

# Padrão de verificação de segurança do aplicativo 4.0.2

## Final

Outubro de 2020

Índice

[Frontispício 7](#_Toc54559837)

[Sobre o padrão 7](#_Toc54559838)

[Copyright e licença 7](#_Toc54559839)

[Líderes de Projeto 7](#_Toc54559840)

[Contribuintes principais 7](#_Toc54559841)

[Outros colaboradores e revisores 7](#_Toc54559842)

[Prefácio 8](#_Toc54559843)

[O que há de novo no 4.0 8](#_Toc54559844)

[Usando o ASVS 10](#_Toc54559845)

[Níveis de verificação de segurança do aplicativo 10](#_Toc54559846)

[Como usar este padrão 11](#_Toc54559847)

[Nível 1 - Primeiras etapas, automatizadas ou visualização de todo o portfólio 11](#_Toc54559848)

[Nível 2 - maioria dos aplicativos 11](#_Toc54559849)

[Nível 3 - alto valor, alta garantia ou alta segurança 11](#_Toc54559850)

[Aplicando ASVS na prática 12](#_Toc54559851)

[Como fazer referência aos requisitos ASVS 12](#_Toc54559852)

[Avaliação e Certificação 13](#_Toc54559853)

[A posição do OWASP sobre as certificações e marcas de confiança ASVS 13](#_Toc54559854)

[Orientação para Organizações Certificadoras 13](#_Toc54559855)

[Método de Teste 13](#_Toc54559856)

[Outros usos para o ASVS 14](#_Toc54559857)

[Como orientação detalhada da arquitetura de segurança 14](#_Toc54559858)

[Como uma substituição para listas de verificação de codificação segura disponíveis no mercado 14](#_Toc54559859)

[Como um guia para unidades automatizadas e testes de integração 14](#_Toc54559860)

[Para treinamento de desenvolvimento seguro 14](#_Toc54559861)

[Como um driver para segurança ágil de aplicativos 14](#_Toc54559862)

[Como uma estrutura para orientar a aquisição de software seguro 14](#_Toc54559863)

[V1: Requisitos de arquitetura, design e modelagem de ameaças 16](#_Toc54559864)

[Objetivo de Controle 16](#_Toc54559865)

[Requisitos de ciclo de vida de desenvolvimento de software seguro V1.1 16](#_Toc54559866)

[Requisitos de arquitetura de autenticação V1.2 17](#_Toc54559867)

[Requisitos de arquitetura de gerenciamento de sessão V1.3 17](#_Toc54559868)

[Requisitos de arquitetura de controle de acesso V1.4 17](#_Toc54559869)

[Requisitos de arquitetura de entrada e saída V1.5 18](#_Toc54559870)

[Requisitos de arquitetura criptográfica V1.6 18](#_Toc54559871)

[Erros V1.7, requisitos arquitetônicos de registro e auditoria 19](#_Toc54559872)

[Requisitos de arquitetura de privacidade e proteção de dados V1.8 19](#_Toc54559873)

[Requisitos de arquitetura de comunicações V1.9 19](#_Toc54559874)

[Requisitos de arquitetura de software malicioso V1.10 19](#_Toc54559875)

[Requisitos de arquitetura de lógica de negócios V1.11 19](#_Toc54559876)

[Requisitos de arquitetura para upload seguro de arquivo V1.12 20](#_Toc54559877)

[Requisitos de arquitetura da API V1.13 20](#_Toc54559878)

[Requisitos de arquitetura de configuração V1.14 20](#_Toc54559879)

[Referências 20](#_Toc54559880)

[V2: Requisitos de verificação de autenticação 21](#_Toc54559881)

[Objetivo de Controle 21](#_Toc54559882)

[NIST 800-63 - padrão de autenticação moderno baseado em evidências 21](#_Toc54559883)

[Selecionando um Nível NIST AAL apropriado 21](#_Toc54559884)

[Lenda 21](#_Toc54559885)

[Requisitos de segurança de senha V2.1 22](#_Toc54559886)

[Requisitos gerais do autenticador V2.2 23](#_Toc54559887)

[Requisitos de ciclo de vida do autenticador V2.3 24](#_Toc54559888)

[Requisitos de armazenamento de credenciais V2.4 24](#_Toc54559889)

[Requisitos de recuperação de credencial V2.5 25](#_Toc54559890)

[V2.6 Requisitos do verificador de segredo de pesquisa 26](#_Toc54559891)

[Requisitos do verificador fora de banda V2.7 26](#_Toc54559892)

[V2.8 Requisitos do verificador único ou multifatorial 27](#_Toc54559893)

[V2.9 Software criptográfico e requisitos do verificador de dispositivos 27](#_Toc54559894)

[Requisitos de autenticação de serviço V2.10 28](#_Toc54559895)

[Requisitos adicionais da agência dos EUA 28](#_Toc54559896)

[Glossário de termos 28](#_Toc54559897)

[Referências 29](#_Toc54559898)

[V3: Requisitos de verificação de gerenciamento de sessão 30](#_Toc54559899)

[Objetivo de Controle 30](#_Toc54559900)

[Requisitos de verificação de segurança 30](#_Toc54559901)

[Requisitos fundamentais de gerenciamento de sessão V3.1 30](#_Toc54559902)

[Requisitos de vinculação de sessão V3.2 30](#_Toc54559903)

[Requisitos de tempo limite e logoff da sessão V3.3 30](#_Toc54559904)

[V3.4 Gerenciamento de sessão baseado em cookie 31](#_Toc54559905)

[Gerenciamento de sessão baseado em token V3.5 31](#_Toc54559906)

[V3.6 Reautenticação de uma Federação ou Asserção 32](#_Toc54559907)

[V3.7 Defesas contra exploits de gerenciamento de sessão 32](#_Toc54559908)

[Descrição do Ataque semi-aberto 32](#_Toc54559909)

[Referências 33](#_Toc54559910)

[V4: Requisitos de verificação de controle de acesso 34](#_Toc54559911)

[Objetivo de Controle 34](#_Toc54559912)

[Requisitos de verificação de segurança 34](#_Toc54559913)

[Projeto de controle de acesso geral V4.1 34](#_Toc54559914)

[V4.2 Controle de acesso de nível de operação 34](#_Toc54559915)

[V4.3 Outras Considerações de Controle de Acesso 34](#_Toc54559916)

[Referências 35](#_Toc54559917)

[V5: Requisitos de validação, higienização e verificação de codificação 36](#_Toc54559918)

[Objetivo de Controle 36](#_Toc54559919)

[Requisitos de validação de entrada V5.1 36](#_Toc54559920)

[Requisitos de sanitização e sandbox da V5.2 37](#_Toc54559921)

[Requisitos de codificação de saída e prevenção de injeção V5.3 37](#_Toc54559922)

[Requisitos de memória, string e código não gerenciado V5.4 38](#_Toc54559923)

[Requisitos de prevenção de desserialização V5.5 38](#_Toc54559924)

[Referências 39](#_Toc54559925)

[V6: Requisitos de verificação de criptografia armazenada 40](#_Toc54559926)

[Objetivo de Controle 40](#_Toc54559927)

[Classificação de dados V6.1 40](#_Toc54559928)

[Algoritmos V6.2 40](#_Toc54559929)

[Valores Aleatórios V6.3 41](#_Toc54559930)

[Gerenciamento de segredo V6.4 41](#_Toc54559931)

[Referências 41](#_Toc54559932)

[V7: Tratamento de erros e requisitos de verificação de registro 42](#_Toc54559933)

[Objetivo de Controle 42](#_Toc54559934)

[Requisitos de conteúdo de log V7.1 42](#_Toc54559935)

[Requisitos de processamento de log V7.2 42](#_Toc54559936)

[Requisitos de proteção de log V7.3 43](#_Toc54559937)

[Tratamento de erros V7.4 43](#_Toc54559938)

[Referências 43](#_Toc54559939)

[V8: Requisitos de verificação de proteção de dados 45](#_Toc54559940)

[Objetivo de Controle 45](#_Toc54559941)

[V8.1 Proteção Geral de Dados 45](#_Toc54559942)

[Proteção de dados do lado do cliente V8.2 45](#_Toc54559943)

[V8.3 Dados Privados Sensíveis 46](#_Toc54559944)

[Referências 46](#_Toc54559945)

[V9: Requisitos de verificação de comunicações 48](#_Toc54559946)

[Objetivo de Controle 48](#_Toc54559947)

[Requisitos de segurança de comunicações do cliente V9.1 48](#_Toc54559948)

[Requisitos de segurança de comunicações do servidor V9.2 48](#_Toc54559949)

[Referências 49](#_Toc54559950)

[V10: Requisitos de verificação de código malicioso 50](#_Toc54559951)

[Objetivo de Controle 50](#_Toc54559952)

[Controles de integridade de código V10.1 50](#_Toc54559953)

[V10.2 Pesquisa de código malicioso 50](#_Toc54559954)

[V10.3 Controles de integridade de aplicativos implantados 51](#_Toc54559955)

[Referências 51](#_Toc54559956)

[V11: Requisitos de verificação de lógica de negócios 52](#_Toc54559957)

[Objetivo de Controle 52](#_Toc54559958)

[Requisitos de segurança da lógica de negócios V11.1 52](#_Toc54559959)

[Referências 52](#_Toc54559960)

[V12: Requisitos de verificação de arquivos e recursos 54](#_Toc54559961)

[Objetivo de Controle 54](#_Toc54559962)

[Requisitos de upload de arquivo V12.1 54](#_Toc54559963)

[Requisitos de integridade de arquivo V12.2 54](#_Toc54559964)

[Requisitos de execução de arquivo V12.3 54](#_Toc54559965)

[Requisitos de armazenamento de arquivo V12.4 55](#_Toc54559966)

[Requisitos de download de arquivo V12.5 55](#_Toc54559967)

[Requisitos de proteção V12.6 SSRF 55](#_Toc54559968)

[Referências 55](#_Toc54559969)

[V13: Requisitos de verificação de API e serviço da Web 56](#_Toc54559970)

[Objetivo de Controle 56](#_Toc54559971)

[Requisitos de verificação de segurança de serviço da web genérico V13.1 56](#_Toc54559972)

[Requisitos de verificação de serviço da Web RESTful V13.2 56](#_Toc54559973)

[Requisitos de verificação de serviço da Web SOAP V13.3 57](#_Toc54559974)

[V13.4 GraphQL e outros requisitos de segurança da camada de dados de serviço da web 57](#_Toc54559975)

[Referências 57](#_Toc54559976)

[V14: Requisitos de verificação de configuração 59](#_Toc54559977)

[Objetivo de Controle 59](#_Toc54559978)

[Versão V14.1 59](#_Toc54559979)

[Dependência V14.2 60](#_Toc54559980)

[V14.3 Requisitos de divulgação de segurança não intencionais 60](#_Toc54559981)

[Requisitos de cabeçalhos de segurança HTTP V14.4 61](#_Toc54559982)

[V14.5 Validar Requisitos de Cabeçalho de Solicitação HTTP 61](#_Toc54559983)

[Referências 61](#_Toc54559984)

[Apêndice A: Glossário 63](#_Toc54559985)

[Apêndice B: Referências 66](#_Toc54559986)

[Projetos Centrais OWASP 66](#_Toc54559987)

[Projeto OWASP Cheat Sheet Series 66](#_Toc54559988)

[Projetos relacionados à segurança móvel 66](#_Toc54559989)

[Projetos relacionados à Internet das Coisas da OWASP 66](#_Toc54559990)

[Projetos sem servidor OWASP 66](#_Toc54559991)

[Outras 66](#_Toc54559992)

[Apêndice C: Requisitos de verificação da Internet das coisas 67](#_Toc54559993)

[Objetivo de Controle 67](#_Toc54559994)

[Requisitos de verificação de segurança 67](#_Toc54559995)

[Referências 69](#_Toc54559996)

# Frontispício

## Sobre o padrão

O Application Security Verification Standard é uma lista de requisitos ou testes de segurança de aplicativos que podem ser usados ​​por arquitetos, desenvolvedores, testadores, profissionais de segurança, fornecedores de ferramentas e consumidores para definir, construir, testar e verificar aplicativos seguros.

## Copyright e licença

Versão 4.0.2, outubro de 2020



Copyright © 2008-2020 The OWASP Foundation. Este documento foi lançado sob o[Licença Creative Commons Attribution ShareAlike 3.0](https://creativecommons.org/licenses/by-sa/3.0/). Para qualquer reutilização ou distribuição, você deve deixar claro para os outros os termos da licença desta obra.

## Líderes de Projeto

| Andrew van der Stock | Daniel Cuthbert | Jim Manico |
| --- | --- | --- |
| Josh C Grossman | Mark Burnett |  |

## Contribuintes principais

| Abhay Bhargav | Benedikt Bauer | Elar Lang |
| --- | --- | --- |
| Osama Elnaggar | Ron Perris | Tonimir Kisasondi |

## Outros colaboradores e revisores

| Aaron Guzman | Anthony Weems | Barbara Schachner | Christopher Loessl | Clément Notin |
| --- | --- | --- | --- | --- |
| Dan Cornell | Daniël Geerts | David Clarke | David Johansson | David Quisenberry |
| Erlend Oftedal | Fatih Ersinadim | Filip van Laenen | Geoff Baskwill | Glenn ten Cate |
| Grant Ongers | olá7s | Jacob Salassi | James Sulinski | Jason Axley |
| Jason Morrow | Javier Dominguez | Jet Anderson | Jim Newman | Jonathan Schnittger |
| Joseph Kerby | Kelby Ludwig | Lars Haulin | Lewis Ardern | código lyz |
| Marc Aubry | Marco Schnüriger | Philippe De Ryck | Ralph Andalis | Ravi Balla |
| Rick Mitchell | Riotaro Okada | Robin Wood | Rogan Dawes | Ryan Goltry |
| Sajjad Pourali | Serg Belkommen | Siim Puustusmaa | Ståle Pettersen | Stuart Gunter |
| Tal Argoni | Tomasz Wrobel | Vincent De Schutter |  |  |

Se um crédito estiver faltando na lista de crédito 4.0.2 acima, registre um tíquete no GitHub para ser reconhecido em futuras atualizações 4.x.

O Application Security Verification Standard é construído sobre os ombros dos envolvidos do ASVS 1.0 em 2008 ao 3.0 em 2016. Grande parte da estrutura e dos itens de verificação que ainda estão no ASVS hoje foram originalmente escritos por Mike Boberski, Jeff Williams e Dave Wichers, mas existem muitos mais contribuidores. Obrigado a todos os envolvidos anteriormente. Para obter uma lista abrangente de todos aqueles que contribuíram para as versões anteriores, consulte cada versão anterior.

# Prefácio

Bem-vindo ao Application Security Verification Standard (ASVS) versão 4.0. O ASVS é um esforço conduzido pela comunidade para estabelecer uma estrutura de requisitos e controles de segurança que se concentram na definição dos controles de segurança funcionais e não funcionais necessários ao projetar, desenvolver e testar aplicativos e serviços da Web modernos.

A versão 4.0.2 é o segundo patch menor para a v4.0 com o objetivo de corrigir erros de ortografia e tornar os requisitos mais claros sem fazer alterações significativas, como alterar requisitos materialmente ou adicionar / remover requisitos.

ASVS v4.0 é o culminar do esforço da comunidade e feedback da indústria na última década. Tentamos tornar mais fácil adotar o ASVS para uma variedade de casos de uso diferentes em qualquer ciclo de vida de desenvolvimento de software seguro.

Esperamos que provavelmente nunca haja 100% de acordo sobre o conteúdo de qualquer padrão de aplicativo da web, incluindo o ASVS. A análise de risco é sempre subjetiva até certo ponto, o que cria um desafio ao tentar generalizar em um padrão único. No entanto, esperamos que as atualizações mais recentes feitas nesta versão sejam um passo na direção certa e aprimorem os conceitos introduzidos neste padrão crítico da indústria.

## O que há de novo no 4.0

A mudança mais significativa nesta versão é a adoção das Diretrizes de Identidade Digital NIST 800-63-3, introduzindo controles de autenticação modernos, baseados em evidências e avançados. Embora esperemos algum retrocesso no alinhamento com um padrão de autenticação avançado, sentimos que é essencial que os padrões sejam alinhados, principalmente quando outro padrão de segurança de aplicativo bem considerado é baseado em evidências.

Os padrões de segurança da informação devem tentar minimizar o número de requisitos exclusivos, para que as organizações em conformidade não tenham que decidir sobre controles concorrentes ou incompatíveis. O OWASP Top 10 2017 e agora o OWASP Application Security Verification Standard estão alinhados com o NIST 800-63 para autenticação e gerenciamento de sessão. Encorajamos outros órgãos de definição de padrões a trabalhar conosco, NIST e outros para chegar a um conjunto geralmente aceito de controles de segurança de aplicativos para maximizar a segurança e minimizar os custos de conformidade.

ASVS 4.0 foi totalmente renumerado do início ao fim. O novo esquema de numeração nos permitiu fechar lacunas de capítulos há muito desaparecidos e nos permitir segmentar capítulos mais longos para minimizar o número de controles que um desenvolvedor ou equipe deve cumprir. Por exemplo, se um aplicativo não usa JWT, toda a seção sobre JWT no gerenciamento de sessão não é aplicável.

A novidade do 4.0 é um mapeamento abrangente para o Common Weakness Enumeration (CWE), uma das solicitações de recursos mais desejadas que recebemos na última década. O mapeamento CWE permite que os fabricantes de ferramentas e aqueles que usam software de gerenciamento de vulnerabilidade correspondam os resultados de outras ferramentas e versões anteriores do ASVS para 4.0 e posteriores. Para abrir espaço para a entrada do CWE, tivemos que retirar a coluna "Desde", que, conforme renumeramos completamente, faz menos sentido do que nas versões anteriores do ASVS. Nem todos os itens no ASVS têm um CWE associado e, como o CWE tem muitas duplicações, tentamos usar o mais comumente usado, em vez de necessariamente a correspondência mais próxima. Os controles de verificação nem sempre são mapeáveis ​​para pontos fracos equivalentes.

Trabalhamos para atender e superar de forma abrangente os requisitos para abordar o OWASP Top 10 2017 e os OWASP Proactive Controls 2018. Como o OWASP Top 10 2017 é o mínimo para evitar negligência, deliberadamente tornamos todos os 10 principais requisitos de registro específico em Nível 1 controles, tornando mais fácil para os 10 maiores usuários do OWASP atingirem um padrão de segurança real.

Nós nos propusemos a garantir que o ASVS 4.0 Nível 1 seja um superconjunto abrangente do PCI DSS 3.2.1 Seções 6.5, para design de aplicativos, codificação, testes, revisões de código seguro e testes de penetração. Isso exigia a cobertura de estouro de buffer e operações de memória inseguras na V5 e sinalizadores de compilação relacionados à memória insegura na V14, além dos requisitos de verificação de serviço da web e aplicativos líderes de mercado existentes.

Concluímos a mudança do ASVS de controles monolíticos apenas do lado do servidor para fornecer controles de segurança para todos os aplicativos e APIs modernos. Nos dias de programação funcional, API sem servidor, móvel, nuvem, contêineres, CI / CD e DevSecOps, federação e muito mais, não podemos continuar a ignorar a arquitetura de aplicativo moderna. Os aplicativos modernos são projetados de forma muito diferente daqueles construídos quando o ASVS original foi lançado em 2009. O ASVS deve sempre olhar para o futuro distante para que possamos fornecer bons conselhos para nosso público principal - os desenvolvedores. Esclarecemos ou eliminamos qualquer requisito que presuma que os aplicativos são executados em sistemas pertencentes a uma única organização.

Devido ao tamanho do ASVS 4.0, bem como ao nosso desejo de nos tornarmos o ASVS de base para todos os outros esforços do ASVS, retiramos a seção móvel, em favor do Mobile Application Security Verification Standard (MASVS). O apêndice da Internet das Coisas aparecerá em um futuro IoT ASVS do OWASP Internet of Things Project. Incluímos uma prévia do ASVS da IoT no Apêndice C. Agradecemos tanto a Equipe OWASP Mobile quanto a Equipe do Projeto OWASP IoT por seu apoio ao ASVS e esperamos trabalhar com eles no futuro para fornecer padrões complementares.

