeto a especificação da necessidade e a atribuição da responsabilidade de realização de backups do banco de dados e dos códigos-fonte do sistema, bem como as políticas de acesso a este *backup*.
M16.1.2	Deve-se definir um procedimento estruturado para a restauração de *backups*.
M16.1.3	Deve-se definir e capacitar responsáveis pela recuperação dos *backups*.
C16.1.1	Deve-se criar *baselines* das versões do sistema, facilitando a recuperação ágil para uma versão anterior.
C16.1.2	Deve-se realizar simulações de restauração de dados continuamente.

16.2 Testes

ID	Diretriz
M16.2.1	Deve-se realizar testes manuais de segurança antes de cada versão do *software* em que sua estrutura tenha sido modificada. *Eg*., telas de *login*, serviços não autenticados, novos formulários com interação com o usuário, *etc*.
M16.2.2	Deve-se garantir, através de testes automatizados, que os serviços e dados sigilosos estão protegidos e disponíveis apenas para os usuários detentores das informações.
M16.2.3	Deve-se elaborar uma política de testes, automatizados ou não, visando a garantia de não vulnerabilidade aos principais ataques conhecidos em sistemas.
M16.2.4	Deve-se definir cenários de testes voltados à garantia dos requisitos não

	funcionais do *software*, preferencialmente realizado por uma equipe de testes diferente da equipe de desenvolvimento do *software*, com intuito de se evitar vícios.
M16.2.5	Deve-se definir cenários de testes, principalmente nos aspectos de segurança, para os casos de atualizações na arquitetura do sistema. *Eg*., servidores de aplicação, banco de dados, versões de navegador *Web*, versões de sistema operacional, etc.
C16.2.1	Deve-se propor constantes desafios entre as equipes para testar a segurança dos sistemas em formato de competição
C16.2.2	Deve-se submeter os sistemas a ferramentas de testes de invasão.
C16.2.3	Deve-se submeter imagens de container, de sistemas que utilizam essa tecnologia, à análise de vulnerabilidades.
C16.2.4	Deve-se submeter o código do sistema a ferramentas de análise estática de segurança (SAST).

16.3 Ocorrências

ID	Diretriz
M16.3.1	Deve-se manter procedimento planejado para imediata indisponibilização do sistema e realização de manutenção corretiva.
M16.3.2	Deve-se definir uma política de acompanhamento pós-correção de ocorrências de falha de segurança.
C16.3.1	Deve-se utilizar lições aprendidas nas ocorrências passadas para revisar a política de testes e incrementar a segurança dos sistemas.

17 CICLO DE VIDA DE *SOFTWARE*

Diretrizes para reforço da segurança de *software* nas diferentes fases de seu ciclo de vida; projeto, codificação e manutenção.

17.1 Projeto

As práticas aqui elencadas são consonantes com o que consta na portaria que institui a Política de Segurança da Informação neste Tribunal[13].

ID	Diretriz
M17.1.1	Deve-se empregar modelo de projeto de *software* que contemple etapa de modelagem de ameaças; definição clara dos riscos de segurança e nível de severidade que o comprometimento de dados sensíveis traria ao sistema e à instituição.
M17.1.2	Não se deve omitir, durante o projeto de desenvolvimento de sistema e sua execução, a definição de responsabilidades pela segurança de dados do sistema e como essa responsabilidade será verificada.
M17.1.3	Deve-se utilizar cronograma de projeto que contemple pontos de verificação de segurança do sistema desenvolvido ao longo de sua construção.

17.2 Codificação

ID	Diretriz
M17.2.1	Deve-se documentar, inclusive no código da aplicação, as medidas protetivas aplicadas no código-fonte, de modo a indicar precisamente o procedimento utilizado e suas peculiaridades.
M17.2.2	Deve-se utilizar mecanismos de verificação de integridade por *checksum* ou *hash* para verificar a integridade do código interpretado, bibliotecas, arquivos executáveis e arquivos de configuração.

[13] Tribunal Regional do Trabalho da 4a Região (2008). “Portaria No 4.772 de 23 de setembro de 2008”. Anexo 06.

17.3 Manutenção

ID	Diretriz
M17.3.1	Não se deve habilitar as atualizações automáticas de *software* ou componentes utilizados na construção de um sistema, sob pena de introdução indevida de falhas de segurança.
M17.3.2	Não se deve modificar *software* de terceiros, salvo quando estritamente necessário; controles de segurança internos podem ser invalidados. A mudança deve ser feita pelo desenvolvedor original do sistema sempre que possível.

17.4 Pessoal

ID	Diretriz
M17.4.1	Deve-se proporcionar treinamento e capacitação de programadores para aquisição e revisão de princípios de segurança computacional e desenvolvimento de *software* seguro.