Por último, desduplicamos e retiramos os controles menos impactantes. Com o tempo, o ASVS passou a ser um conjunto abrangente de controles, mas nem todos os controles são iguais na produção de software seguro. Esse esforço para eliminar itens de baixo impacto pode ir mais longe. Em uma edição futura do ASVS, o Common Weakness Scoring System (CWSS) ajudará a priorizar ainda mais os controles que são realmente importantes e aqueles que devem ser retirados.

A partir da versão 4.0, o ASVS se concentrará exclusivamente em ser o padrão de serviço e aplicativos da web líder, cobrindo a arquitetura de aplicativos tradicional e moderna, práticas de segurança ágeis e cultura DevSecOps.

# Usando o ASVS

ASVS tem dois objetivos principais:

* para ajudar as organizações a desenvolver e manter aplicativos seguros.
* para permitir que fornecedores de serviços de segurança, fornecedores de ferramentas de segurança e consumidores alinhem seus requisitos e ofertas.

## Níveis de verificação de segurança do aplicativo

O Application Security Verification Standard define três níveis de verificação de segurança, com cada nível aumentando em profundidade.

* ASVS Nível 1 é para níveis de garantia baixos e é totalmente testável de penetração
* ASVS Nível 2 é para aplicativos que contêm dados confidenciais, o que requer proteção e é o nível recomendado para a maioria dos aplicativos
* O ASVS Nível 3 é para os aplicativos mais críticos - aplicativos que realizam transações de alto valor, contêm dados médicos confidenciais ou qualquer aplicativo que requer o mais alto nível de confiança.

Cada nível ASVS contém uma lista de requisitos de segurança. Cada um desses requisitos também pode ser mapeado para recursos e capacidades específicos de segurança que devem ser integrados ao software pelos desenvolvedores.



Figura 1 - Níveis do padrão 4.0 de verificação de segurança do aplicativo OWASP

O nível 1 é o único nível completamente testável de penetração em humanos. Todos os outros requerem acesso à documentação, código-fonte, configuração e às pessoas envolvidas no processo de desenvolvimento. No entanto, mesmo se L1 permitir a ocorrência de testes de "caixa preta" (sem documentação e sem fonte), não é uma atividade de garantia eficaz e deve ser ativamente desencorajada. Os invasores mal-intencionados têm muito tempo, a maioria dos testes de penetração termina em algumas semanas. Os defensores precisam criar controles de segurança, proteger, encontrar e resolver todos os pontos fracos e detectar e responder a agentes mal-intencionados em um tempo razoável. Atores mal-intencionados têm essencialmente um tempo infinito e requerem apenas uma única defesa porosa, uma única fraqueza ou detecção ausente para ter sucesso. Teste de caixa preta, muitas vezes realizado no final do desenvolvimento, rapidamente,

Nos últimos mais de 30 anos, os testes de caixa preta provaram repetidamente que não observam problemas de segurança críticos que levaram diretamente a violações cada vez mais massivas. Incentivamos fortemente o uso de uma ampla gama de garantia e verificação de segurança, incluindo a substituição de testes de penetração por testes de penetração conduzidos por código-fonte (híbridos) no Nível 1, com acesso total aos desenvolvedores e documentação durante todo o processo de desenvolvimento. Os reguladores financeiros não toleram auditorias financeiras externas sem acesso aos livros, transações de amostra ou às pessoas que executam os controles. A indústria e os governos devem exigir o mesmo padrão de transparência no campo da engenharia de software.

Nós encorajamos fortemente o uso de ferramentas de segurança dentro do próprio processo de desenvolvimento. As ferramentas DAST e SAST podem ser usadas continuamente pelo pipeline de construção para encontrar problemas de segurança fáceis de encontrar que nunca deveriam estar presentes.

Ferramentas automatizadas e varreduras online são incapazes de completar mais da metade do ASVS sem ajuda humana. Se for necessária uma automação de teste abrangente para cada construção, então uma combinação de unidade customizada e testes de integração, junto com varreduras online iniciadas pela construção, são usadas. Falhas de lógica de negócios e testes de controle de acesso só são possíveis com auxílio humano. Eles devem ser transformados em testes de unidade e integração.

## Como usar este padrão

Uma das melhores maneiras de usar o Application Security Verification Standard é usá-lo como um plano para criar uma Lista de verificação de codificação segura específica para seu aplicativo, plataforma ou organização. Ajustar o ASVS para seus casos de uso aumentará o foco nos requisitos de segurança que são mais importantes para seus projetos e ambientes.

### Nível 1 - Primeiras etapas, automatizadas ou visualização de todo o portfólio

Um aplicativo atinge o ASVS Nível 1 se ele se defende adequadamente contra vulnerabilidades de segurança do aplicativo que são fáceis de descobrir e incluídas no OWASP Top 10 e outras listas de verificação semelhantes.

O nível 1 é o mínimo absoluto pelo qual todos os aplicativos devem se esforçar. Também é útil como uma primeira etapa em um esforço de várias fases ou quando os aplicativos não armazenam ou manipulam dados confidenciais e, portanto, não precisam dos controles mais rigorosos do Nível 2 ou 3. Os controles do Nível 1 podem ser verificados automaticamente por ferramentas ou simplesmente manualmente, sem acesso ao código-fonte. Consideramos o Nível 1 o mínimo necessário para todas as aplicações.

As ameaças ao aplicativo provavelmente virão de invasores que estão usando técnicas simples e de baixo esforço para identificar vulnerabilidades fáceis de encontrar e explorar. Isso contrasta com um invasor determinado que gastará energia concentrada para visar especificamente o aplicativo. Se os dados processados ​​por seu aplicativo tiverem alto valor, raramente você desejará parar em uma revisão de Nível 1.

### Nível 2 - maioria dos aplicativos

Um aplicativo atinge ASVS Nível 2 (ou Padrão) se ele se defende adequadamente contra a maioria dos riscos associados ao software hoje.

O nível 2 garante que os controles de segurança estejam implantados, sejam eficazes e sejam usados ​​no aplicativo. O Nível 2 é normalmente apropriado para aplicativos que lidam com transações significativas entre empresas, incluindo aquelas que processam informações de saúde, implementam funções críticas ou confidenciais para os negócios, ou processam outros ativos confidenciais, ou setores onde a integridade é uma faceta crítica para proteger seus negócios , como a indústria de jogos para impedir trapaceiros e hacks de jogos.

Ameaças aos aplicativos de Nível 2 serão tipicamente invasores qualificados e motivados, com foco em alvos específicos, usando ferramentas e técnicas altamente praticadas e eficazes na descoberta e exploração de pontos fracos nos aplicativos.

### Nível 3 - alto valor, alta garantia ou alta segurança

ASVS Nível 3 é o nível mais alto de verificação dentro do ASVS. Este nível é normalmente reservado para aplicativos que requerem níveis significativos de verificação de segurança, como aqueles que podem ser encontrados em áreas militares, de saúde e segurança, infraestrutura crítica, etc.

As organizações podem exigir ASVS Nível 3 para aplicativos que executam funções críticas, onde a falha pode afetar significativamente as operações da organização e até mesmo sua capacidade de sobrevivência. Um exemplo de orientação sobre a aplicação do ASVS Nível 3 é fornecido abaixo. Um aplicativo atinge ASVS Nível 3 (ou Avançado) se ele se defende adequadamente contra vulnerabilidades de segurança de aplicativo avançado e também demonstra princípios de um bom design de segurança.

Um aplicativo no ASVS Nível 3 requer uma análise mais profunda da arquitetura, codificação e teste do que todos os outros níveis. Um aplicativo seguro é modularizado de uma forma significativa (para facilitar a resiliência, escalabilidade e, acima de tudo, camadas de segurança), e cada módulo (separado por conexão de rede e / ou instância física) cuida de suas próprias responsabilidades de segurança (defesa em profundidade), que precisam ser devidamente documentados. As responsabilidades incluem controles para garantir a confidencialidade (por exemplo, criptografia), integridade (por exemplo, transações, validação de entrada), disponibilidade (por exemplo, lidar com a carga normalmente), autenticação (incluindo entre sistemas), não repúdio, autorização e auditoria (registro).

## Aplicando ASVS na prática

Ameaças diferentes têm motivações diferentes. Alguns setores têm ativos de informação e tecnologia exclusivos e requisitos de conformidade regulatória específicos de domínio.

As organizações são fortemente encorajadas a examinar profundamente suas características de risco exclusivas com base na natureza de seus negócios e, com base nesse risco e nos requisitos de negócios, determinar o nível de ASVS apropriado.

## Como fazer referência aos requisitos ASVS

Cada requisito possui um identificador no formato <capítulo>. <Seção>. <Requisito> onde cada elemento é um número, por exemplo: 1.11.3.

* O valor <capítulo> corresponde ao capítulo de onde vem o requisito, por exemplo: todos os requisitos 1. #. # São do capítulo Arquitetura.
* O valor <seção> corresponde à seção dentro desse capítulo onde o requisito aparece, por exemplo: todos os requisitos 1.11. # Estão na seção Requisitos de Lógica de Negócios do capítulo Arquitetura.
* O valor <requirement> identifica o requisito específico dentro do capítulo e seção, por exemplo: 1.11.3 que a partir da versão 4.0.2 desta norma é:

Verifique se todos os fluxos de lógica de negócios de alto valor, incluindo autenticação, gerenciamento de sessão e controle de acesso são thread-safe e resistentes a condições de tempo de verificação e tempo de uso.

Os identificadores podem mudar entre as versões do padrão, portanto, é preferível que outros documentos, relatórios ou ferramentas usem o formato: v <versão> - <capítulo>. <Seção>. <Requisito>, onde: 'versão' é o ASVS tag de versão. Por exemplo: v4.0.2-1.11.3 seria entendido como significando especificamente o terceiro requisito na seção 'Requisitos de Arquitetura da Lógica de Negócios' do capítulo 'Arquitetura' da versão 4.0.2. (Isso pode ser resumido como v <version> - <requirement\_identifier>.)

Nota: O v que precede a parte da versão deve ser minúsculo.

Se os identificadores forem usados ​​sem incluir o elemento v <version>, eles devem ser considerados como referindo-se ao conteúdo mais recente do Application Security Verification Standard. Obviamente, conforme o padrão cresce e muda, isso se torna problemático, e é por isso que os escritores ou desenvolvedores devem incluir o elemento de versão.

As listas de requisitos ASVS são disponibilizadas em CSV, JSON e outros formatos que podem ser úteis para referência ou uso programático.

# Avaliação e Certificação

## A posição do OWASP sobre as certificações e marcas de confiança ASVS

OWASP, como uma organização sem fins lucrativos neutra em relação a fornecedores, atualmente não certifica nenhum fornecedor, verificador ou software.

Todas essas afirmações de garantia, marcas de confiança ou certificações não são oficialmente examinadas, registradas ou certificadas pelo OWASP, portanto, uma organização que se baseia em tal visão precisa ser cautelosa quanto à confiança depositada em qualquer terceiro ou marca de confiança que reivindique a certificação ASVS.

Isso não deve inibir as organizações de oferecer tais serviços de garantia, desde que não reivindiquem a certificação oficial OWASP.

## Orientação para Organizações Certificadoras

O Application Security Verification Standard pode ser usado como uma verificação de livro aberto do aplicativo, incluindo acesso aberto e irrestrito aos principais recursos, como arquitetos e desenvolvedores, documentação do projeto, código-fonte, acesso autenticado a sistemas de teste (incluindo acesso a uma ou mais contas em cada função), particularmente para verificações L2 e L3.

Historicamente, os testes de penetração e as revisões de código seguro incluíram problemas “por exceção” - ou seja, apenas os testes que falharam aparecem no relatório final. Uma organização certificadora deve incluir em qualquer relatório o escopo da verificação (especialmente se um componente-chave estiver fora do escopo, como autenticação SSO), um resumo dos resultados da verificação, incluindo testes aprovados e reprovados, com indicações claras de como resolver o testes falharam.

Certos requisitos de verificação podem não ser aplicáveis ​​ao aplicativo em teste. Por exemplo, se você fornecer uma API de camada de serviço sem estado sem uma implementação de cliente para seus clientes, muitos dos requisitos no Gerenciamento de Sessão V3 não são diretamente aplicáveis. Em tais casos, uma organização certificadora ainda pode alegar conformidade total com o ASVS, mas deve indicar claramente em qualquer relatório uma razão para a não aplicabilidade de tais requisitos de verificação excluídos.

Manter papéis de trabalho detalhados, capturas de tela ou filmes, scripts para explorar um problema de maneira confiável e repetida e registros eletrônicos de testes, como a interceptação de logs de proxy e notas associadas, como uma lista de limpeza, é considerada uma prática padrão da indústria e pode ser realmente útil como provas das descobertas para os desenvolvedores mais duvidosos. Não é suficiente simplesmente executar uma ferramenta e relatar as falhas; isso não fornece (de forma alguma) evidência suficiente de que todos os problemas em um nível de certificação foram testados e testados exaustivamente. Em caso de disputa, deve haver evidência de garantia suficiente para demonstrar que cada um dos requisitos verificados foi realmente testado.

### Método de Teste

As organizações certificadoras são livres para escolher o (s) método (s) de teste apropriado (s), mas devem indicá-los em um relatório.

Dependendo da aplicação em teste e do requisito de verificação, diferentes métodos de teste podem ser usados ​​para obter confiança semelhante nos resultados. Por exemplo, a validação da eficácia dos mecanismos de verificação de entrada de um aplicativo pode ser analisada com um teste de penetração manual ou por meio de análises de código-fonte.

#### O papel das ferramentas de teste de segurança automatizadas

O uso de ferramentas de teste de penetração automatizadas é incentivado para fornecer o máximo de cobertura possível.

Não é possível concluir totalmente a verificação ASVS usando apenas ferramentas de teste de penetração automatizadas. Embora a grande maioria dos requisitos em L1 possa ser realizada por meio de testes automatizados, a maioria geral dos requisitos não são passíveis de testes de penetração automatizados.

Observe que os limites entre o teste automatizado e o manual se tornaram indistintos à medida que o setor de segurança de aplicativos amadurece. As ferramentas automatizadas costumam ser ajustadas manualmente por especialistas e os testadores manuais costumam aproveitar uma ampla variedade de ferramentas automatizadas.

#### O papel do teste de penetração

Na versão 4.0, decidimos tornar a penetração L1 completamente testável sem acesso ao código-fonte, documentação ou desenvolvedores. Dois itens de registro, que são necessários para cumprir o OWASP Top 10 2017 A10, exigirão entrevistas, capturas de tela ou outra coleta de evidências, assim como fazem no OWASP Top 10 2017. No entanto, testar sem acesso às informações necessárias não é um método ideal de verificação de segurança, pois perde a possibilidade de revisar a fonte, identificar ameaças e controles ausentes e realizar um teste muito mais completo em um período de tempo mais curto.

Sempre que possível, o acesso a desenvolvedores, documentação, código e acesso a um aplicativo de teste com dados de não produção é necessário ao realizar uma avaliação L2 ou L3. Os testes de penetração feitos nesses níveis requerem esse nível de acesso, que chamamos de "análises híbridas" ou "testes de penetração híbridos".

## Outros usos para o ASVS

Além de ser usado para avaliar a segurança de um aplicativo, identificamos vários outros usos potenciais para o ASVS.

### Como orientação detalhada da arquitetura de segurança

Um dos usos mais comuns do Application Security Verification Standard é como recurso para arquitetos de segurança. A Sherwood Applied Business Security Architecture (SABSA) está faltando uma grande quantidade de informações que são necessárias para concluir uma revisão completa da arquitetura de segurança do aplicativo. O ASVS pode ser usado para preencher essas lacunas, permitindo que os arquitetos de segurança escolham os melhores controles para problemas comuns, como padrões de proteção de dados e estratégias de validação de entrada.

### Como uma substituição para listas de verificação de codificação segura disponíveis no mercado

Muitas organizações podem se beneficiar da adoção do ASVS, escolhendo um dos três níveis ou bifurcando o ASVS e alterando o que é necessário para cada nível de risco do aplicativo de uma maneira específica do domínio. Encorajamos esse tipo de bifurcação, desde que a rastreabilidade seja mantida, de forma que, se um aplicativo tiver passado pelo requisito 4.1, isso signifique o mesmo para cópias bifurcadas e padrão conforme ele evolui.

### Como um guia para unidades automatizadas e testes de integração

O ASVS foi projetado para ser altamente testável, com exceção dos requisitos arquitetônicos e de código malicioso. Ao construir testes de unidade e integração que testam casos específicos e relevantes de difusão e abuso, o aplicativo torna-se quase autoverificável a cada construção. Por exemplo, testes adicionais podem ser criados para o conjunto de testes para um controlador de login, testando o parâmetro de nome de usuário para nomes de usuário padrão comuns, enumeração de conta, força bruta, injeção de LDAP e SQL e XSS. Da mesma forma, um teste no parâmetro de senha deve incluir senhas comuns, comprimento de senha, injeção de byte nulo, remoção do parâmetro, XSS e muito mais.

### Para treinamento de desenvolvimento seguro

O ASVS também pode ser usado para definir características de software seguro. Muitos cursos de “codificação segura” são simplesmente cursos de hacking ético com uma leve mancha de dicas de codificação. Isso pode não necessariamente ajudar os desenvolvedores a escrever um código mais seguro. Em vez disso, os cursos de desenvolvimento seguro podem usar o ASVS com um forte foco nos controles proativos encontrados no ASVS, em vez das 10 principais coisas negativas a não fazer.

### Como um driver para segurança ágil de aplicativos

O ASVS pode ser usado em um processo de desenvolvimento ágil como uma estrutura para definir tarefas específicas que precisam ser implementadas pela equipe para ter um produto seguro. Uma abordagem pode ser: Começando com o Nível 1, verifique o aplicativo ou sistema específico de acordo com os requisitos do ASVS para o nível especificado, encontre quais controles estão faltando e aumente tíquetes / tarefas específicos no backlog. Isso ajuda na priorização de tarefas específicas (ou preparação) e torna a segurança visível no processo ágil. Isso também pode ser usado para priorizar tarefas de auditoria e revisão na organização, onde um requisito ASVS específico pode ser um motivador para revisão, refatoração ou auditoria para um membro da equipe específico e visível como "dívida" no backlog que precisa ser feito eventualmente .

### Como uma estrutura para orientar a aquisição de software seguro

ASVS é uma ótima estrutura para ajudar na aquisição segura de software ou aquisição de serviços de desenvolvimento personalizados. O comprador pode simplesmente definir um requisito de que o software que deseja adquirir deve ser desenvolvido em ASVS nível X e solicitar que o vendedor prove que o software atende ASVS nível X. Isso funciona bem quando combinado com o Anexo de Contrato de Software Seguro OWASP

# V1: Requisitos de arquitetura, design e modelagem de ameaças

## Objetivo de Controle

A arquitetura de segurança quase se tornou uma arte perdida em muitas organizações. Os dias do arquiteto corporativo já passaram na era do DevSecOps. O campo de segurança de aplicativos deve acompanhar e adotar princípios de segurança ágeis enquanto reintroduz os principais princípios de arquitetura de segurança aos profissionais de software. Arquitetura não é uma implementação, mas uma maneira de pensar sobre um problema que tem potencialmente muitas respostas diferentes e nenhuma única resposta "correta". Freqüentemente, a segurança é vista como inflexível e exigindo que os desenvolvedores corrijam o código de uma maneira particular, quando os desenvolvedores podem conhecer uma maneira muito melhor de resolver o problema. Não existe uma solução única e simples para a arquitetura, e fingir o contrário é um desserviço ao campo da engenharia de software.

É provável que uma implementação específica de um aplicativo da web seja revisada continuamente ao longo de sua vida útil, mas a arquitetura geral provavelmente raramente mudará, mas evoluirá lentamente. A arquitetura de segurança é idêntica - precisamos de autenticação hoje, vamos exigir autenticação amanhã e vamos precisar dela daqui a cinco anos. Se tomarmos decisões corretas hoje, podemos economizar muito esforço, tempo e dinheiro se selecionarmos e reutilizarmos soluções compatíveis com arquitetura. Por exemplo, há uma década, a autenticação multifator raramente era implementada.

Se os desenvolvedores tivessem investido em um único modelo de provedor de identidade seguro, como identidade federada SAML, o provedor de identidade poderia ser atualizado para incorporar novos requisitos, como conformidade com NIST 800-63, sem alterar as interfaces do aplicativo original. Se muitos aplicativos compartilham a mesma arquitetura de segurança e, portanto, o mesmo componente, todos se beneficiam dessa atualização de uma vez. No entanto, o SAML nem sempre será a melhor ou mais adequada solução de autenticação - pode ser necessário trocá-lo por outras soluções conforme os requisitos mudam. Mudanças como essa são complicadas e caras a ponto de exigir uma reescrita completa ou totalmente impossíveis sem uma arquitetura de segurança.

Neste capítulo, o ASVS cobre os principais aspectos de qualquer arquitetura de segurança sólida: disponibilidade, confidencialidade, integridade de processamento, não repúdio e privacidade. Cada um desses princípios de segurança deve ser integrado e inato a todos os aplicativos. É fundamental "mudar para a esquerda", começando com a capacitação do desenvolvedor com listas de verificação de codificação segura, orientação e treinamento, codificação e teste, construção, implantação, configuração e operações, e terminando com testes independentes de acompanhamento para garantir que todos os controles de segurança estão presentes e funcionais. A última etapa costumava ser tudo o que fazíamos como setor, mas isso não é mais suficiente quando os desenvolvedores colocam o código em produção dezenas ou centenas de vezes por dia. Os profissionais de segurança de aplicativos devem acompanhar as técnicas ágeis,

## Requisitos de ciclo de vida de desenvolvimento de software seguro V1.1

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.1.1** | Verifique o uso de um ciclo de vida de desenvolvimento de software seguro que trate da segurança em todos os estágios de desenvolvimento. ([C1](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ |  |
| **1.1.2** | Verifique o uso de modelagem de ameaças para cada mudança de design ou planejamento de sprint para identificar ameaças, planejar contra-medidas, facilitar respostas de risco apropriadas e orientar os testes de segurança. |  | ✓ | ✓ | 1053 |
| **1.1.3** | Verifique se todas as histórias de usuário e recursos contêm restrições de segurança funcional, como "Como um usuário, devo ser capaz de visualizar e editar meu perfil. Não devo ser capaz de visualizar ou editar o perfil de outra pessoa" |  | ✓ | ✓ | 1110 |
| **1.1.4** | Verifique a documentação e a justificativa de todos os limites de confiança, componentes e fluxos de dados significativos do aplicativo. |  | ✓ | ✓ | 1059 |
| **1.1.5** | Verifique a definição e a análise de segurança da arquitetura de alto nível do aplicativo e de todos os serviços remotos conectados. ([C1](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 1059 |
| **1.1.6** | Verifique a implementação de controles de segurança centralizados, simples (economia de design), verificados, seguros e reutilizáveis ​​para evitar controles duplicados, ausentes, ineficazes ou inseguros. ([C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 637 |
| **1.1.7** | Verifique a disponibilidade de uma lista de verificação de codificação segura, requisitos de segurança, diretriz ou política para todos os desenvolvedores e testadores. |  | ✓ | ✓ | 637 |

## Requisitos de arquitetura de autenticação V1.2

Ao projetar a autenticação, não importa se você tem uma autenticação multifator habilitada por hardware forte, se um invasor puder redefinir uma conta ligando para um call center e respondendo a perguntas comumente conhecidas. Ao verificar a identidade, todos os caminhos de autenticação devem ter a mesma força.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.2.1** | Verifique o uso de contas de sistema operacional exclusivas ou especiais de baixo privilégio para todos os componentes de aplicativos, serviços e servidores. ([C3](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 250 |
| **1.2.2** | Verifique se as comunicações entre os componentes do aplicativo, incluindo APIs, middleware e camadas de dados, são autenticadas. Os componentes devem ter o mínimo de privilégios necessários. ([C3](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 306 |
| **1.2.3** | Verifique se o aplicativo usa um único mecanismo de autenticação verificado que é conhecido por ser seguro, pode ser estendido para incluir autenticação forte e tem registro e monitoramento suficientes para detectar abusos ou violações de conta. |  | ✓ | ✓ | 306 |
| **1.2.4** | Verifique se todos os caminhos de autenticação e APIs de gerenciamento de identidade implementam força de controle de segurança de autenticação consistente, de forma que não haja alternativas mais fracas de acordo com o risco do aplicativo. |  | ✓ | ✓ | 306 |

## Requisitos de arquitetura de gerenciamento de sessão V1.3

Este é um espaço reservado para requisitos arquitetônicos futuros.

## Requisitos de arquitetura de controle de acesso V1.4

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.4.1** | Verifique se os pontos de aplicação confiáveis, como gateways de controle de acesso, servidores e funções sem servidor, aplicam os controles de acesso. Nunca aplique controles de acesso no cliente. |  | ✓ | ✓ | 602 |
| **1.4.2** | Verifique se a solução de controle de acesso escolhida é flexível o suficiente para atender às necessidades do aplicativo. |  | ✓ | ✓ | 284 |
| **1.4.3** | Verifique a aplicação do princípio do menor privilégio em funções, arquivos de dados, URLs, controladores, serviços e outros recursos. Isso implica proteção contra falsificação e elevação de privilégio. |  | ✓ | ✓ | 272 |
| **1.4.4** | Verifique se o aplicativo usa um mecanismo de controle de acesso único e bem verificado para acessar dados e recursos protegidos. Todas as solicitações devem passar por esse único mecanismo para evitar copiar e colar ou caminhos alternativos inseguros. ([C7](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 284 |
| **1.4.5** | Verifique se o controle de acesso baseado em atributo ou recurso é usado por meio do qual o código verifica a autorização do usuário para um recurso / item de dados em vez de apenas sua função. As permissões ainda devem ser alocadas usando funções. ([C7](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 275 |

## Requisitos de arquitetura de entrada e saída V1.5

No 4.0, mudamos o termo "lado do servidor" como um termo de limite de confiança carregado. O limite de confiança ainda é preocupante - a tomada de decisões em navegadores ou dispositivos clientes não confiáveis ​​pode ser evitada. No entanto, nas implantações arquitetônicas tradicionais de hoje, o ponto de imposição de confiança mudou drasticamente. Portanto, onde o termo "camada de serviço confiável" é usado no ASVS, queremos dizer qualquer ponto de aplicação confiável, independentemente da localização, como um microsserviço, API sem servidor, lado do servidor, uma API confiável em um dispositivo cliente com inicialização segura , parceiro ou APIs externas e assim por diante.

O termo "cliente não confiável" aqui se refere às tecnologias do lado do cliente que processam a camada de apresentação, comumente chamadas de tecnologias de 'front-end'. O termo "serialização" aqui não se refere apenas ao envio de dados pela rede como uma matriz de valores ou obter e ler uma estrutura JSON, mas também transmitir objetos complexos que podem conter lógica.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.5.1** | Verifique se os requisitos de entrada e saída definem claramente como lidar e processar dados com base no tipo, conteúdo e leis e regulamentos aplicáveis, além de outras políticas de conformidade. |  | ✓ | ✓ | 1029 |
| **1.5.2** | Verifique se a serialização não é usada ao se comunicar com clientes não confiáveis. Se isso não for possível, certifique-se de que os controles de integridade adequados (e possivelmente a criptografia se dados confidenciais forem enviados) sejam aplicados para evitar ataques de desserialização, incluindo injeção de objeto. |  | ✓ | ✓ | 502 |
| **1.5.3** | Verifique se a validação de entrada é aplicada em uma camada de serviço confiável. ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 602 |
| **1.5.4** | Verifique se a codificação de saída ocorre perto ou pelo intérprete para o qual se destina. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 116 |

## Requisitos de arquitetura criptográfica V1.6

Os aplicativos precisam ser projetados com uma arquitetura criptográfica forte para proteger os ativos de dados de acordo com sua classificação. Criptografar tudo é um desperdício, não criptografar nada é uma negligência legal. Um equilíbrio deve ser alcançado, geralmente durante o projeto arquitetônico ou de alto nível, sprints de design ou picos arquitetônicos. Projetar a criptografia conforme você avança ou adaptá-la inevitavelmente custará muito mais para implementar com segurança do que simplesmente incorporá-la desde o início.

Os requisitos de arquitetura são intrínsecos a toda a base de código e, portanto, difíceis de unir ou integrar o teste. Os requisitos de arquitetura exigem consideração nos padrões de codificação, em toda a fase de codificação, e devem ser revisados ​​durante a arquitetura de segurança, revisões de pares ou de código ou retrospectivas.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.6.1** | Verifique se há uma política explícita para gerenciamento de chaves criptográficas e se o ciclo de vida de uma chave criptográfica segue um padrão de gerenciamento de chaves, como NIST SP 800-57. |  | ✓ | ✓ | 320 |
| **1.6.2** | Verifique se os consumidores de serviços criptográficos protegem o material da chave e outros segredos usando cofres de chaves ou alternativas baseadas em API. |  | ✓ | ✓ | 320 |
| **1.6.3** | Verifique se todas as chaves e senhas são substituíveis e fazem parte de um processo bem definido para criptografar novamente os dados confidenciais. |  | ✓ | ✓ | 320 |
| **1.6.4** | Verifique se a arquitetura trata os segredos do lado do cliente - como chaves simétricas, senhas ou tokens de API - como inseguros e nunca os usa para proteger ou acessar dados confidenciais. |  | ✓ | ✓ | 320 |

## Erros V1.7, requisitos arquitetônicos de registro e auditoria

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.7.1** | Verifique se um formato e abordagem de registro comuns são usados ​​em todo o sistema. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 1009 |
| **1.7.2** | Verifique se os logs são transmitidos com segurança para um sistema preferencialmente remoto para análise, detecção, alerta e escalonamento. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ |  |

## Requisitos de arquitetura de privacidade e proteção de dados V1.8

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.8.1** | Verifique se todos os dados confidenciais são identificados e classificados em níveis de proteção. |  | ✓ | ✓ |  |
| **1.8.2** | Verifique se todos os níveis de proteção têm um conjunto associado de requisitos de proteção, como requisitos de criptografia, requisitos de integridade, retenção, privacidade e outros requisitos de confidencialidade, e que estes são aplicados na arquitetura. |  | ✓ | ✓ |  |

## Requisitos de arquitetura de comunicações V1.9

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.9.1** | Verifique se o aplicativo criptografa as comunicações entre os componentes, principalmente quando esses componentes estão em diferentes contêineres, sistemas, sites ou provedores de nuvem. ([C3](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 319 |
| **1.9.2** | Verifique se os componentes do aplicativo verificam a autenticidade de cada lado em um link de comunicação para evitar ataques de intermediários. Por exemplo, os componentes do aplicativo devem validar certificados e cadeias TLS. |  | ✓ | ✓ | 295 |

## Requisitos de arquitetura de software malicioso V1.10

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1,10.1** | Verifique se um sistema de controle de código-fonte está em uso, com procedimentos para garantir que os check-ins sejam acompanhados por ocorrências ou tíquetes de alteração. O sistema de controle do código-fonte deve ter controle de acesso e usuários identificáveis ​​para permitir a rastreabilidade de quaisquer alterações. |  | ✓ | ✓ | 284 |

## Requisitos de arquitetura de lógica de negócios V1.11

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.11.1** | Verifique a definição e a documentação de todos os componentes do aplicativo em termos das funções de negócios ou de segurança que eles fornecem. |  | ✓ | ✓ | 1059 |
| **1.11.2** | Verifique se todos os fluxos de lógica de negócios de alto valor, incluindo autenticação, gerenciamento de sessão e controle de acesso, não compartilham o estado não sincronizado. |  | ✓ | ✓ | 362 |
| **1.11.3** | Verifique se todos os fluxos de lógica de negócios de alto valor, incluindo autenticação, gerenciamento de sessão e controle de acesso são thread-safe e resistentes a condições de tempo de verificação e tempo de uso. |  |  | ✓ | 367 |

## Requisitos de arquitetura para upload seguro de arquivo V1.12

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.12.1** | Verifique se os arquivos carregados pelo usuário estão armazenados fora da raiz da web. |  | ✓ | ✓ | 552 |
| **1.12.2** | Verifique se os arquivos carregados pelo usuário - se necessário para serem exibidos ou baixados do aplicativo - são servidos por downloads de fluxo de octeto ou de um domínio não relacionado, como um depósito de armazenamento de arquivo em nuvem. Implemente uma Política de Segurança de Conteúdo (CSP) adequada para reduzir o risco de vetores XSS ou outros ataques do arquivo carregado. |  | ✓ | ✓ | 646 |

## Requisitos de arquitetura da API V1.13

Este é um espaço reservado para requisitos arquitetônicos futuros.

## Requisitos de arquitetura de configuração V1.14

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **1.14.1** | Verifique a segregação de componentes de diferentes níveis de confiança por meio de controles de segurança bem definidos, regras de firewall, gateways de API, proxies reversos, grupos de segurança baseados em nuvem ou mecanismos semelhantes. |  | ✓ | ✓ | 923 |
| **1.14.2** | Verifique se as assinaturas binárias, as conexões confiáveis ​​e os terminais verificados são usados ​​para implantar binários em dispositivos remotos. |  | ✓ | ✓ | 494 |
| **1.14.3** | Verifique se o pipeline de construção avisa sobre componentes desatualizados ou inseguros e executa as ações apropriadas. |  | ✓ | ✓ | 1104 |
| **1.14.4** | Verifique se o pipeline de construção contém uma etapa de construção para construir e verificar automaticamente a implementação segura do aplicativo, especialmente se a infraestrutura do aplicativo for definida por software, como scripts de construção de ambiente de nuvem. |  | ✓ | ✓ |  |
| **1,14,5** | Verifique se as implementações de aplicativos fornecem proteção, contêineres e / ou isolam adequadamente no nível da rede para atrasar e impedir que invasores ataquem outros aplicativos, especialmente quando estão executando ações confidenciais ou perigosas, como desserialização. ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 265 |
| **1.14.6** | Verifique se o aplicativo não usa tecnologias não suportadas, inseguras ou obsoletas do lado do cliente, como plug-ins NSAPI, Flash, Shockwave, ActiveX, Silverlight, NACL ou miniaplicativos Java do lado do cliente. |  | ✓ | ✓ | 477 |

## Referências

Para obter mais informações, consulte também:

* [Folha de referências da modelagem de ameaças OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html)
* [Folha de dicas de análise de superfície de ataque OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Attack_Surface_Analysis_Cheat_Sheet.html)
* [Modelagem de ameaças OWASP](https://owasp.org/www-community/Application_Threat_Modeling)
* [Projeto de modelo de maturidade do OWASP Software Assurance](https://owasp.org/www-project-samm/)
* [Microsoft SDL](https://www.microsoft.com/en-us/sdl/)
* [NIST SP 800-57](https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-5/final)

# V2: Requisitos de verificação de autenticação

## Objetivo de Controle

Autenticação é o ato de estabelecer ou confirmar alguém (ou algo) como autêntico e que as declarações feitas por uma pessoa ou sobre um dispositivo são corretas, resistentes à falsificação de identidade e evitam a recuperação ou interceptação de senhas.

Quando o ASVS foi lançado pela primeira vez, nome de usuário + senha era a forma mais comum de autenticação fora dos sistemas de alta segurança. A autenticação multifator (MFA) era comumente aceita nos círculos de segurança, mas raramente exigida em outros lugares. À medida que o número de violações de senha aumentava, a ideia de que os nomes de usuário são de alguma forma confidenciais e as senhas desconhecidas tornou muitos controles de segurança insustentáveis. Por exemplo, NIST 800-63 considera nomes de usuário e autenticação baseada em conhecimento (KBA) como informações públicas, SMS e notificações por e-mail como[tipos de autenticador "restrito"](https://pages.nist.gov/800-63-FAQ/#q-b1)e senhas como pré-violadas. Essa realidade torna os autenticadores baseados em conhecimento, recuperação de SMS e e-mail, histórico de senha, complexidade e controles de rotação inúteis. Esses controles sempre foram menos do que úteis, muitas vezes forçando os usuários a criarem senhas fracas a cada poucos meses, mas com o lançamento de mais de 5 bilhões de violações de nome de usuário e senha, é hora de seguir em frente.

De todas as seções do ASVS, os capítulos de autenticação e gerenciamento de sessão foram os que mais mudaram. A adoção de práticas de liderança eficazes e baseadas em evidências será um desafio para muitos, e isso é perfeitamente normal. Temos que iniciar a transição para um futuro pós-senha agora.

## NIST 800-63 - padrão de autenticação moderno baseado em evidências

[NIST 800-63b](https://pages.nist.gov/800-63-3/sp800-63b.html)é um padrão moderno baseado em evidências e representa o melhor conselho disponível, independentemente da aplicabilidade. O padrão é útil para todas as organizações em todo o mundo, mas é particularmente relevante para agências dos Estados Unidos e aquelas que lidam com agências dos Estados Unidos.

A terminologia do NIST 800-63 pode ser um pouco confusa no início, especialmente se você está acostumado apenas com a autenticação de nome de usuário + senha. Avanços na autenticação moderna são necessários, portanto, temos que introduzir uma terminologia que se tornará comum no futuro, mas entendemos a dificuldade de compreensão até que a indústria aceite esses novos termos. Fornecemos um glossário no final deste capítulo para ajudá-lo. Reformulamos muitos requisitos para satisfazer a intenção do requisito, em vez da letra do requisito. Por exemplo, o ASVS usa o termo "senha" quando o NIST usa "segredo memorizado" em todo este padrão.

ASVS V2 Authentication, V3 Session Management e, em menor grau, V4 Access Controls foram adaptados para ser um subconjunto compatível de controles selecionados do NIST 800-63b, focados em ameaças comuns e fraquezas de autenticação comumente exploradas. Onde for necessária a conformidade total com o NIST 800-63, consulte o NIST 800-63.

### Selecionando um Nível NIST AAL apropriado

O Application Security Verification Standard tentou mapear os requisitos ASVS L1 para NIST AAL1, L2 para AAL2 e L3 para AAL3. No entanto, a abordagem do ASVS Nível 1 como controles "essenciais" pode não ser necessariamente o nível AAL correto para verificar um aplicativo ou API. Por exemplo, se o aplicativo é um aplicativo de Nível 3 ou tem requisitos regulatórios para ser AAL3, o Nível 3 deve ser escolhido nas Seções V2 e V3 Gerenciamento de Sessão. A escolha do Nível de Asserção de Autenticação (AAL) compatível com NIST deve ser realizada de acordo com as diretrizes NIST 800-63b conforme estabelecido em Seleção de AAL em[NIST 800-63b Seção 6.2](https://pages.nist.gov/800-63-3/sp800-63-3.html#AAL_CYOA).

## Lenda

Os aplicativos sempre podem exceder os requisitos do nível atual, especialmente se a autenticação moderna estiver no roteiro de um aplicativo. Anteriormente, o ASVS exigia um MFA obrigatório. O NIST não exige MFA obrigatório. Portanto, usamos uma designação opcional neste capítulo para indicar onde o ASVS incentiva, mas não requer um controle. As seguintes chaves são usadas em todo este padrão:

| marca | Descrição |
| --- | --- |
|  | Não requerido |
| o | Recomendado, mas não obrigatório |
| ✓ | Requeridos |

## Requisitos de segurança de senha V2.1

As senhas, chamadas de "segredos memorizados" pelo NIST 800-63, incluem senhas, PINs, padrões de desbloqueio, escolha o gatinho correto ou outro elemento de imagem e frases secretas. Eles são geralmente considerados "algo que você sabe" e frequentemente usados ​​como autenticadores de fator único. Existem desafios significativos para o uso contínuo da autenticação de fator único, incluindo bilhões de nomes de usuário e senhas válidos divulgados na Internet, senhas padrão ou fracas, tabelas de arco-íris e dicionários ordenados das senhas mais comuns.

Os aplicativos devem encorajar fortemente os usuários a se inscreverem na autenticação multifator e permitir que os usuários reutilizem tokens que já possuem, como tokens FIDO ou U2F, ou vincular a um provedor de serviços de credencial que forneça autenticação multifator.