GLOSSÁRIO

AD. Microsoft *Active Directory* - base de dados de identidades, autenticação e autorização utilizada internamente no TRT4.

Análise de vulnerabilidade. Atividade que tem por objetivo buscar por fragilidades existentes na aplicação e nas bibliotecas, componentes e infraestrutura por ela utilizados.

Autenticação. Ato de comprovação da identificação por meio de um ou mais fatores (senha, biometria, certificado digital, token, *One-Time Password*, etc).

Autorização. Uma vez autenticado com sucesso, o usuário passa a ter acesso a recursos e informações previamente concedidos para ele e/ou seu perfil.

Ataque de força-bruta. Tipo de ataque que busca alcançar seu objetivo (quebra de senha, tentativa de acesso indevido, etc) por meio de um número expressivo de tentativas e combinações possíveis.

Cifrador. É um par de algoritmos que realizam a encriptação e a decriptação.

Cifrador Assimétrico. É um cifrador que usa chaves diferentes, uma *pública*, uma *privada*, para encriptação e decriptação. Mais lentos, em geral, mas com usos para assinatura e verificação de autenticidade. Exemplos: Rivest-Shamir-Adleman (RSA)[14] e *Elliptic Curve Cryptography* (ECC)[15].

Cifrador de bloco. Cifrador que opera sobre blocos de bits de tamanho fixo com uma transformação invariável que é especificada por uma chave simétrica.

[14] Rivest, R.; Shamir, A.; Adleman, L. (February 1978). "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM. 21 (2): 120–126. doi:10.1145/359340.359342.
[15] Koblitz, N. (1987). "Elliptic curve cryptosystems". Mathematics of Computation. 48 (177): 203–209. JSTOR 2007884. doi:10.2307/2007884

Cifrador Simétrico. É um cifrador que usa a mesma chave para encriptação e decriptação. Mais rápidos, em geral. Exemplos: *Advanced Encryptation Standard* (AES)[16] e *Data Encryptation Standard* (DES)[17].

Chave. É uma sequência de bits utilizada como parâmetro secreto no cifrador, necessária para realizar encriptação e/ou decriptação. A única maneira de descobrir uma chave deve ser por *força-bruta*; tentar todas as alternativas no espaço de chaves possíveis. O algoritmo utilizado pelo cifrador deve garantir que chaves longas implicam em um tempo impraticável para descobrir a chave por tentativa-e-erro.

Decriptação. É o processo de converter texto cifrado em seu texto em claro original.

Encriptação. É o processo de converter texto em claro em texto cifrado.

Identificação. Ato de informar uma credencial de um usuário.

***Hash* Criptográfico.** É uma função matemática que mapeia uma entrada de tamanho arbitrário, em *bits*, para um saída de tamanho fixo e que é utilizada para criptografia. A função também é de "mão única", no sentido de que é impossível invertê-la. A função deve ser determinística, de rápida computação e de alta entropia.

Não-repúdio. Diz respeito à impossibilidade de negar a autoria de determinada ação.

***OAuth2*.** É um protocolo de autorização que possibilita que aplicativos/aplicações obtenham acesso limitado a contas de usuários em um serviço HTTP sem a necessidade de enviar seu usuário e senha.

***Open Relay*.** Os servidores de correio eletrônico são classificados como Open Relay quando ele processa um e-mail onde o remetente e o destinatário não são usuários do servidor em questão.

Princípio de Kerckhoffs. Princípio que estabelece que a segurança deve ser estabelecida

[16] Joan Daemen, Steve Borg e Vincent Rijmen, "The Design of Rijndael: AES - The Advanced Encryption Standard." Springer-Verlag, 2002. ISBN 3-540-42580-2.
[17] National Bureau of Standards, Data Encryption Standard, FIPS-Pub.46. National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977.

pela força da chave e não pelo segredo do método de criptografia.

REST. Protocolo para comunicação entre sistemas utilizando os métodos do protocolo HTTP.

***Salted Hash*.** Fragmento adicionado ao conteúdo original do hash para que a saída mude mesmo que o conteúdo original seja o mesmo.

SOAP. Protocolo para troca de mensagens em formato XML.

***SQL Injection*.** É uma forma de ataque em sistemas, realizado via interface, no qual o usuário informa trechos de SQL em campos de texto (ou até mesmo em telas de login ou de pesquisa), alterando a consulta prevista pelo desenvolvedor, sendo que o atacante poderá receber privilégios especiais ou poderá manipular indevidamente o banco de dados[18].

Static Application Security Testing (SAST) - Um conjunto de tecnologias desenvolvidas para analisar o código fonte, *byte code* e binários de aplicações buscando por indicativos de vulnerabilidades de segurança. Soluções SAST analisam a aplicação em um estado de não execução.