Provedores de serviços de credencial (CSPs) fornecem identidade federada para usuários. Os usuários geralmente terão mais de uma identidade com vários CSPs, como uma identidade corporativa usando Azure AD, Okta, Ping Identity ou Google, ou identidade de consumidor usando Facebook, Twitter, Google ou WeChat, para citar apenas algumas alternativas comuns. Esta lista não é um endosso dessas empresas ou serviços, mas simplesmente um incentivo para os desenvolvedores considerarem a realidade de que muitos usuários têm muitas identidades estabelecidas. As organizações devem considerar a integração com identidades de usuário existentes, de acordo com o perfil de risco da força da prova de identidade do CSP. Por exemplo, é improvável que uma organização governamental aceite uma identidade de mídia social como um login para sistemas confidenciais, pois é fácil criar identidades falsas ou jogar fora,

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.1.1** | Verifique se as senhas definidas pelo usuário têm pelo menos 12 caracteres (depois que vários espaços são combinados). ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.2** | Verifique se as senhas de 64 caracteres ou mais são permitidas, mas não podem ter mais de 128 caracteres. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.3** | Verifique se o truncamento de senha não foi executado. No entanto, vários espaços consecutivos podem ser substituídos por um único espaço. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.4** | Verifique se qualquer caractere Unicode imprimível, incluindo caracteres de idioma neutro, como espaços e emojis, são permitidos nas senhas. | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.5** | Verifique se os usuários podem alterar suas senhas. | ✓ | ✓ | ✓ | 620 | 5.1.1.2 |
| **2.1.6** | Verifique se a funcionalidade de alteração de senha requer a senha atual e a nova do usuário. | ✓ | ✓ | ✓ | 620 | 5.1.1.2 |
| **2.1.7** | Verifique se as senhas enviadas durante o registro da conta, login e alteração de senha são comparadas a um conjunto de senhas violadas localmente (como as 1.000 ou 10.000 senhas mais comuns que correspondem à política de senha do sistema) ou usando uma API externa. Se estiver usando uma API, uma prova de conhecimento zero ou outro mecanismo deve ser usado para garantir que a senha em texto simples não seja enviada ou usada na verificação do status de violação da senha. Se a senha for violada, o aplicativo deve exigir que o usuário defina uma nova senha não violada. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.8** | Verifique se um medidor de força da senha é fornecido para ajudar os usuários a definir uma senha mais forte. | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.9** | Verifique se não há regras de composição de senha que limitem o tipo de caracteres permitidos. Não deve haver nenhum requisito para letras maiúsculas ou minúsculas, números ou caracteres especiais. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.10** | Verifique se não há rotação periódica de credenciais ou requisitos de histórico de senha. | ✓ | ✓ | ✓ | 263 | 5.1.1.2 |
| **2.1.11** | Verifique se a funcionalidade "colar", os ajudantes de senha do navegador e os gerenciadores de senha externos são permitidos. | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |
| **2.1.12** | Verifique se o usuário pode escolher visualizar temporariamente a senha mascarada inteira ou visualizar temporariamente o último caractere digitado da senha em plataformas que não possuem essa funcionalidade incorporada. | ✓ | ✓ | ✓ | 521 | 5.1.1.2 |

Nota: O objetivo de permitir que o usuário visualize sua senha ou veja o último caractere temporariamente é melhorar a usabilidade da entrada de credenciais, particularmente em torno do uso de senhas, frases de acesso e gerenciadores de senhas mais longos. Outro motivo para incluir o requisito é impedir ou impedir relatórios de teste que exijam desnecessariamente que as organizações substituam o comportamento do campo de senha da plataforma integrada para remover essa experiência de segurança moderna e amigável.

## Requisitos gerais do autenticador V2.2

A agilidade do autenticador é essencial para aplicativos preparados para o futuro. Refatore os verificadores de aplicativos para permitir autenticadores adicionais de acordo com as preferências do usuário, bem como permitir a desativação de autenticadores obsoletos ou inseguros de maneira ordenada.

NIST considera e-mail e SMS como [tipos de autenticador "restrito"](https://pages.nist.gov/800-63-FAQ/#q-b1), e provavelmente serão removidos do NIST 800-63 e, portanto, do ASVS em algum momento no futuro. Os aplicativos devem planejar um roteiro que não exija o uso de e-mail ou SMS.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.2.1** | Verifique se os controles anti-automação são eficazes na mitigação de testes de credenciais violados, força bruta e ataques de bloqueio de contas. Esses controles incluem o bloqueio das senhas violadas mais comuns, bloqueios flexíveis, limitação de taxa, CAPTCHA, atrasos cada vez maiores entre as tentativas, restrições de endereço IP ou restrições baseadas em risco, como localização, primeiro login em um dispositivo, tentativas recentes de desbloquear a conta, ou similar. Verifique se não é possível mais de 100 tentativas malsucedidas por hora em uma única conta. | ✓ | ✓ | ✓ | 307 | 5.2.2 / 5.1.1.2 / 5.1.4.2 / 5.1.5.2 |
| **2.2.2** | Verifique se o uso de autenticadores fracos (como SMS e e-mail) é limitado à verificação secundária e aprovação da transação e não como um substituto para métodos de autenticação mais seguros. Verifique se métodos mais fortes são oferecidos antes dos métodos fracos, se os usuários estão cientes dos riscos ou se as medidas adequadas estão em vigor para limitar os riscos de comprometimento da conta. | ✓ | ✓ | ✓ | 304 | 5,10 |
| **2.2.3** | Verifique se as notificações seguras são enviadas aos usuários após atualizações nos detalhes de autenticação, como redefinições de credenciais, alterações de e-mail ou endereço, login de locais desconhecidos ou arriscados. O uso de notificações push - em vez de SMS ou e-mail - é preferível, mas na ausência de notificações push, SMS ou e-mail é aceitável, desde que nenhuma informação sensível seja divulgada na notificação. | ✓ | ✓ | ✓ | 620 |  |
| **2.2.4** | Verifique a resistência à falsificação de identidade contra phishing, como o uso de autenticação multifator, dispositivos criptográficos com intenção (como chaves conectadas com um push para autenticação) ou, em níveis AAL mais altos, certificados do lado do cliente. |  |  | ✓ | 308 | 5.2.5 |
| **2.2.5** | Verifique se onde um provedor de serviços de credencial (CSP) e o aplicativo que verifica a autenticação estão separados, o TLS mutuamente autenticado está instalado entre os dois pontos de extremidade. |  |  | ✓ | 319 | 5.2.6 |
| **2.2.6** | Verifique a resistência à reprodução por meio do uso obrigatório de dispositivos de senha única (OTP), autenticadores criptográficos ou códigos de pesquisa. |  |  | ✓ | 308 | 5.2.8 |
| **2.2.7** | Verifique a intenção de autenticação exigindo a entrada de um token OTP ou ação iniciada pelo usuário, como um pressionamento de botão em uma chave de hardware FIDO. |  |  | ✓ | 308 | 5.2.9 |

## Requisitos de ciclo de vida do autenticador V2.3

Os autenticadores são senhas, tokens de software, tokens de hardware e dispositivos biométricos. O ciclo de vida dos autenticadores é crítico para a segurança de um aplicativo - se alguém pode autorregistrar uma conta sem nenhuma evidência de identidade, pode haver pouca confiança na afirmação da identidade. Para sites de mídia social como o Reddit, isso é perfeitamente normal. Para sistemas bancários, um foco maior no registro e emissão de credenciais e dispositivos é fundamental para a segurança do aplicativo.

Nota: As senhas não devem ter uma vida útil máxima ou estar sujeitas à rotação de senha. As senhas devem ser verificadas quanto à violação, e não substituídas regularmente.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.3.1** | Verifique se as senhas ou códigos de ativação iniciais gerados pelo sistema DEVEM ser gerados de forma aleatória e segura, DEVEM ter pelo menos 6 caracteres e PODEM conter letras e números e expiram após um curto período de tempo. Não se deve permitir que esses segredos iniciais se tornem a senha de longo prazo. | ✓ | ✓ | ✓ | 330 | 5.1.1.2 / A.3 |
| **2.3.2** | Verifique se o registro e o uso de dispositivos de autenticação fornecidos pelo assinante são suportados, como tokens U2F ou FIDO. |  | ✓ | ✓ | 308 | 6.1.3 |
| **2.3.3** | Verifique se as instruções de renovação são enviadas com tempo suficiente para renovar os autenticadores com limite de tempo. |  | ✓ | ✓ | 287 | 6.1.4 |

## Requisitos de armazenamento de credenciais V2.4

Arquitetos e desenvolvedores devem seguir esta seção ao construir ou refatorar código. Esta seção só pode ser totalmente verificada usando revisão do código-fonte ou por meio de unidade segura ou testes de integração. O teste de penetração não consegue identificar nenhum desses problemas.

A lista de funções de derivação de chave unilateral aprovadas é detalhada no NIST 800-63 B seção 5.1.1.2, e em [BSI Kryptographische Verfahren: Empfehlungen und Schlussellängen (2018)](https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102.pdf?__blob=publicationFile). O algoritmo nacional ou regional mais recente e os padrões de comprimento de chave podem ser escolhidos no lugar dessas escolhas.

Esta seção não pode ser testada quanto à penetração, portanto, os controles não são marcados como L1. No entanto, esta seção é de vital importância para a segurança das credenciais se elas forem roubadas, portanto, se for forçar o ASVS para uma arquitetura ou diretriz de codificação ou lista de verificação de revisão de código-fonte, coloque esses controles de volta em L1 em ​​sua versão privada.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.4.1** | Verifique se as senhas estão armazenadas em um formato resistente a ataques offline. As senhas DEVEM ser salgadas e hash usando uma derivação de chave unilateral aprovada ou função hash de senha. As funções de derivação de chave e hashing de senha usam uma senha, um salt e um fator de custo como entradas ao gerar um hash de senha. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 916 | 5.1.1.2 |
| **2.4.2** | Verifique se o salt tem pelo menos 32 bits de comprimento e seja escolhido arbitrariamente para minimizar as colisões de valores de sal entre os hashes armazenados. Para cada credencial, um valor de sal exclusivo e o hash resultante DEVERÃO ser armazenados. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 916 | 5.1.1.2 |
| **2.4.3** | Verifique se se PBKDF2 for usado, a contagem de iterações DEVE ser tão grande quanto o desempenho do servidor de verificação permitir, normalmente pelo menos 100.000 iterações. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 916 | 5.1.1.2 |
| **2.4.4** | Verifique se bcrypt é usado, o fator de trabalho DEVE ser tão grande quanto o desempenho do servidor de verificação permitir, normalmente pelo menos 13. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 916 | 5.1.1.2 |
| **2.4.5** | Verifique se uma iteração adicional de uma função de derivação de chave é executada, usando um valor salt que é secreto e conhecido apenas pelo verificador. Gere o valor de sal usando um gerador de bits aleatório aprovado [SP 800-90Ar1] e forneça pelo menos a força de segurança mínima especificada na última revisão de SP 800-131A. O valor salt secreto DEVERÁ ser armazenado separadamente das senhas com hash (por exemplo, em um dispositivo especializado como um módulo de segurança de hardware). |  | ✓ | ✓ | 916 | 5.1.1.2 |

Onde os padrões dos EUA são mencionados, um padrão regional ou local pode ser usado no lugar ou além do padrão dos EUA, conforme necessário.

## Requisitos de recuperação de credencial V2.5

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.5.1** | Verifique se uma ativação inicial gerada pelo sistema ou segredo de recuperação não foi enviada em texto não criptografado para o usuário. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 640 | 5.1.1.2 |
| **2.5.2** | Verifique se as dicas de senha ou autenticação baseada em conhecimento (as chamadas "perguntas secretas") não estão presentes. | ✓ | ✓ | ✓ | 640 | 5.1.1.2 |
| **2.5.3** | Verifique se a recuperação de credencial de senha não revela a senha atual de forma alguma. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 640 | 5.1.1.2 |
| **2.5.4** | Verifique se as contas compartilhadas ou padrão não estão presentes (por exemplo, "root", "admin" ou "sa"). | ✓ | ✓ | ✓ | 16 | 5.1.1.2 / A.3 |
| **2.5.5** | Verifique se um fator de autenticação é alterado ou substituído, se o usuário é notificado desse evento. | ✓ | ✓ | ✓ | 304 | 6.1.2.3 |
| **2.5.6** | Verifique a senha esquecida e outros caminhos de recuperação usam um mecanismo de recuperação seguro, como OTP baseado em tempo (TOTP) ou outro soft token, push móvel ou outro mecanismo de recuperação offline. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 640 | 5.1.1.2 |
| **2.5.7** | Verifique se se os fatores de autenticação multifator ou OTP forem perdidos, essa evidência de prova de identidade é executada no mesmo nível durante a inscrição. |  | ✓ | ✓ | 308 | 6.1.2.3 |

## V2.6 Requisitos do verificador de segredo de pesquisa

Os segredos de pesquisa são listas pré-geradas de códigos secretos, semelhantes aos Números de Autorização de Transação (TAN), códigos de recuperação de mídia social ou uma grade contendo um conjunto de valores aleatórios. Eles são distribuídos com segurança aos usuários. Esses códigos de pesquisa são usados ​​uma vez e, depois que todos são usados, a lista de segredos de pesquisa é descartada. Este tipo de autenticador é considerado "algo que você possui".

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.6.1** | Verifique se os segredos de pesquisa podem ser usados ​​apenas uma vez. |  | ✓ | ✓ | 308 | 5.1.2.2 |
| **2.6.2** | Verifique se os segredos de pesquisa têm aleatoriedade suficiente (112 bits de entropia) ou, se menos de 112 bits de entropia, salgados com um sal de 32 bits único e aleatório e hash com um hash unilateral aprovado. |  | ✓ | ✓ | 330 | 5.1.2.2 |
| **2.6.3** | Verifique se os segredos de pesquisa são resistentes a ataques offline, como valores previsíveis. |  | ✓ | ✓ | 310 | 5.1.2.2 |

## Requisitos do verificador fora de banda V2.7

No passado, um verificador fora da banda comum seria um e-mail ou SMS contendo um link de redefinição de senha. Os invasores usam esse mecanismo fraco para redefinir contas que eles ainda não controlam, como assumir o controle da conta de e-mail de uma pessoa e reutilizar quaisquer links de redefinição descobertos. Existem maneiras melhores de lidar com a verificação fora da banda.

Autenticadores fora de banda seguros são dispositivos físicos que podem se comunicar com o verificador por meio de um canal secundário seguro. Os exemplos incluem notificações push para dispositivos móveis. Este tipo de autenticador é considerado "algo que você tem". Quando um usuário deseja autenticar, o aplicativo de verificação envia uma mensagem para o autenticador fora de banda por meio de uma conexão com o autenticador direta ou indiretamente por meio de um serviço de terceiros. A mensagem contém um código de autenticação (normalmente um número aleatório de seis dígitos ou uma caixa de diálogo de aprovação modal). O aplicativo de verificação espera receber o código de autenticação por meio do canal primário e compara o hash do valor recebido com o hash do código de autenticação original. Se eles corresponderem, o verificador fora da banda pode assumir que o usuário foi autenticado.

O ASVS assume que apenas alguns desenvolvedores desenvolverão novos autenticadores fora de banda, como notificações push e, portanto, os seguintes controles ASVS se aplicam a verificadores, como API de autenticação, aplicativos e implementações de logon único. Se estiver desenvolvendo um novo autenticador fora de banda, consulte NIST 800-63B § 5.1.3.1.

Autenticadores fora de banda inseguros, como e-mail e VOIP, não são permitidos. A autenticação PSTN e SMS são atualmente "restritas" pelo NIST e devem ser substituídas por notificações push ou semelhantes. Se você precisar usar a autenticação fora de banda por telefone ou SMS, consulte § 5.1.3.3.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.7.1** | Verifique se os autenticadores de texto não criptografado fora da banda (NIST "restrito"), como SMS ou PSTN, não são oferecidos por padrão e se alternativas mais fortes, como notificações push, são oferecidas primeiro. | ✓ | ✓ | ✓ | 287 | 5.1.3.2 |
| **2.7.2** | Verifique se o verificador fora da banda expira as solicitações, códigos ou tokens de autenticação fora da banda após 10 minutos. | ✓ | ✓ | ✓ | 287 | 5.1.3.2 |
| **2.7.3** | Verifique se as solicitações, códigos ou tokens de autenticação do verificador fora da banda só podem ser usados ​​uma vez e apenas para a solicitação de autenticação original. | ✓ | ✓ | ✓ | 287 | 5.1.3.2 |
| **2.7.4** | Verifique se o autenticador e verificador fora da banda se comunicam por meio de um canal independente e seguro. | ✓ | ✓ | ✓ | 523 | 5.1.3.2 |
| **2.7.5** | Verifique se o verificador fora da banda retém apenas uma versão com hash do código de autenticação. |  | ✓ | ✓ | 256 | 5.1.3.2 |
| **2.7.6** | Verifique se o código de autenticação inicial é gerado por um gerador de número aleatório seguro, contendo pelo menos 20 bits de entropia (normalmente, um número aleatório digital seis é suficiente). |  | ✓ | ✓ | 310 | 5.1.3.2 |

## V2.8 Requisitos do verificador único ou multifatorial

As senhas de uso único (OTPs) de fator único são tokens físicos ou soft que exibem um desafio único pseudo-aleatório em constante mudança. Esses dispositivos tornam o phishing (falsificação de identidade) difícil, mas não impossível. Este tipo de autenticador é considerado "algo que você tem". Os tokens multifatoriais são semelhantes às OTPs de fator único, mas exigem um código PIN válido, desbloqueio biométrico, inserção USB ou emparelhamento NFC ou algum valor adicional (como calculadoras de assinatura de transação) a ser inserido para criar a OTP final.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.8.1** | Verifique se os OTPs baseados em tempo têm uma vida útil definida antes de expirar. | ✓ | ✓ | ✓ | 613 | 5.1.4.2 / 5.1.5.2 |
| **2.8.2** | Verifique se as chaves simétricas usadas para verificar OTPs enviadas são altamente protegidas, como usando um módulo de segurança de hardware ou armazenamento de chave baseado em sistema operacional seguro. |  | ✓ | ✓ | 320 | 5.1.4.2 / 5.1.5.2 |
| **2.8.3** | Verifique se os algoritmos criptográficos aprovados são usados ​​na geração, propagação e verificação de OTPs. |  | ✓ | ✓ | 326 | 5.1.4.2 / 5.1.5.2 |
| **2.8.4** | Verifique se o OTP baseado em tempo pode ser usado apenas uma vez dentro do período de validade. |  | ✓ | ✓ | 287 | 5.1.4.2 / 5.1.5.2 |
| **2.8.5** | Verifique se um token OTP multifatorial baseado em tempo é reutilizado durante o período de validade, ele é registrado e rejeitado com notificações seguras sendo enviadas ao titular do dispositivo. |  | ✓ | ✓ | 287 | 5.1.5.2 |
| **2.8.6** | Verifique se o gerador físico de fator único de OTP pode ser revogado em caso de roubo ou outra perda. Certifique-se de que a revogação tenha efeito imediato nas sessões conectadas, independentemente do local. |  | ✓ | ✓ | 613 | 5.2.1 |
| **2.8.7** | Verifique se os autenticadores biométricos estão limitados ao uso apenas como fatores secundários em conjunto com algo que você possui e com algo que você conhece. |  | o | ✓ | 308 | 5.2.3 |

## V2.9 Software criptográfico e requisitos do verificador de dispositivos

As chaves de segurança criptográficas são cartões inteligentes ou chaves FIDO, em que o usuário deve conectar ou emparelhar o dispositivo criptográfico ao computador para concluir a autenticação. Os verificadores enviam um desafio aos dispositivos criptográficos ou software, e o dispositivo ou software calcula uma resposta com base em uma chave criptográfica armazenada com segurança.

Os requisitos para dispositivos e software criptográficos de fator único e dispositivos e software criptográficos multifatoriais são os mesmos, pois a verificação do autenticador criptográfico prova a posse do fator de autenticação.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.9.1** | Verifique se as chaves criptográficas usadas na verificação estão armazenadas com segurança e protegidas contra divulgação, como usar um Módulo de plataforma confiável (TPM) ou Módulo de segurança de hardware (HSM) ou um serviço de sistema operacional que pode usar esse armazenamento seguro. |  | ✓ | ✓ | 320 | 5.1.7.2 |
| **2.9.2** | Verifique se o nonce de desafio tem pelo menos 64 bits de comprimento e é estatisticamente exclusivo ou exclusivo durante a vida útil do dispositivo criptográfico. |  | ✓ | ✓ | 330 | 5.1.7.2 |
| **2.9.3** | Verifique se os algoritmos criptográficos aprovados são usados ​​na geração, propagação e verificação. |  | ✓ | ✓ | 327 | 5.1.7.2 |

## Requisitos de autenticação de serviço V2.10

Esta seção não é testável por penetração, portanto, não possui requisitos L1. No entanto, se usado em uma arquitetura, codificação ou revisão de código seguro, assuma que o software (assim como o Java Key Store) é o requisito mínimo em L1. O armazenamento de texto não criptografado de segredos não é aceitável em nenhuma circunstância.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **2.10.1** | Verifique se os segredos internos do serviço não dependem de credenciais imutáveis, como senhas, chaves de API ou contas compartilhadas com acesso privilegiado. |  | Assistido por sistema operacional | HSM | 287 | 5.1.1.1 |
| **2.10.2** | Verifique se as senhas são exigidas para autenticação de serviço, a conta de serviço usada não é uma credencial padrão. (por exemplo, root / root ou admin / admin são padrão em alguns serviços durante a instalação). |  | Assistido por sistema operacional | HSM | 255 | 5.1.1.1 |
| **2.10.3** | Verifique se as senhas estão armazenadas com proteção suficiente para evitar ataques de recuperação offline, incluindo acesso ao sistema local. |  | Assistido por sistema operacional | HSM | 522 | 5.1.1.1 |
| **2.10.4** | Verifique se as senhas, integrações com bancos de dados e sistemas de terceiros, sementes e segredos internos e chaves de API são gerenciadas com segurança e não estão incluídas no código-fonte ou armazenadas em repositórios de código-fonte. Esse armazenamento DEVE resistir a ataques offline. O uso de um armazenamento de chaves de software seguro (L1), TPM de hardware ou um HSM (L3) é recomendado para armazenamento de senha. |  | Assistido por sistema operacional | HSM | 798 |  |

## Requisitos adicionais da agência dos EUA

As agências dos EUA têm requisitos obrigatórios em relação ao NIST 800-63. O Application Security Verification Standard sempre foi sobre 80% dos controles que se aplicam a quase 100% dos aplicativos, e não os últimos 20% dos controles avançados ou aqueles que têm aplicabilidade limitada. Como tal, o ASVS é um subconjunto estrito do NIST 800-63, especialmente para as classificações IAL1 / 2 e AAL1 / 2, mas não é suficientemente abrangente, particularmente no que diz respeito às classificações IAL3 / AAL3.

Recomendamos enfaticamente que as agências governamentais dos Estados Unidos revisem e implementem o NIST 800-63 em sua totalidade.

## Glossário de termos

| Prazo | Significado |
| --- | --- |
| CSP | Provedor de serviços de credencial, também chamado de provedor de identidade |
| Autenticador | Código que autentica uma senha, token, MFA, asserção federada e assim por diante. |
| Verificador | "Uma entidade que verifica a identidade do reclamante ao verificar a posse e o controle do reclamante de um ou dois autenticadores usando um protocolo de autenticação. Para fazer isso, o verificador também pode precisar validar as credenciais que ligam o (s) autenticador (es) ao identificador do assinante e verificar o status deles " |
| OTP | Senha de uso único |
| SFA | Autenticadores de fator único, como algo que você conhece (segredos memorizados, senhas, frases-senha, PINs), algo que você é (biometria, impressão digital, varreduras de rosto) ou algo que você possui (tokens OTP, um dispositivo criptográfico como um cartão inteligente) , |
| MFA | Autenticação multifator, que inclui dois ou mais fatores únicos |

## Referências

Para obter mais informações, consulte também:

* [NIST 800-63 - Diretrizes de identidade digital](https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf)
* [NIST 800-63 A - Inscrição e prova de identidade](https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63a.pdf)
* [NIST 800-63 B - Gerenciamento de autenticação e ciclo de vida](https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf)
* [NIST 800-63 C - Federação e Asserções](https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63c.pdf)
* [NIST 800-63 FAQ](https://pages.nist.gov/800-63-FAQ/)
* [OWASP Testing Guide 4.0: Testing for Authentication](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/04-Authentication_Testing/README.html)
* [Folha de referências do OWASP - Armazenamento de senha](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html)
* [Folha de referências do OWASP - Esqueci a senha](https://cheatsheetseries.owasp.org/cheatsheets/Forgot_Password_Cheat_Sheet.html)
* [Folha de referências do OWASP - Escolhendo e usando perguntas de segurança](https://cheatsheetseries.owasp.org/cheatsheets/Choosing_and_Using_Security_Questions_Cheat_Sheet.html)

# V3: Requisitos de verificação de gerenciamento de sessão

## Objetivo de Controle

Um dos principais componentes de qualquer aplicativo baseado na web ou API com estado é o mecanismo pelo qual ele controla e mantém o estado de um usuário ou dispositivo que interage com ele. O gerenciamento de sessão altera um protocolo sem estado para com estado, o que é crítico para diferenciar diferentes usuários ou dispositivos.

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de gerenciamento de sessão de alto nível:

* As sessões são exclusivas para cada indivíduo e não podem ser adivinhadas ou compartilhadas.
* As sessões são invalidadas quando não são mais necessárias e expiram durante os períodos de inatividade.

Conforme observado anteriormente, esses requisitos foram adaptados para ser um subconjunto compatível de controles NIST 800-63b selecionados, focados em ameaças comuns e fraquezas de autenticação comumente exploradas. Os requisitos de verificação anteriores foram retirados, desduplicados ou, na maioria dos casos, adaptados para serem fortemente alinhados com a intenção de[NIST 800-63b](https://pages.nist.gov/800-63-3/sp800-63b.html) requisitos.

## Requisitos de verificação de segurança

## Requisitos fundamentais de gerenciamento de sessão V3.1

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.1.1** | Verifique se o aplicativo nunca revela tokens de sessão em parâmetros de URL. | ✓ | ✓ | ✓ | 598 |  |

## Requisitos de vinculação de sessão V3.2

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.2.1** | Verifique se o aplicativo gera um novo token de sessão na autenticação do usuário. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 384 | 7,1 |
| **3.2.2** | Verifique se os tokens de sessão possuem pelo menos 64 bits de entropia. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 331 | 7,1 |
| **3.2.3** | Verifique se o aplicativo armazena apenas tokens de sessão no navegador usando métodos seguros, como cookies devidamente protegidos (consulte a seção 3.4) ou armazenamento de sessão HTML 5. | ✓ | ✓ | ✓ | 539 | 7,1 |
| **3.2.4** | Verifique se o token de sessão é gerado usando algoritmos criptográficos aprovados. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 331 | 7,1 |

TLS ou outro canal de transporte seguro é obrigatório para o gerenciamento de sessão. Isso é abordado no capítulo Segurança das comunicações.

## Requisitos de tempo limite e logoff da sessão V3.3

Os tempos limite de sessão foram alinhados com o NIST 800-63, que permite tempos limite de sessão muito mais longos do que o tradicionalmente permitido pelos padrões de segurança. As organizações devem revisar a tabela abaixo e, se um tempo limite maior for desejável com base no risco do aplicativo, o valor NIST deve ser o limite superior dos tempos limite ociosos da sessão.

L1, neste contexto, é IAL1 / AAL1, L2 é IAL2 / AAL3, L3 é IAL3 / AAL3. Para IAL2 / AAL2 e IAL3 / AAL3, quanto menor o tempo limite de inatividade, o limite inferior de tempos de inatividade para ser desconectado ou autenticado novamente para retomar a sessão.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.3.1** | Verifique se o logout e a expiração invalidam o token de sessão, de forma que o botão Voltar ou uma terceira parte confiável não retome uma sessão autenticada, inclusive entre as partes confiáveis. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 613 | 7,1 |
| **3.3.2** | Se os autenticadores permitirem que os usuários permaneçam conectados, verifique se a reautenticação ocorre periodicamente tanto quando usados ​​ativamente quanto após um período ocioso. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | 30 dias | 12 horas ou 30 minutos de inatividade, 2FA opcional | 12 horas ou 15 minutos de inatividade, com 2FA | 613 | 7,2 |
| **3.3.3** | Verifique se o aplicativo oferece a opção de encerrar todas as outras sessões ativas após uma alteração de senha bem-sucedida (incluindo alteração por meio de redefinição / recuperação de senha) e se isso é eficaz em todo o aplicativo, login federado (se presente) e quaisquer partes confiantes. |  | ✓ | ✓ | 613 |  |
| **3.3.4** | Verifique se os usuários são capazes de visualizar e (após inserir as credenciais de login novamente) fazer logout de qualquer ou de todas as sessões e dispositivos atualmente ativos. |  | ✓ | ✓ | 613 | 7,1 |

## V3.4 Gerenciamento de sessão baseado em cookie

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.4.1** | Verifique se os tokens de sessão baseados em cookies têm o atributo 'Seguro' definido. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 614 | 7.1.1 |
| **3.4.2** | Verifique se os tokens de sessão baseados em cookies têm o atributo 'HttpOnly' definido. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 1004 | 7.1.1 |
| **3.4.3** | Verifique se os tokens de sessão baseados em cookies utilizam o atributo 'SameSite' para limitar a exposição a ataques de falsificação de solicitação entre sites. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 16 | 7.1.1 |
| **3.4.4** | Verifique se os tokens de sessão baseados em cookie usam o prefixo "\_\_Host-" (consulte as referências) para fornecer a confidencialidade do cookie de sessão. | ✓ | ✓ | ✓ | 16 | 7.1.1 |
| **3.4.5** | Verifique se o aplicativo é publicado sob um nome de domínio com outros aplicativos que definem ou usam cookies de sessão que podem substituir ou divulgar os cookies de sessão, defina o atributo de caminho em tokens de sessão baseados em cookies usando o caminho mais preciso possível. ([C6](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 16 | 7.1.1 |

## Gerenciamento de sessão baseado em token V3.5

O gerenciamento de sessão baseado em token inclui chaves JWT, OAuth, SAML e API. Destas, as chaves de API são conhecidas por serem fracas e não devem ser usadas em novos códigos.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.5.1** | Verifique se o aplicativo permite que os usuários revoguem tokens OAuth que formam relações de confiança com aplicativos vinculados. |  | ✓ | ✓ | 290 | 7.1.2 |
| **3.5.2** | Verifique se o aplicativo usa tokens de sessão em vez de chaves e segredos de API estáticos, exceto com implementações legadas. |  | ✓ | ✓ | 798 |  |
| **3.5.3** | Verifique se os tokens de sessão sem estado usam assinaturas digitais, criptografia e outras contramedidas para proteger contra adulteração, envolvimento, reprodução, cifra nula e ataques de substituição de chave. |  | ✓ | ✓ | 345 |  |

## V3.6 Reautenticação de uma Federação ou Asserção

Esta seção se refere àqueles que estão escrevendo o código de Parte Confiante (RP) ou Provedor de Serviços de Credencial (CSP). Se depender do código que implementa esses recursos, certifique-se de que esses problemas sejam tratados corretamente.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.6.1** | Verifique se as partes confiáveis ​​especificam o tempo máximo de autenticação para Provedores de Serviços de Credencial (CSPs) e se os CSPs autenticam novamente o assinante se eles não tiverem usado uma sessão dentro desse período. |  |  | ✓ | 613 | 7.2.1 |
| **3.6.2** | Verifique se os Provedores de Serviços de Credencial (CSPs) informam as Partes Confiantes (RPs) sobre o último evento de autenticação, para permitir que os RPs determinem se precisam reautenticar o usuário. |  |  | ✓ | 613 | 7.2.1 |

## V3.7 Defesas contra exploits de gerenciamento de sessão

Há um pequeno número de ataques de gerenciamento de sessão, alguns relacionados à experiência do usuário (UX) das sessões. Anteriormente, com base nos requisitos da ISO 27002, o ASVS exigia o bloqueio de várias sessões simultâneas. Bloquear sessões simultâneas não é mais apropriado, não apenas porque os usuários modernos têm muitos dispositivos ou o aplicativo é uma API sem uma sessão do navegador, mas na maioria dessas implementações, o último autenticador vence, que muitas vezes é o invasor. Esta seção fornece orientações sobre como dissuadir, atrasar e detectar ataques de gerenciamento de sessão usando código.

### Descrição do Ataque semi-aberto

No início de 2018, várias instituições financeiras foram comprometidas usando o que os invasores chamaram de "ataques semi-abertos". Este termo ficou preso na indústria. Os invasores atingiram várias instituições com diferentes bases de código proprietário e, de fato, parecem diferentes bases de código dentro das mesmas instituições. O ataque semi-aberto está explorando uma falha de padrão de design comumente encontrada em muitos sistemas existentes de autenticação, gerenciamento de sessão e controle de acesso.

Os invasores iniciam um ataque semi-aberto tentando bloquear, redefinir ou recuperar uma credencial. Um padrão de design de gerenciamento de sessão popular reutiliza objetos / modelos de sessão de perfil de usuário entre código não autenticado, meio autenticado (redefinições de senha, nome de usuário esquecido) e código totalmente autenticado. Este padrão de design preenche um objeto de sessão válido ou token contendo o perfil da vítima, incluindo hashes de senha e funções. Se as verificações de controle de acesso em controladores ou roteadores não verificarem corretamente se o usuário está totalmente conectado, o invasor poderá agir como o usuário. Os ataques podem incluir alterar a senha do usuário para um valor conhecido, atualizar o endereço de e-mail para realizar uma redefinição de senha válida, desativar a autenticação multifator ou inscrever um novo dispositivo MFA, revelar ou alterar chaves de API e assim por diante.

| # | Descrição | L1 | L2 | L3 | CWE | [NIST §](https://pages.nist.gov/800-63-3/sp800-63b.html) |
| --- | --- | --- | --- | --- | --- | --- |
| **3.7.1** | Verifique se o aplicativo garante uma sessão de login válida e completa ou requer reautenticação ou verificação secundária antes de permitir quaisquer transações confidenciais ou modificações na conta. | ✓ | ✓ | ✓ | 306 |  |

## Referências

Para obter mais informações, consulte também:

* [OWASP Testing Guide 4.0: Session Management Testing](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/README.html)
* [Folha de referências do gerenciamento de sessão OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html)
* [Set-Cookie \_\_Host- detalhes do prefixo](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie#Directives)

# V4: Requisitos de verificação de controle de acesso

## Objetivo de Controle

Autorização é o conceito de permitir o acesso aos recursos apenas para aqueles que têm permissão para usá-los. Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* Pessoas que acessam recursos possuem credenciais válidas para fazer isso.
* Os usuários são associados a um conjunto bem definido de funções e privilégios.
* Os metadados de função e permissão são protegidos contra reprodução ou adulteração.

## Requisitos de verificação de segurança

## Projeto de controle de acesso geral V4.1

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **4.1.1** | Verifique se o aplicativo aplica as regras de controle de acesso em uma camada de serviço confiável, especialmente se o controle de acesso do lado do cliente estiver presente e puder ser ignorado. | ✓ | ✓ | ✓ | 602 |
| **4.1.2** | Verifique se todos os usuários e atributos de dados e informações de política usados ​​pelos controles de acesso não podem ser manipulados pelos usuários finais, a menos que especificamente autorizado. | ✓ | ✓ | ✓ | 639 |
| **4.1.3** | Verifique se o princípio do menor privilégio existe - os usuários só devem ser capazes de acessar funções, arquivos de dados, URLs, controladores, serviços e outros recursos, para os quais possuam autorização específica. Isso implica proteção contra falsificação e elevação de privilégio. ([C7](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 285 |
| **4.1.4** | Verifique se o princípio de negar por padrão existe, em que novos usuários / funções começam com o mínimo ou nenhuma permissão e os usuários / funções não recebem acesso aos novos recursos até que o acesso seja explicitamente atribuído. ([C7](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 276 |
| **4.1.5** | Verifique se os controles de acesso falham com segurança, incluindo quando ocorre uma exceção. ([C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 285 |

## V4.2 Controle de acesso de nível de operação

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **4.2.1** | Verifique se os dados confidenciais e APIs estão protegidos contra ataques Insecure Direct Object Reference (IDOR) que visam a criação, leitura, atualização e exclusão de registros, como criar ou atualizar o registro de outra pessoa, visualizar os registros de todos ou excluir todos os registros. | ✓ | ✓ | ✓ | 639 |
| **4.2.2** | Verifique se o aplicativo ou estrutura impõe um forte mecanismo anti-CSRF para proteger a funcionalidade autenticada e a anti-automação ou anti-CSRF eficaz protege a funcionalidade não autenticada. | ✓ | ✓ | ✓ | 352 |

## V4.3 Outras Considerações de Controle de Acesso

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **4.3.1** | Verifique se as interfaces administrativas usam autenticação multifator apropriada para evitar o uso não autorizado. | ✓ | ✓ | ✓ | 419 |
| **4.3.2** | Verifique se a navegação no diretório está desabilitada, a menos que seja deliberadamente desejado. Além disso, os aplicativos não devem permitir a descoberta ou divulgação de metadados de arquivos ou diretórios, como pastas Thumbs.db, .DS\_Store, .git ou .svn. | ✓ | ✓ | ✓ | 548 |
| **4.3.3** | Verifique se o aplicativo tem autorização adicional (como autenticação progressiva ou adaptativa) para sistemas de valor inferior e / ou segregação de funções para aplicativos de alto valor para impor controles antifraude de acordo com o risco de aplicativo e fraude anterior. |  | ✓ | ✓ | 732 |

## Referências

Para obter mais informações, consulte também:

* [Guia de Teste OWASP 4.0: Autorização](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/05-Authorization_Testing/README.html)
* [Folha de referências do OWASP: controle de acesso](https://cheatsheetseries.owasp.org/cheatsheets/Access_Control_Cheat_Sheet.html)
* [Folha de referências do OWASP CSRF](https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html)
* [Folha de referências do OWASP REST](https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html)

# V5: Requisitos de validação, higienização e verificação de codificação

## Objetivo de Controle

O ponto fraco de segurança de aplicativo da web mais comum é a falha em validar adequadamente a entrada vinda do cliente ou do ambiente antes de usá-la diretamente, sem qualquer codificação de saída. Essa fraqueza leva a quase todas as vulnerabilidades significativas em aplicativos da web, como Cross-Site Scripting (XSS), injeção de SQL, injeção de intérprete, ataques de local / Unicode, ataques de sistema de arquivos e estouro de buffer.

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* A validação de entrada e a arquitetura de codificação de saída têm um pipeline acordado para evitar ataques de injeção.
* Os dados de entrada são fortemente digitados, validados, o intervalo ou comprimento verificado ou, na pior das hipóteses, higienizados ou filtrados.
* Os dados de saída são codificados ou escapados de acordo com o contexto dos dados, o mais próximo possível do intérprete.

Com a arquitetura moderna de aplicativos da web, a codificação de saída é mais importante do que nunca. É difícil fornecer validação de entrada robusta em certos cenários, portanto, o uso de API mais segura, como consultas parametrizadas, estruturas de modelagem com escape automático ou codificação de saída cuidadosamente escolhida é fundamental para a segurança do aplicativo.