Teste de invasão. Atividade que tem por objetivo explorar falhas e vulnerabilidades existentes na aplicação e nas bibliotecas, componentes e infraestrutura por ela utilizados, com vistas a obter acesso indevido.

WS-ReliableMessaging. Protocolo para entrega segura de mensagens SOAP.

[18] Mais informações sobre *SQL Injection* podem ser encontrados em [https://www.owasp.org/index.php/ SQL_Injection]

REFERÊNCIAS E INDICAÇÕES

Algumas referências listadas abaixo não foram explicitamente usadas no texto deste documento, porém contribuíram de alguma forma para a produção deste material e ficarão registradas para consulta em futuras revisões.

Blum, Manuel; Feldman, Paul; Micali, Silvio. **Non-Interactive Zero-Knowledge and Its Applications**. Proceedings of the twentieth annual ACM symposium on Theory of computing (STOC 1988): 103–112. Disponível em https://dl.acm.org/doi/10.1145/62212.62222. Acesso em: 13/01/2023.

GovBr**. Guia de Segurança em Aplicações Web**. Disponível em: https://www.gov.br/governodigital/pt-br/seguranca-e-protecao-de-dados/guias/guia_seguranca_aplicacoesweb.pdf . Acesso em: 10/01/2023.

Institute for Security and Open Methodologies. **The Open Source Security Testing Methodology Manual**. Disponível em: https://www.isecom.org/OSSTMM.3.pdf. Acesso em:10/01/2023.

Joan Daemen, Steve Borg e Vincent Rijmen, **The Design of Rijndael: AES - The Advanced Encryption Standard.** Springer-Verlag, 2002.

Hibernate Community Documentation. **Hibernate Envers Reference Documentation**. Disponível em: https://docs.jboss.org/envers/docs/. Acesso em: 12/01/2023.

KOBLITZ, Neal. Elliptic curve cryptosystems. Mathematics Of Computation, [S.L.], v. 48, n. 177, p. 203-209, 1987. American Mathematical Society (AMS). http://dx.doi.org/10.1090/s0025-5718-1987-0866109-5.

National Bureau of Standards, **Data Encryption Standard**, FIPS-Pub.46. National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977.

National Cyber Security Centre. **Secure development and deployment guidance**. Disponível em: https://www.ncsc.gov.uk/collection/developers-collection. Acesso em: 10/01/2023.

NIST800-18. **Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities**. Disponível em: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf. Acesso em: 10/01/2023.

Oracle. **The Essentials of Filters**. Disponível em: https://www.oracle.com/java/technologies/filters.html. Acesso em 12/01/2023.

OWASP ASVS. **Application Security Verification Standard v4.0.3**. Disponível em: https://github.com/OWASP/ASVS/raw/v4.0.3/4.0/OWASP%20Application%20Security%20Verification%20Standard%204.0.3-en.pdf. Acesso em: 10/01/2023.

OWASP Cheat Sheet Series. **Sql Injection Prevention Cheat Sheet**. Disponível em: https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet. Acesso: 10/01/2023.

OWASP. **SQL Injection**. Disponível em: https://www.owasp.org/index.php/SQL_Injection. Acesso: 12/01/2023.

OWASP Proactive Controls. **OWASP Top 10 Proactive Controls 2018**. Disponível em: https://owasp-top-10-proactive-controls-2018.readthedocs.io/en/latest/index.html. Acesso em: 10/01/2023.

OWASP SCP. **Melhores Práticas de Codificação Segura OWASP: Guia de Referência Rápida**. Disponível em: https://github.com/OWASP/secure-coding-practices-quick-reference-guide/releases/download/v2.0.1/OWASP_SCP_Quick_Reference_Guide.pt-BR.pdf. Acesso em: 10/01/2023.

OWASP WTSG. **OWASP Web Security Testing Guide v4.2**. Disponível em: https://owasp.org/www-project-web-security-testing-guide/v42/. Acesso em: 13/01/2023.

PostgreSQL Wiki. **Audit trigger 91plus**. Disponível em: https://wiki.postgresql.org/wiki/Audit_trigger_91plus. Acesso em: 12/01/2023.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L.. A method for obtaining digital signatures and public-key cryptosystems. **Communications Of The Acm**, [S.L.], v. 21, n. 2, p. 120-126, fev. 1978. Association for Computing Machinery (ACM). http://dx.doi.org/10.1145/359340.359342.

Tribunal Regional do Trabalho da 4a Região (2008). **Portaria No 4.772 de 23 de setembro de 2008**. Anexo 06. Disponível em https://www.trt4.jus.br/portais/media/1507089/PSI%20-%20Compilada%20-%204920-2022.pdf. Acesso em: 12/01/2023.