## Requisitos de validação de entrada V5.1

Controles de validação de entrada implementados adequadamente, usando listas de permissões positivas e forte tipagem de dados, podem eliminar mais de 90% de todos os ataques de injeção. Verificações de comprimento e alcance podem reduzir isso ainda mais. Construir uma validação de entrada segura é necessária durante a arquitetura do aplicativo, sprints de design, codificação e teste de unidade e integração. Embora muitos desses itens não possam ser encontrados em testes de penetração, os resultados da não implementação deles são geralmente encontrados em V5.3 - Requisitos de codificação de saída e prevenção de injeção. Os desenvolvedores e revisores de código seguro são recomendados para tratar esta seção como se L1 fosse necessário para todos os itens para evitar injeções.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **5.1.1** | Verifique se o aplicativo tem defesas contra ataques de poluição de parâmetro HTTP, especialmente se a estrutura do aplicativo não faz distinção sobre a origem dos parâmetros de solicitação (GET, POST, cookies, cabeçalhos ou variáveis ​​de ambiente). | ✓ | ✓ | ✓ | 235 |
| **5.1.2** | Verifique se as estruturas protegem contra ataques de atribuição de parâmetro em massa ou se o aplicativo tem contramedidas para proteger contra atribuição de parâmetro insegura, como marcar campos como privados ou semelhantes. ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 915 |
| **5.1.3** | Verifique se todas as entradas (campos de formulário HTML, solicitações REST, parâmetros de URL, cabeçalhos HTTP, cookies, arquivos em lote, feeds RSS, etc.) são validadas usando validação positiva (listas de permissão). ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 20 |
| **5.1.4** | Verifique se os dados estruturados são fortemente digitados e validados em relação a um esquema definido, incluindo caracteres permitidos, comprimento e padrão (por exemplo, números de cartão de crédito ou telefone, ou validar se dois campos relacionados são razoáveis, como verificar se o bairro e o CEP / código postal correspondem). ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 20 |
| **5.1.5** | Verifique se os redirecionamentos e encaminhamentos de URL permitem apenas destinos que aparecem em uma lista de permissões ou mostram um aviso ao redirecionar para conteúdo potencialmente não confiável. | ✓ | ✓ | ✓ | 601 |

## 

## Requisitos de sanitização e sandbox da V5.2

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **5.2.1** | Verifique se todas as entradas HTML não confiáveis ​​de editores WYSIWYG ou similares foram devidamente sanitizadas com uma biblioteca de sanitizer HTML ou recurso de estrutura. ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 116 |
| **5.2.2** | Verifique se os dados não estruturados são limpos para aplicar medidas de segurança, como caracteres permitidos e comprimento. | ✓ | ✓ | ✓ | 138 |
| **5.2.3** | Verifique se o aplicativo limpa a entrada do usuário antes de passar para os sistemas de e-mail para proteção contra injeção de SMTP ou IMAP. | ✓ | ✓ | ✓ | 147 |
| **5.2.4** | Verifique se o aplicativo evita o uso de eval () ou outros recursos de execução de código dinâmico. Onde não houver alternativa, qualquer entrada do usuário incluída deve ser higienizada ou colocada em área restrita antes de ser executada. | ✓ | ✓ | ✓ | 95 |
| **5.2.5** | Verifique se o aplicativo protege contra ataques de injeção de modelo, garantindo que qualquer entrada do usuário incluída seja higienizada ou em área restrita. | ✓ | ✓ | ✓ | 94 |
| **5.2.6** | Verifique se o aplicativo protege contra ataques SSRF, validando ou higienizando dados não confiáveis ​​ou metadados de arquivos HTTP, como nomes de arquivos e campos de entrada de URL, e usa listas de permissão de protocolos, domínios, caminhos e portas. | ✓ | ✓ | ✓ | 918 |
| **5.2.7** | Verifique se o aplicativo limpa, desabilita ou protege o conteúdo de script SVG (Scalable Vector Graphics) fornecido pelo usuário, especialmente no que se refere a XSS resultante de scripts embutidos e de ForeignObject. | ✓ | ✓ | ✓ | 159 |
| **5.2.8** | Verifique se o aplicativo higieniza, desabilita ou protege o conteúdo de linguagem de modelo de expressão ou script fornecido pelo usuário, como Markdown, CSS ou folhas de estilo XSL, BBCode ou semelhante. | ✓ | ✓ | ✓ | 94 |

## Requisitos de codificação de saída e prevenção de injeção V5.3

A codificação de saída próxima ou adjacente ao interpretador em uso é crítica para a segurança de qualquer aplicativo. Normalmente, a codificação de saída não é persistente, mas usada para tornar a saída segura no contexto de saída apropriado para uso imediato. A falha na codificação de saída resultará em um aplicativo inseguro, injetável e inseguro.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **5.3.1** | Verifique se a codificação de saída é relevante para o interpretador e o contexto necessário. Por exemplo, use codificadores especificamente para valores HTML, atributos HTML, JavaScript, parâmetros de URL, cabeçalhos HTTP, SMTP e outros conforme o contexto exigir, especialmente de entradas não confiáveis ​​(por exemplo, nomes com Unicode ou apóstrofos, como ね こ ou O'Hara) . ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 116 |
| **5.3.2** | Verifique se a codificação de saída preserva o conjunto de caracteres e localidade escolhidos pelo usuário, de forma que qualquer ponto de caractere Unicode seja válido e manipulado com segurança. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 176 |
| **5.3.3** | Verifique se o escape de saída ciente do contexto, de preferência automatizado - ou, na pior das hipóteses, manual - protege contra XSS refletido, armazenado e baseado em DOM. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 79 |
| **5.3.4** | Verifique se a seleção de dados ou as consultas de banco de dados (por exemplo, SQL, HQL, ORM, NoSQL) usam consultas parametrizadas, ORMs, estruturas de entidade ou estão protegidos de ataques de injeção de banco de dados. ([C3](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 89 |
| **5.3.5** | Verifique se onde os mecanismos parametrizados ou mais seguros não estão presentes, a codificação de saída específica do contexto é usada para proteger contra ataques de injeção, como o uso de escape de SQL para proteger contra injeção de SQL. ([C3, C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 89 |
| **5.3.6** | Verifique se o aplicativo protege contra ataques de injeção de JavaScript ou JSON, incluindo para ataques eval, JavaScript remoto inclui, contornos de política de segurança de conteúdo (CSP), DOM XSS e avaliação de expressão JavaScript. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 830 |
| **5.3.7** | Verifique se o aplicativo protege contra vulnerabilidades de injeção de LDAP ou se os controles de segurança específicos para impedir a injeção de LDAP foram implementados. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 90 |
| **5.3.8** | Verifique se o aplicativo protege contra injeção de comando do sistema operacional e se as chamadas do sistema operacional usam consultas parametrizadas do sistema operacional ou codificação de saída de linha de comando contextual. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 78 |
| **5.3.9** | Verifique se o aplicativo protege contra ataques de Inclusão de Arquivo Local (LFI) ou Inclusão de Arquivo Remoto (RFI). | ✓ | ✓ | ✓ | 829 |
| **5.3.10** | Verifique se o aplicativo protege contra injeção de XPath ou ataques de injeção de XML. ([C4](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 643 |

Nota: Usar consultas parametrizadas ou escapar de SQL nem sempre é suficiente; nomes de tabelas e colunas, ORDER BY e assim por diante, não podem ser escapados. A inclusão de dados fornecidos pelo usuário de escape nesses campos resulta em consultas com falha ou injeção de SQL.

Observação: o formato SVG permite explicitamente o script ECMA em quase todos os contextos, portanto, pode não ser possível bloquear todos os vetores XSS SVG completamente. Se o upload de SVG for necessário, recomendamos veicularmente esses arquivos carregados como texto / simples ou usar um domínio de conteúdo separado fornecido pelo usuário para evitar que o XSS assuma o controle do aplicativo.

## Requisitos de memória, string e código não gerenciado V5.4

Os requisitos a seguir só se aplicam quando o aplicativo usa uma linguagem de sistema ou código não gerenciado.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **5.4.1** | Verifique se o aplicativo usa string segura para memória, cópia de memória mais segura e aritmética de ponteiro para detectar ou evitar estouros de pilha, buffer ou heap. |  | ✓ | ✓ | 120 |
| **5.4.2** | Verifique se as strings de formato não aceitam entradas potencialmente hostis e são constantes. |  | ✓ | ✓ | 134 |
| **5.4.3** | Verifique se as técnicas de validação de sinal, intervalo e entrada são usadas para evitar estouros de inteiros. |  | ✓ | ✓ | 190 |

## Requisitos de prevenção de desserialização V5.5

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **5.5.1** | Verifique se os objetos serializados usam verificações de integridade ou são criptografados para evitar a criação de objetos hostis ou adulteração de dados. ([C5](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 502 |
| **5.5.2** | Verifique se o aplicativo restringe corretamente os analisadores XML para usar apenas a configuração mais restritiva possível e para garantir que recursos não seguros, como a resolução de entidades externas, sejam desabilitados para evitar ataques XML eXternal Entity (XXE). | ✓ | ✓ | ✓ | 611 |
| **5.5.3** | Verifique se a desserialização de dados não confiáveis ​​é evitada ou protegida no código personalizado e em bibliotecas de terceiros (como analisadores JSON, XML e YAML). | ✓ | ✓ | ✓ | 502 |
| **5.5.4** | Verifique se ao analisar JSON em navegadores ou back-ends baseados em JavaScript, JSON.parse é usado para analisar o documento JSON. Não use eval () para analisar JSON. | ✓ | ✓ | ✓ | 95 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Testing Guide 4.0: Input Validation Testing](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/README.html)
* [Folha de referências do OWASP: validação de entrada](https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html)
* [OWASP Testing Guide 4.0: Testing for HTTP Parameter Pollution](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/04-Testing_for_HTTP_Parameter_Pollution.html)
* [Folha de dicas de injeção OWASP LDAP](https://cheatsheetseries.owasp.org/cheatsheets/LDAP_Injection_Prevention_Cheat_Sheet.html)
* [OWASP Testing Guide 4.0: Client Side Testing](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/11-Client_Side_Testing/)
* [Folha de referências da prevenção de scripts entre sites OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html)
* [Folha de dicas de prevenção de scripts entre sites com base em DOM OWASP](https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html)
* [Projeto de codificação OWASP Java](https://owasp.org/owasp-java-encoder/)
* [Folha de referências da prevenção de atribuição em massa OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Mass_Assignment_Cheat_Sheet.html)
* [DOMPurify - Biblioteca de Sanitização de HTML do lado do cliente](https://github.com/cure53/DOMPurify)
* [Folha de referências de prevenção de XML de Entidade Externa (XXE)](https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html)

Para obter mais informações sobre escape automático, consulte:

* [Reduzindo XSS por meio de escape automático contextual em sistemas de modelo](https://googleonlinesecurity.blogspot.com/2009/03/reducing-xss-by-way-of-automatic.html)
* [AngularJS Strict Contextual Escape](https://docs.angularjs.org/api/ng/service/$sce)
* [AngularJS ngBind](https://docs.angularjs.org/api/ng/directive/ngBind)
* [Sanitização Angular](https://angular.io/guide/security#sanitization-and-security-contexts)
* [Segurança de modelo angular](https://angular.io/guide/template-syntax#content-security)
* [ReactJS Escaping](https://reactjs.org/docs/introducing-jsx.html#jsx-prevents-injection-attacks)
* [Modificação inadequadamente controlada de atributos de objetos determinados dinamicamente](https://cwe.mitre.org/data/definitions/915.html)

Para obter mais informações sobre desserialização, consulte:

* [Folha de referências da desserialização OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Deserialization_Cheat_Sheet.html)
* [Guia de desserialização de dados não confiáveis ​​OWASP](https://owasp.org/www-community/vulnerabilities/Deserialization_of_untrusted_data)

# V6: Requisitos de verificação de criptografia armazenada

## Objetivo de Controle

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* Todos os módulos criptográficos falham de maneira segura e os erros são tratados corretamente.
* Um gerador de números aleatórios adequado é usado.
* O acesso às chaves é gerenciado com segurança.

## Classificação de dados V6.1

O ativo mais importante são os dados processados, armazenados ou transmitidos por um aplicativo. Sempre execute uma avaliação do impacto da privacidade para classificar as necessidades de proteção de dados de quaisquer dados armazenados corretamente.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **6.1.1** | Verifique se os dados privados regulamentados são armazenados criptografados em repouso, como informações de identificação pessoal (PII), informações pessoais confidenciais ou dados avaliados com probabilidade de estarem sujeitos ao GDPR da UE. |  | ✓ | ✓ | 311 |
| **6.1.2** | Verifique se os dados regulamentados de saúde são armazenados criptografados em repouso, como registros médicos, detalhes de dispositivos médicos ou registros de pesquisa anonimizados. |  | ✓ | ✓ | 311 |
| **6.1.3** | Verifique se os dados financeiros regulamentados são armazenados criptografados em repouso, como contas financeiras, padrões ou histórico de crédito, registros de impostos, histórico de pagamentos, beneficiários ou mercado anonimizado ou registros de pesquisa. |  | ✓ | ✓ | 311 |

## Algoritmos V6.2

Avanços recentes em criptografia significam que algoritmos e comprimentos de chave anteriormente seguros não são mais seguros ou suficientes para proteger os dados. Portanto, deve ser possível alterar algoritmos.

Embora esta seção não seja facilmente testada contra penetração, os desenvolvedores devem considerar esta seção inteira como obrigatória, embora L1 esteja faltando na maioria dos itens.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **6.2.1** | Verifique se todos os módulos criptográficos falham com segurança e se os erros são tratados de uma forma que não permite ataques Padding Oracle. | ✓ | ✓ | ✓ | 310 |
| **6.2.2** | Verifique se algoritmos criptográficos, modos e bibliotecas comprovados pela indústria ou aprovados pelo governo são usados, em vez de criptografia codificada personalizada. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 327 |
| **6.2.3** | Verifique se o vetor de inicialização de criptografia, a configuração de cifra e os modos de bloqueio estão configurados com segurança usando o conselho mais recente. |  | ✓ | ✓ | 326 |
| **6.2.4** | Verifique se o número aleatório, criptografia ou algoritmos de hash, comprimentos de chave, rodadas, cifras ou modos podem ser reconfigurados, atualizados ou trocados a qualquer momento, para proteção contra quebras criptográficas. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 326 |
| **6.2.5** | Verifique se os modos de bloco inseguros conhecidos (ou seja, ECB, etc.), modos de preenchimento (ou seja, PKCS # 1 v1.5, etc.), cifras com tamanhos de bloco pequenos (ou seja, Triple-DES, Blowfish, etc.) e algoritmos de hash fracos (ou seja, MD5, SHA1, etc.) não são usados, a menos que seja necessário para compatibilidade com versões anteriores. |  | ✓ | ✓ | 326 |
| **6.2.6** | Verifique se nonces, vetores de inicialização e outros números de uso único não devem ser usados ​​mais de uma vez com uma determinada chave de criptografia. O método de geração deve ser apropriado para o algoritmo que está sendo usado. |  | ✓ | ✓ | 326 |
| **6.2.7** | Verifique se os dados criptografados são autenticados por meio de assinaturas, modos de criptografia autenticados ou HMAC para garantir que o texto criptografado não seja alterado por terceiros não autorizados. |  |  | ✓ | 326 |
| **6.2.8** | Verifique se todas as operações criptográficas são de tempo constante, sem operações de 'curto-circuito' em comparações, cálculos ou retornos, para evitar o vazamento de informações. |  |  | ✓ | 385 |

## Valores Aleatórios V6.3

A verdadeira geração de números pseudo-aleatórios (PRNG) é incrivelmente difícil de acertar. Geralmente, boas fontes de entropia dentro de um sistema serão rapidamente esgotadas se usadas em excesso, mas fontes com menos aleatoriedade podem levar a chaves e segredos previsíveis.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **6.3.1** | Verifique se todos os números aleatórios, nomes de arquivo aleatórios, GUIDs aleatórios e cadeias de caracteres aleatórias são gerados usando o gerador de número aleatório criptograficamente seguro aprovado do módulo criptográfico quando esses valores aleatórios não devem ser adivinhados por um invasor. |  | ✓ | ✓ | 338 |
| **6.3.2** | Verifique se os GUIDs aleatórios são criados usando o algoritmo GUID v4 e um gerador de números pseudoaleatórios criptograficamente seguro (CSPRNG). GUIDs criados usando outros geradores de números pseudo-aleatórios podem ser previsíveis. |  | ✓ | ✓ | 338 |
| **6.3.3** | Verifique se os números aleatórios são criados com a entropia adequada, mesmo quando o aplicativo está sob carga pesada, ou se o aplicativo se degrada normalmente em tais circunstâncias. |  |  | ✓ | 338 |

## Gerenciamento de segredo V6.4

Embora esta seção não seja facilmente testada contra penetração, os desenvolvedores devem considerar esta seção inteira como obrigatória, embora L1 esteja faltando na maioria dos itens.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **6.4.1** | Verifique se uma solução de gerenciamento de segredos, como um cofre de chaves, é usada para criar, armazenar, controlar o acesso e destruir segredos com segurança. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 798 |
| **6.4.2** | Verifique se o material da chave não está exposto ao aplicativo, mas em vez disso, use um módulo de segurança isolado como um cofre para operações criptográficas. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 320 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Testing Guide 4.0: Testando para criptografia fraca](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/09-Testing_for_Weak_Cryptography/README.html)
* [Folha de referências do OWASP: armazenamento criptográfico](https://cheatsheetseries.owasp.org/cheatsheets/Cryptographic_Storage_Cheat_Sheet.html)
* [FIPS 140-2](https://csrc.nist.gov/publications/detail/fips/140/2/final)

# V7: Tratamento de erros e requisitos de verificação de registro

## Objetivo de Controle

O objetivo principal do tratamento e registro de erros é fornecer informações úteis para o usuário, administradores e equipes de resposta a incidentes. O objetivo não é criar grandes quantidades de toras, mas toras de alta qualidade, com mais sinal do que ruído descartado.

Logs de alta qualidade geralmente contêm dados confidenciais e devem ser protegidos de acordo com as leis ou diretivas locais de privacidade de dados. Isso deve incluir:

* Não coletar ou registrar informações confidenciais, a menos que seja especificamente necessário.
* Garantir que todas as informações registradas sejam tratadas com segurança e protegidas de acordo com sua classificação de dados.
* Garantir que os logs não sejam armazenados para sempre, mas tenham uma vida útil absoluta que seja a mais curta possível.

Se os logs contiverem dados privados ou confidenciais, cuja definição varia de país para país, os logs se tornam algumas das informações mais confidenciais mantidas pelo aplicativo e, portanto, muito atraentes para os invasores por direito próprio.

Também é importante garantir que o aplicativo falhe com segurança e que os erros não revelem informações desnecessárias.

## Requisitos de conteúdo de log V7.1

O registro de informações confidenciais é perigoso - os próprios registros tornam-se classificados, o que significa que precisam ser criptografados, estão sujeitos a políticas de retenção e devem ser divulgados em auditorias de segurança. Garanta que apenas as informações necessárias sejam mantidas em registros e, certamente, nenhum pagamento, credenciais (incluindo tokens de sessão), informações confidenciais ou de identificação pessoal.

V7.1 abrange OWASP Top 10 2017: A10. Como 2017: A10 e esta seção não são testáveis ​​de penetração, é importante para:

* Os desenvolvedores devem garantir a conformidade total com esta seção, como se todos os itens estivessem marcados como L1
* Testadores de penetração para validar a conformidade total de todos os itens na V7.1 por meio de entrevista, capturas de tela ou declaração

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **7.1.1** | Verifique se o aplicativo não registra credenciais ou detalhes de pagamento. Os tokens de sessão só devem ser armazenados em logs de forma irreversível e com hash. ([C9, C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 532 |
| **7.1.2** | Verifique se o aplicativo não registra outros dados confidenciais, conforme definido nas leis de privacidade locais ou na política de segurança relevante. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 532 |
| **7.1.3** | Verifique se o aplicativo registra eventos relevantes de segurança, incluindo eventos de autenticação bem-sucedidos e com falha, falhas de controle de acesso, falhas de desserialização e falhas de validação de entrada. ([C5, C7](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 778 |
| **7.1.4** | Verifique se cada evento de log inclui as informações necessárias que permitiriam uma investigação detalhada da linha do tempo quando um evento acontecer. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 778 |

## Requisitos de processamento de log V7.2

O registro oportuno é crítico para eventos de auditoria, triagem e escalonamento. Certifique-se de que os logs do aplicativo estejam claros e possam ser facilmente monitorados e analisados ​​localmente ou enviados para um sistema de monitoramento remoto.

V7.2 abrange OWASP Top 10 2017: A10. Como 2017: A10 e esta seção não são testáveis ​​de penetração, é importante para:

* Os desenvolvedores devem garantir a conformidade total com esta seção, como se todos os itens estivessem marcados como L1
* Testadores de penetração para validar a conformidade total de todos os itens na V7.2 por meio de entrevista, capturas de tela ou declaração

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **7.2.1** | Verifique se todas as decisões de autenticação são registradas, sem armazenar tokens de sessão ou senhas confidenciais. Isso deve incluir solicitações com metadados relevantes necessários para investigações de segurança. |  | ✓ | ✓ | 778 |
| **7.2.2** | Verifique se todas as decisões de controle de acesso podem ser registradas e todas as decisões com falha são registradas. Isso deve incluir solicitações com metadados relevantes necessários para investigações de segurança. |  | ✓ | ✓ | 285 |

## Requisitos de proteção de log V7.3

Logs que podem ser modificados ou excluídos trivialmente são inúteis para investigações e processos. A divulgação de logs pode expor detalhes internos sobre o aplicativo ou os dados que ele contém. Deve-se tomar cuidado ao proteger os logs de divulgação, modificação ou exclusão não autorizada.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **7.3.1** | Verifique se o aplicativo codifica adequadamente os dados fornecidos pelo usuário para evitar injeção de log. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 117 |
| **7.3.2** | Verifique se todos os eventos estão protegidos contra injeção quando visualizados no software de visualização de registros. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 117 |
| **7.3.3** | Verifique se os logs de segurança estão protegidos contra acesso não autorizado e modificação. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 200 |
| **7.3.4** | Verifique se as fontes de tempo estão sincronizadas com a hora e o fuso horário corretos. Considere fortemente o registro apenas em UTC se os sistemas forem globais para auxiliar na análise forense pós-incidente. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ |  |

Nota: A codificação de log (7.3.1) é difícil de testar e revisar usando ferramentas dinâmicas automatizadas e testes de penetração, mas arquitetos, desenvolvedores e revisores de código-fonte devem considerá-la um requisito L1.

## Tratamento de erros V7.4

O objetivo do tratamento de erros é permitir que o aplicativo forneça eventos relevantes de segurança para monitoramento, triagem e escalonamento. O objetivo não é criar logs. Ao registrar eventos relacionados à segurança, certifique-se de que haja uma finalidade para o registro e que ele possa ser diferenciado pelo SIEM ou pelo software de análise.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **7.4.1** | Verifique se uma mensagem genérica é mostrada quando ocorre um erro inesperado ou sensível à segurança, potencialmente com um ID exclusivo que a equipe de suporte pode usar para investigar. ([C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 210 |
| **7.4.2** | Verifique se o tratamento de exceções (ou um equivalente funcional) é usado na base de código para levar em conta as condições de erro esperadas e inesperadas. ([C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 544 |
| **7.4.3** | Verifique se um manipulador de erros de "último recurso" está definido, o qual detectará todas as exceções não tratadas. ([C10](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 431 |

Nota: Certas linguagens, como Swift e Go - e através da prática de design comum - muitas linguagens funcionais, não oferecem suporte a exceções ou manipuladores de eventos de último recurso. Nesse caso, arquitetos e desenvolvedores devem usar um padrão, linguagem ou estrutura amigável para garantir que os aplicativos possam lidar com eventos excepcionais, inesperados ou relacionados à segurança com segurança.

## Referências

Para obter mais informações, consulte também:

* [Conteúdo do OWASP Testing Guide 4.0: Testing for Error Handling](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/08-Testing_for_Error_Handling/README.html)
* [Seção da folha de dicas de autenticação OWASP sobre mensagens de erro](https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#authentication-and-error-messages)

# V8: Requisitos de verificação de proteção de dados

## Objetivo de Controle

Existem três elementos-chave para uma proteção sólida de dados: Confidencialidade, Integridade e Disponibilidade (CIA). Este padrão assume que a proteção de dados é aplicada em um sistema confiável, como um servidor, que foi reforçado e tem proteções suficientes.

Os aplicativos devem presumir que todos os dispositivos do usuário estão comprometidos de alguma forma. Quando um aplicativo transmite ou armazena informações confidenciais em dispositivos inseguros, como computadores, telefones e tablets compartilhados, o aplicativo é responsável por garantir que os dados armazenados nesses dispositivos sejam criptografados e não possam ser facilmente obtidos, alterados ou divulgados de forma ilícita.

Certifique-se de que um aplicativo verificado atenda aos seguintes requisitos de proteção de dados de alto nível:

* Confidencialidade: os dados devem ser protegidos de observação ou divulgação não autorizada, tanto em trânsito como quando armazenados.
* Integridade: os dados devem ser protegidos contra criação, alteração ou exclusão maliciosa de invasores não autorizados.
* Disponibilidade: os dados devem estar disponíveis para usuários autorizados, conforme necessário.

## V8.1 Proteção Geral de Dados

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **8.1.1** | Verifique se o aplicativo protege os dados confidenciais de serem armazenados em cache em componentes do servidor, como balanceadores de carga e caches de aplicativos. |  | ✓ | ✓ | 524 |
| **8.1.2** | Verifique se todas as cópias em cache ou temporárias de dados confidenciais armazenados no servidor estão protegidas contra acesso não autorizado ou eliminadas / invalidadas após o usuário autorizado acessar os dados confidenciais. |  | ✓ | ✓ | 524 |
| **8.1.3** | Verifique se o aplicativo minimiza o número de parâmetros em uma solicitação, como campos ocultos, variáveis ​​Ajax, cookies e valores de cabeçalho. |  | ✓ | ✓ | 233 |
| **8.1.4** | Verifique se o aplicativo pode detectar e alertar sobre números anormais de solicitações, como por IP, usuário, total por hora ou dia ou o que fizer sentido para o aplicativo. |  | ✓ | ✓ | 770 |
| **8.1.5** | Verifique se backups regulares de dados importantes são executados e se a restauração de teste de dados é executada. |  |  | ✓ | 19 |
| **8.1.6** | Verifique se os backups estão armazenados com segurança para evitar que os dados sejam roubados ou corrompidos. |  |  | ✓ | 19 |

## Proteção de dados do lado do cliente V8.2

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **8.2.1** | Verifique se o aplicativo define cabeçalhos anti-cache suficientes para que dados confidenciais não sejam armazenados em cache em navegadores modernos. | ✓ | ✓ | ✓ | 525 |
| **8.2.2** | Verifique se os dados armazenados no armazenamento do navegador (como armazenamento local HTML5, armazenamento de sessão, IndexedDB ou cookies) não contêm dados confidenciais ou PII. | ✓ | ✓ | ✓ | 922 |
| **8.2.3** | Verifique se os dados autenticados são apagados do armazenamento do cliente, como o DOM do navegador, após o cliente ou a sessão ser encerrada. | ✓ | ✓ | ✓ | 922 |

## V8.3 Dados Privados Sensíveis

Esta seção ajuda a proteger dados confidenciais de serem criados, lidos, atualizados ou excluídos sem autorização, especialmente em grandes quantidades.

A conformidade com esta seção implica a conformidade com o controle de acesso V4 e, em particular, com a V4.2. Por exemplo, para se proteger contra atualizações não autorizadas ou divulgação de informações pessoais confidenciais, é necessário seguir a V4.2.1. Cumpra esta seção e V4 para cobertura total.

Observação: os regulamentos e leis de privacidade, como os Princípios de privacidade australianos APP-11 ou GDPR, afetam diretamente como os aplicativos devem abordar a implementação de armazenamento, uso e transmissão de informações pessoais confidenciais. Isso varia de penalidades severas a conselhos simples. Consulte as leis e regulamentos locais e consulte um especialista qualificado em privacidade ou advogado, conforme necessário.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **8.3.1** | Verifique se os dados confidenciais são enviados ao servidor no corpo ou cabeçalhos da mensagem HTTP e se os parâmetros da string de consulta de qualquer verbo HTTP não contêm dados confidenciais. | ✓ | ✓ | ✓ | 319 |
| **8.3.2** | Verifique se os usuários têm um método para remover ou exportar seus dados sob demanda. | ✓ | ✓ | ✓ | 212 |
| **8.3.3** | Verifique se os usuários recebem uma linguagem clara em relação à coleta e uso das informações pessoais fornecidas e se os usuários deram consentimento para o uso desses dados antes de serem usados ​​de qualquer forma. | ✓ | ✓ | ✓ | 285 |
| **8.3.4** | Verifique se todos os dados confidenciais criados e processados ​​pelo aplicativo foram identificados e certifique-se de que haja uma política em vigor sobre como lidar com dados confidenciais. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 200 |
| **8.3.5** | Verifique se o acesso aos dados confidenciais é auditado (sem registrar os próprios dados confidenciais), se os dados são coletados de acordo com as diretivas de proteção de dados relevantes ou onde o registro de acesso é necessário. |  | ✓ | ✓ | 532 |
| **8.3.6** | Verifique se as informações confidenciais contidas na memória são sobrescritas assim que não são mais necessárias para mitigar ataques de despejo de memória, usando zeros ou dados aleatórios. |  | ✓ | ✓ | 226 |
| **8.3.7** | Verifique se as informações confidenciais ou privadas que precisam ser criptografadas são criptografadas usando algoritmos aprovados que fornecem confidencialidade e integridade. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 327 |
| **8.3.8** | Verifique se as informações pessoais confidenciais estão sujeitas à classificação de retenção de dados, de forma que dados antigos ou desatualizados sejam excluídos automaticamente, de acordo com uma programação ou conforme a situação exigir. |  | ✓ | ✓ | 285 |

Ao considerar a proteção de dados, uma consideração primária deve ser em torno da extração ou modificação em massa ou do uso excessivo. Por exemplo, muitos sistemas de mídia social só permitem que os usuários adicionem 100 novos amigos por dia, mas de qual sistema essas solicitações vieram não é importante. Uma plataforma bancária pode desejar bloquear mais de 5 transações por hora, transferindo mais de 1000 euros de fundos para instituições externas. Os requisitos de cada sistema são provavelmente muito diferentes, portanto, a decisão de "anormal" deve levar em consideração o modelo de ameaça e o risco do negócio. Critérios importantes são a capacidade de detectar, impedir ou, de preferência, bloquear essas ações em massa anormais.

## Referências

Para obter mais informações, consulte também:

* [Considere o uso do site Security Headers para verificar a segurança e os cabeçalhos anti-cache](https://securityheaders.io)
* [Projeto OWASP Secure Headers](https://owasp.org/www-project-secure-headers/)
* [Projeto de Riscos de Privacidade OWASP](https://owasp.org/www-project-top-10-privacy-risks/)
* [Folha de dicas de proteção de privacidade do usuário OWASP](https://cheatsheetseries.owasp.org/cheatsheets/User_Privacy_Protection_Cheat_Sheet.html)
* [Visão geral do Regulamento Geral de Proteção de Dados da União Europeia (GDPR)](https://edps.europa.eu/data-protection_en)
* [Supervisor de Proteção de Dados da União Europeia - Rede de Engenharia de Privacidade da Internet](https://edps.europa.eu/data-protection/ipen-internet-privacy-engineering-network_en)

# V9: Requisitos de verificação de comunicações

## Objetivo de Controle

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* TLS ou criptografia forte é sempre usada, independentemente da sensibilidade dos dados que estão sendo transmitidos
* O conselho de configuração líder mais recente é usado para habilitar e solicitar algoritmos e cifras preferidos
* Algoritmos e cifras fracos ou em breve descontinuados são solicitados como último recurso
* Algoritmos e cifras obsoletos ou conhecidos e inseguros estão desativados.

A consultoria líder do setor sobre mudanças seguras de configuração de TLS com frequência, muitas vezes devido a quebras catastróficas nos algoritmos e cifras existentes. Sempre use as versões mais recentes das ferramentas de revisão de configuração TLS (como SSLyze ou outros scanners TLS) para configurar a ordem preferencial e a seleção de algoritmo. A configuração deve ser verificada periodicamente para garantir que a configuração de comunicações seguras esteja sempre presente e eficaz.

## Requisitos de segurança de comunicações do cliente V9.1

Todas as comunicações do cliente devem ocorrer apenas em caminhos de comunicação criptografados. Em particular, o uso de TLS 1.2 ou posterior é essencialmente exigido por navegadores e mecanismos de pesquisa modernos. A configuração deve ser revisada regularmente usando ferramentas online para garantir que as práticas de ponta mais recentes estejam em vigor.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **9.1.1** | Verifique se o TLS protegido é usado para toda a conectividade do cliente e não recorre a protocolos inseguros ou não criptografados. ([C8](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 319 |
| **9.1.2** | Verifique, usando ferramentas de teste TLS on-line ou atualizadas, se apenas algoritmos, cifras e protocolos fortes estão ativados, com os algoritmos e cifras mais fortes definidos como preferidos. | ✓ | ✓ | ✓ | 326 |
| **9.1.3** | Verifique se as versões antigas de protocolos, algoritmos, cifras e configurações SSL e TLS estão desabilitadas, como SSLv2, SSLv3 ou TLS 1.0 e TLS 1.1. A versão mais recente do TLS deve ser o pacote de criptografia preferido. | ✓ | ✓ | ✓ | 326 |

## Requisitos de segurança de comunicações do servidor V9.2

As comunicações do servidor são mais do que apenas HTTP. Conexões seguras de e para outros sistemas, como sistemas de monitoramento, ferramentas de gerenciamento, acesso remoto e ssh, middleware, banco de dados, mainframes, parceiros ou sistemas de origem externa - devem estar em vigor. Todos eles devem ser criptografados para evitar "difícil por fora, trivialmente fácil de interceptar por dentro".

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **9.2.1** | Verifique se as conexões de e para o servidor usam certificados TLS confiáveis. Quando certificados gerados internamente ou autoassinados são usados, o servidor deve ser configurado para confiar apenas em CAs internos específicos e certificados autoassinados específicos. Todos os outros devem ser rejeitados. |  | ✓ | ✓ | 295 |
| **9.2.2** | Verifique se as comunicações criptografadas, como TLS, são usadas para todas as conexões de entrada e saída, incluindo portas de gerenciamento, monitoramento, autenticação, API ou chamadas de serviço da web, banco de dados, nuvem, sem servidor, mainframe, conexões externas e de parceiros. O servidor não deve recorrer a protocolos inseguros ou não criptografados. |  | ✓ | ✓ | 319 |
| **9.2.3** | Verifique se todas as conexões criptografadas com sistemas externos que envolvem informações ou funções confidenciais estão autenticadas. |  | ✓ | ✓ | 287 |
| **9.2.4** | Verifique se a revogação de certificação adequada, como grampeamento de protocolo de status de certificado online (OCSP), está habilitada e configurada. |  | ✓ | ✓ | 299 |
| **9.2.5** | Verifique se as falhas de conexão TLS de back-end são registradas. |  |  | ✓ | 544 |

## Referências

Para obter mais informações, consulte também:

* [OWASP - Folha de referências do TLS](https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html)
* [OWASP - Guia de fixação](https://owasp.org/www-community/controls/Certificate_and_Public_Key_Pinning)
* Notas sobre “Modos aprovados de TLS”. No passado, o ASVS se referia ao padrão dos EUA FIPS 140-2, mas como um padrão global, aplicar os padrões dos EUA pode ser difícil, contraditório ou confuso de aplicar. Um método melhor de alcançar a conformidade com 9.1.3 seria revisar guias como[TLS do lado do servidor da Mozilla](https://wiki.mozilla.org/Security/Server_Side_TLS) ou [gerar configurações boas conhecidas](https://mozilla.github.io/server-side-tls/ssl-config-generator/)e usar ferramentas de avaliação de TLS conhecidas, como sslyze, vários scanners de vulnerabilidade ou serviços de avaliação on-line de TLS confiáveis ​​para obter o nível desejado de segurança. Em geral, vemos a não conformidade para esta seção sendo o uso de cifras e algoritmos desatualizados ou inseguros, a falta de sigilo de encaminhamento perfeito, protocolos SSL desatualizados ou inseguros, cifras preferenciais fracas e assim por diante.

# V10: Requisitos de verificação de código malicioso

## Objetivo de Controle

Certifique-se de que o código atenda aos seguintes requisitos de alto nível:

* A atividade maliciosa é tratada de forma segura e adequada para não afetar o resto do aplicativo.
* Não tem bombas-relógio ou outros ataques baseados no tempo.
* Não "liga para casa" para destinos maliciosos ou não autorizados.
* Não tem portas traseiras, ovos de Páscoa, ataques de salame, rootkits ou código não autorizado que pode ser controlado por um invasor.

Encontrar o código malicioso é a prova do negativo, que é impossível de validar completamente. Os melhores esforços devem ser realizados para garantir que o código não tenha nenhum código malicioso inerente ou funcionalidade indesejada.

## Controles de integridade de código V10.1

A melhor defesa contra códigos maliciosos é "confiar, mas verificar". A introdução de código não autorizado ou malicioso no código é frequentemente um crime em muitas jurisdições. Políticas e procedimentos devem deixar claras as sanções relativas a códigos maliciosos.

Os desenvolvedores líderes devem revisar regularmente as verificações de código, especialmente aquelas que podem acessar o tempo, E / S ou funções de rede.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **10.1.1** | Verifique se uma ferramenta de análise de código está em uso e pode detectar código potencialmente malicioso, como funções de hora, operações de arquivo inseguras e conexões de rede. |  |  | ✓ | 749 |

## V10.2 Pesquisa de código malicioso

O código malicioso é extremamente raro e difícil de detectar. A revisão de código linha por linha manual pode ajudar na procura de bombas lógicas, mas mesmo o revisor de código mais experiente terá dificuldade em encontrar código malicioso, mesmo se souber que ele existe.

Obedecer a esta seção não é possível sem acesso completo ao código-fonte, incluindo bibliotecas de terceiros.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **10.2.1** | Verifique se o código-fonte do aplicativo e as bibliotecas de terceiros não contêm telefone residencial ou recursos de coleta de dados não autorizados. Onde tal funcionalidade existir, obtenha a permissão do usuário para que ela opere antes de coletar quaisquer dados. |  | ✓ | ✓ | 359 |
| **10.2.2** | Verifique se o aplicativo não pede permissões desnecessárias ou excessivas para recursos ou sensores relacionados à privacidade, como contatos, câmeras, microfones ou localização. |  | ✓ | ✓ | 272 |
| **10.2.3** | Verifique se o código-fonte do aplicativo e as bibliotecas de terceiros não contêm backdoors, como contas ou chaves codificadas ou adicionais não documentadas, ofuscação de código, blobs binários não documentados, rootkits ou anti-depuração, recursos de depuração inseguros ou de outra forma fora de funcionalidade de data, insegura ou oculta que pode ser usada maliciosamente se descoberta. |  |  | ✓ | 507 |
| **10.2.4** | Verifique se o código-fonte do aplicativo e as bibliotecas de terceiros não contêm bombas-relógio pesquisando funções relacionadas a data e hora. |  |  | ✓ | 511 |
| **10.2.5** | Verifique se o código-fonte do aplicativo e as bibliotecas de terceiros não contêm código malicioso, como ataques de salame, desvios de lógica ou bombas lógicas. |  |  | ✓ | 511 |
| **10.2.6** | Verifique se o código-fonte do aplicativo e as bibliotecas de terceiros não contêm ovos de Páscoa ou qualquer outra funcionalidade potencialmente indesejada. |  |  | ✓ | 507 |

## V10.3 Controles de integridade de aplicativos implantados

Depois que um aplicativo é implantado, o código malicioso ainda pode ser inserido. Os aplicativos precisam se proteger contra ataques comuns, como a execução de código não assinado de fontes não confiáveis ​​e controle de subdomínios.

O cumprimento desta seção provavelmente será operacional e contínuo.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **10.3.1** | Verifique se o aplicativo possui um recurso de atualização automática de cliente ou servidor, as atualizações devem ser obtidas através de canais seguros e assinadas digitalmente. O código de atualização deve validar a assinatura digital da atualização antes de instalar ou executar a atualização. | ✓ | ✓ | ✓ | 16 |
| **10.3.2** | Verifique se o aplicativo emprega proteções de integridade, como assinatura de código ou integridade de sub-recursos. O aplicativo não deve carregar ou executar código de fontes não confiáveis, como inclusões de carregamento, módulos, plug-ins, código ou bibliotecas de fontes não confiáveis ​​ou da Internet. | ✓ | ✓ | ✓ | 353 |
| **10.3.3** | Verifique se o aplicativo tem proteção contra controle de subdomínio se o aplicativo depende de entradas DNS ou subdomínios DNS, como nomes de domínio expirados, ponteiros DNS desatualizados ou CNAMEs, projetos expirados em repositórios de código-fonte públicos ou APIs de nuvem transitória, funções sem servidor, ou baldes de armazenamento (autogen-bucket-id.cloud.example.com) ou semelhante. As proteções podem incluir a garantia de que os nomes DNS usados ​​pelos aplicativos sejam verificados regularmente quanto à expiração ou alteração. | ✓ | ✓ | ✓ | 350 |

## Referências

* [Aquisição de subdomínio hostil, Detectify Labs](https://labs.detectify.com/2014/10/21/hostile-subdomain-takeover-using-herokugithubdesk-more/)
* [Seqüestro de subdomínios abandonados parte 2, Detectify Labs](https://labs.detectify.com/2014/12/08/hijacking-of-abandoned-subdomains-part-2/)

# V11: Requisitos de verificação de lógica de negócios

## Objetivo de Controle

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* O fluxo de lógica de negócios é sequencial, processado em ordem e não pode ser ignorado.
* A lógica de negócios inclui limites para detectar e prevenir ataques automatizados, como transferências contínuas de pequenos fundos ou adição de um milhão de amigos, um de cada vez, e assim por diante.
* Fluxos de lógica de negócios de alto valor consideraram casos de abuso e agentes mal-intencionados e têm proteções contra falsificação, adulteração, repúdio, divulgação de informações e ataques de elevação de privilégio.

## Requisitos de segurança da lógica de negócios V11.1

A segurança da lógica de negócios é tão individual para cada aplicativo que nenhuma lista de verificação será aplicada. A segurança da lógica de negócios deve ser projetada para proteger contra prováveis ​​ameaças externas - ela não pode ser adicionada usando firewalls de aplicativos da web ou comunicações seguras. Recomendamos o uso de modelagem de ameaças durante sprints de design, por exemplo, usando o OWASP Cornucopia ou ferramentas semelhantes.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **11.1.1** | Verifique se o aplicativo processará apenas fluxos de lógica de negócios para o mesmo usuário na ordem de etapas sequenciais e sem pular etapas. | ✓ | ✓ | ✓ | 841 |
| **11.1.2** | Verifique se o aplicativo processará apenas fluxos de lógica de negócios com todas as etapas sendo processadas em tempo humano realista, ou seja, as transações não são enviadas muito rapidamente. | ✓ | ✓ | ✓ | 799 |
| **11.1.3** | Verifique se o aplicativo tem limites apropriados para ações de negócios ou transações específicas que são aplicadas corretamente por usuário. | ✓ | ✓ | ✓ | 770 |
| **11.1.4** | Verifique se o aplicativo tem controles anti-automação suficientes para detectar e proteger contra exfiltração de dados, solicitações excessivas de lógica de negócios, uploads excessivos de arquivos ou ataques de negação de serviço. | ✓ | ✓ | ✓ | 770 |
| **11.1.5** | Verifique se o aplicativo tem limites de lógica de negócios ou validação para proteger contra prováveis ​​riscos ou ameaças de negócios, identificados por meio de modelagem de ameaças ou metodologias semelhantes. | ✓ | ✓ | ✓ | 841 |
| **11.1.6** | Verifique se o aplicativo não sofre de problemas de "Tempo de verificação para tempo de uso" (TOCTOU) ou outras condições de corrida para operações confidenciais. |  | ✓ | ✓ | 367 |
| **11.1.7** | Verifique se o aplicativo monitora eventos ou atividades incomuns de uma perspectiva de lógica de negócios. Por exemplo, tentativas de executar ações fora de ordem ou ações que um usuário normal nunca tentaria. ([C9](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 754 |
| **11.1.8** | Verifique se o aplicativo possui alertas configuráveis ​​quando ataques automatizados ou atividade incomum são detectados. |  | ✓ | ✓ | 390 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Web Security Testing Guide 4.1: Teste de lógica de negócios](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/10-Business_Logic_Testing/README.html)
* A anti-automação pode ser alcançada de várias maneiras, incluindo o uso de [OWASP AppSensor](https://github.com/jtmelton/appsensor) e [Ameaças automatizadas OWASP para aplicativos da Web](https://owasp.org/www-project-automated-threats-to-web-applications/)
* [OWASP AppSensor](https://github.com/jtmelton/appsensor) também pode ajudar na detecção e resposta a ataques.
* [OWASP Cornucopia](https://owasp.org/www-project-cornucopia/)

# V12: Requisitos de verificação de arquivos e recursos

## Objetivo de Controle

Certifique-se de que um aplicativo verificado satisfaça os seguintes requisitos de alto nível:

* Dados de arquivos não confiáveis ​​devem ser tratados de maneira adequada e segura.
* Dados de arquivos não confiáveis ​​obtidos de fontes não confiáveis ​​são armazenados fora da raiz da web e com permissões limitadas.

## Requisitos de upload de arquivo V12.1

Embora as bombas zip sejam eminentemente testáveis ​​usando técnicas de teste de penetração, elas são consideradas L2 e superiores para encorajar a consideração de design e desenvolvimento com testes manuais cuidadosos e para evitar testes de penetração manuais automatizados ou não qualificados de uma condição de negação de serviço.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.1.1** | Verifique se o aplicativo não aceita arquivos grandes que podem encher o armazenamento ou causar uma negação de serviço. | ✓ | ✓ | ✓ | 400 |
| **12.1.2** | Verifique se os arquivos compactados estão verificados em busca de "bombas zip" - pequenos arquivos de entrada que serão descompactados em arquivos enormes, esgotando assim os limites de armazenamento de arquivos. |  | ✓ | ✓ | 409 |
| **12.1.3** | Verifique se uma cota de tamanho de arquivo e um número máximo de arquivos por usuário são aplicados para garantir que um único usuário não consiga preencher o armazenamento com muitos arquivos ou arquivos excessivamente grandes. |  | ✓ | ✓ | 770 |

## Requisitos de integridade de arquivo V12.2

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.2.1** | Verifique se os arquivos obtidos de fontes não confiáveis ​​são validados para serem do tipo esperado com base no conteúdo do arquivo. |  | ✓ | ✓ | 434 |

## Requisitos de execução de arquivo V12.3

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.3.1** | Verifique se os metadados de nome de arquivo enviados pelo usuário não são usados ​​diretamente pelo sistema ou por sistemas de arquivos de estrutura e se uma API de URL é usada para proteção contra travessia de caminho. | ✓ | ✓ | ✓ | 22 |
| **12.3.2** | Verifique se os metadados de nome de arquivo enviados pelo usuário são validados ou ignorados para evitar a divulgação, criação, atualização ou remoção de arquivos locais (LFI). | ✓ | ✓ | ✓ | 73 |
| **12.3.3** | Verifique se os metadados de nome de arquivo enviados pelo usuário são validados ou ignorados para evitar a divulgação ou execução de arquivos remotos por meio de ataques de inclusão remota de arquivos (RFI) ou de falsificação de solicitações do lado do servidor (SSRF). | ✓ | ✓ | ✓ | 98 |
| **12.3.4** | Verifique se o aplicativo protege contra Reflective File Download (RFD) validando ou ignorando nomes de arquivos enviados pelo usuário em um parâmetro JSON, JSONP ou URL, o cabeçalho Content-Type de resposta deve ser definido como text / plain e o cabeçalho Content-Disposition deve ter um nome de arquivo fixo. | ✓ | ✓ | ✓ | 641 |
| **12.3.5** | Verifique se os metadados de arquivos não confiáveis ​​não são usados ​​diretamente com a API do sistema ou bibliotecas, para proteção contra injeção de comandos do sistema operacional. | ✓ | ✓ | ✓ | 78 |
| **12.3.6** | Verifique se o aplicativo não inclui e executa a funcionalidade de fontes não confiáveis, como redes de distribuição de conteúdo não verificadas, bibliotecas JavaScript, bibliotecas npm de nó ou DLLs do lado do servidor. |  | ✓ | ✓ | 829 |

## Requisitos de armazenamento de arquivo V12.4

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.4.1** | Verifique se os arquivos obtidos de fontes não confiáveis ​​são armazenados fora da raiz da web, com permissões limitadas, de preferência com validação forte. | ✓ | ✓ | ✓ | 922 |
| **12.4.2** | Verifique se os arquivos obtidos de fontes não confiáveis ​​são verificados por antivírus para evitar o upload de conteúdo malicioso conhecido. | ✓ | ✓ | ✓ | 509 |

## Requisitos de download de arquivo V12.5

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.5.1** | Verifique se a camada da web está configurada para servir apenas arquivos com extensões de arquivo específicas para evitar informações não intencionais e vazamento de código-fonte. Por exemplo, arquivos de backup (por exemplo, .bak), arquivos de trabalho temporários (por exemplo .swp), arquivos compactados (.zip, .tar.gz, etc) e outras extensões comumente usadas por editores devem ser bloqueados, a menos que necessário. | ✓ | ✓ | ✓ | 552 |
| **12.5.2** | Verifique se as solicitações diretas para arquivos carregados nunca serão executadas como conteúdo HTML / JavaScript. | ✓ | ✓ | ✓ | 434 |

## Requisitos de proteção V12.6 SSRF

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **12.6.1** | Verifique se o servidor da web ou de aplicativos está configurado com uma lista de permissões de recursos ou sistemas para os quais o servidor pode enviar solicitações ou carregar dados / arquivos. | ✓ | ✓ | ✓ | 918 |

## Referências

Para obter mais informações, consulte também:

* [Manuseio de extensão de arquivo para informações confidenciais](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload)
* [Download do arquivo reflexivo por Oren Hafif](https://www.trustwave.com/Resources/SpiderLabs-Blog/Reflected-File-Download---A-New-Web-Attack-Vector/)
* [Folha de dicas de gerenciamento de JavaScript de terceiros OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Third_Party_Javascript_Management_Cheat_Sheet.html)

# V13: Requisitos de verificação de API e serviço da Web

## Objetivo de Controle

Certifique-se de que um aplicativo verificado que usa APIs de camada de serviço confiável (geralmente usando JSON ou XML ou GraphQL) tem:

* Autenticação adequada, gerenciamento de sessão e autorização de todos os serviços da web.
* Validação de entrada de todos os parâmetros que transitam de um nível de confiança inferior para o superior.
* Controles de segurança eficazes para todos os tipos de API, incluindo nuvem e API sem servidor

Leia este capítulo em combinação com todos os outros capítulos do mesmo nível; não duplicamos mais questões de autenticação ou gerenciamento de sessão de API.

## Requisitos de verificação de segurança de serviço da web genérico V13.1

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **13.1.1** | Verifique se todos os componentes do aplicativo usam as mesmas codificações e analisadores para evitar ataques de análise que exploram URI diferente ou comportamento de análise de arquivo que pode ser usado em ataques SSRF e RFI. | ✓ | ✓ | ✓ | 116 |
| **13.1.2** | Verifique se o acesso às funções de administração e gerenciamento é limitado a administradores autorizados. | ✓ | ✓ | ✓ | 419 |
| **13.1.3** | Verifique se os URLs da API não expõem informações confidenciais, como a chave da API, tokens de sessão etc. | ✓ | ✓ | ✓ | 598 |
| **13.1.4** | Verifique se as decisões de autorização são feitas no URI, reforçadas pela segurança programática ou declarativa no controlador ou roteador, e no nível do recurso, reforçadas por permissões baseadas em modelo. |  | ✓ | ✓ | 285 |
| **13.1.5** | Verifique se as solicitações contendo tipos de conteúdo inesperados ou ausentes são rejeitadas com os cabeçalhos apropriados (status de resposta HTTP 406 inaceitável ou 415 tipo de mídia não suportado). |  | ✓ | ✓ | 434 |

## Requisitos de verificação de serviço da Web RESTful V13.2

A validação do esquema JSON está em um estágio de padronização de rascunho (consulte as referências). Ao considerar o uso da validação de esquema JSON, que é a prática recomendada para serviços da Web RESTful, considere o uso dessas estratégias adicionais de validação de dados em combinação com a validação de esquema JSON:

* Validação de análise do objeto JSON, como se houver elementos ausentes ou extras.
* Validação dos valores do objeto JSON usando métodos de validação de entrada padrão, como tipo de dados, formato de dados, comprimento, etc.
* e validação de esquema JSON formal.

Assim que o padrão de validação do esquema JSON for formalizado, o ASVS atualizará seu conselho nesta área. Monitore cuidadosamente todas as bibliotecas de validação de esquema JSON em uso, pois elas precisarão ser atualizadas regularmente até que o padrão seja formalizado e os bugs eliminados das implementações de referência.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **13.2.1** | Verifique se os métodos HTTP RESTful ativados são uma escolha válida para o usuário ou ação, como impedir que usuários normais usem DELETE ou PUT em API ou recursos protegidos. | ✓ | ✓ | ✓ | 650 |
| **13.2.2** | Verifique se a validação do esquema JSON está em vigor e verificada antes de aceitar a entrada. | ✓ | ✓ | ✓ | 20 |
| **13.2.3** | Verifique se os serviços da web RESTful que utilizam cookies estão protegidos contra Cross-Site Request Forgery por meio do uso de pelo menos um ou mais dos seguintes: padrão de cookie de envio duplo, nonces CSRF ou verificações de cabeçalho de solicitação de origem. | ✓ | ✓ | ✓ | 352 |
| **13.2.4** | Verifique se os serviços REST têm controles anti-automação para proteger contra chamadas excessivas, especialmente se a API não estiver autenticada. |  | ✓ | ✓ | 770 |
| **13.2.5** | Verifique se os serviços REST verificam explicitamente o tipo de conteúdo de entrada como o esperado, como application / xml ou application / json. |  | ✓ | ✓ | 436 |
| **13.2.6** | Verifique se os cabeçalhos da mensagem e a carga útil são confiáveis ​​e não foram modificados em trânsito. Exigir criptografia forte para transporte (somente TLS) pode ser suficiente em muitos casos, pois fornece proteção de confidencialidade e integridade. As assinaturas digitais por mensagem podem fornecer garantia adicional sobre as proteções de transporte para aplicativos de alta segurança, mas trazem consigo complexidade e riscos adicionais para comparar os benefícios. |  | ✓ | ✓ | 345 |

## Requisitos de verificação de serviço da Web SOAP V13.3

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **13.3.1** | Verifique se a validação do esquema XSD ocorre para garantir um documento XML devidamente formado, seguido pela validação de cada campo de entrada antes que qualquer processamento desses dados ocorra. | ✓ | ✓ | ✓ | 20 |
| **13.3.2** | Verifique se a carga útil da mensagem está assinada usando WS-Security para garantir o transporte confiável entre o cliente e o serviço. |  | ✓ | ✓ | 345 |

Nota: Devido a problemas com ataques XXE contra DTDs, a validação de DTD não deve ser usada e a avaliação de DTD da estrutura desabilitada de acordo com os requisitos definidos na Configuração V14.

## V13.4 GraphQL e outros requisitos de segurança da camada de dados de serviço da web

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **13,4.1** | Verifique se uma lista de permissão de consulta ou uma combinação de limitação de profundidade e limitação de quantidade é usada para evitar GraphQL ou Denial of Service (DoS) de expressão da camada de dados como resultado de consultas aninhadas caras. Para cenários mais avançados, a análise de custo da consulta deve ser usada. |  | ✓ | ✓ | 770 |
| **13,4.2** | Verifique se GraphQL ou outra lógica de autorização da camada de dados deve ser implementada na camada de lógica de negócios em vez da camada GraphQL. |  | ✓ | ✓ | 285 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Serverless Top 10](https://github.com/OWASP/Serverless-Top-10-Project/raw/master/OWASP-Top-10-Serverless-Interpretation-en.pdf)
* [Projeto sem servidor OWASP](https://owasp.org/www-project-serverless-top-10/)
* [OWASP Testing Guide 4.0: Teste de Gerenciamento de Configuração e Implantação](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/02-Configuration_and_Deployment_Management_Testing/README.html)
* [Folha de referências da falsificação de solicitação entre sites OWASP](https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html)
* [Folha de referências da prevenção de entidades externas OWASP XML - Orientação geral](https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html#general-guidance)
* [JSON Web Tokens (e assinatura)](https://jwt.io/)
* [Folha de referências de segurança REST](https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html)
* [Esquema JSON](https://json-schema.org/specification.html)
* [Ataques de entidade XML DTD](https://www.vsecurity.com/download/publications/XMLDTDEntityAttacks.pdf)
* [Orange Tsai - uma nova era de SSRF, explorando o analisador de URL em linguagens de programação em alta](https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf)

# V14: Requisitos de verificação de configuração

## Objetivo de Controle

Certifique-se de que um aplicativo verificado tenha:

* Um ambiente de compilação seguro, repetível e automatizado.
* Gerenciamento reforçado de biblioteca, dependência e configuração de terceiros, de forma que componentes desatualizados ou inseguros não sejam incluídos no aplicativo.
* Uma configuração segura por padrão, de forma que administradores e usuários tenham que enfraquecer a postura de segurança padrão.

A configuração do aplicativo pronto para uso deve ser segura para estar na Internet, o que significa uma configuração segura para uso imediato.

## Versão V14.1

Pipelines de construção são a base para segurança repetível - toda vez que algo inseguro é descoberto, pode ser resolvido no código-fonte, scripts de construção ou implantação e testado automaticamente. Estamos encorajando fortemente o uso de pipelines de construção com segurança automática e verificações de dependência que avisam ou interrompem a construção para evitar que problemas de segurança conhecidos sejam implantados na produção. Etapas manuais executadas de forma irregular levam diretamente a erros de segurança evitáveis.

À medida que o setor muda para um modelo DevSecOps, é importante garantir a disponibilidade e integridade contínuas de implantação e configuração para atingir um estado "bom conhecido". No passado, se um sistema fosse hackeado, levaria dias ou meses para provar que nenhuma outra invasão havia ocorrido. Hoje, com o advento da infraestrutura definida por software, implantações A / B rápidas com tempo de inatividade zero e compilações automatizadas em contêineres, é possível construir, fortalecer e implantar automaticamente e continuamente um substituto "bom" para qualquer sistema comprometido.

Se os modelos tradicionais ainda estiverem em vigor, etapas manuais devem ser executadas para fortalecer e fazer backup dessa configuração para permitir que os sistemas comprometidos sejam rapidamente substituídos por sistemas não comprometidos de alta integridade em tempo hábil.

A conformidade com esta seção requer um sistema de construção automatizado e acesso a scripts de construção e implantação.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **14.1.1** | Verifique se os processos de construção e implantação de aplicativos são executados de maneira segura e repetível, como automação de CI / CD, gerenciamento de configuração automatizado e scripts de implantação automatizados. |  | ✓ | ✓ |  |
| **14.1.2** | Verifique se os sinalizadores do compilador estão configurados para habilitar todas as proteções e avisos de estouro de buffer disponíveis, incluindo randomização de pilha, prevenção de execução de dados e para interromper a compilação se um ponteiro inseguro, memória, string de formato, inteiro ou operações de string forem encontrados. |  | ✓ | ✓ | 120 |
| **14.1.3** | Verifique se a configuração do servidor é reforçada de acordo com as recomendações do servidor de aplicativos e das estruturas em uso. |  | ✓ | ✓ | 16 |
| **14,1.4** | Verifique se o aplicativo, a configuração e todas as dependências podem ser reimplantadas usando scripts de implantação automatizados, criados a partir de um runbook documentado e testado em um tempo razoável ou restaurados de backups em tempo hábil. |  | ✓ | ✓ |  |
| **14,1.5** | Verifique se os administradores autorizados podem verificar a integridade de todas as configurações relevantes para a segurança para detectar adulteração. |  |  | ✓ |  |

## Dependência V14.2

O gerenciamento de dependências é crítico para a operação segura de qualquer aplicativo de qualquer tipo. A falha em se manter atualizado com dependências desatualizadas ou inseguras é a causa raiz dos maiores e mais caros ataques até hoje.

Nota: No Nível 1, a conformidade 14.2.1 está relacionada a observações ou detecções do lado do cliente e outras bibliotecas e componentes, em vez da análise de código estático ou análise de dependência mais precisa em tempo de construção. Essas técnicas mais precisas podem ser descobertas por entrevista, conforme necessário.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **14.2.1** | Verifique se todos os componentes estão atualizados, de preferência usando um verificador de dependência durante a construção ou compilação. ([C2](https://owasp.org/www-project-proactive-controls/#div-numbering)) | ✓ | ✓ | ✓ | 1026 |
| **14.2.2** | Verifique se todos os recursos, documentação, amostras e configurações desnecessários foram removidos, como aplicativos de amostra, documentação da plataforma e usuários padrão ou de exemplo. | ✓ | ✓ | ✓ | 1002 |
| **14.2.3** | Verifique se os ativos do aplicativo, como bibliotecas JavaScript, CSS ou fontes da web, são hospedados externamente em uma Content Delivery Network (CDN) ou provedor externo, Subresource Integrity (SRI) é usado para validar a integridade do ativo. | ✓ | ✓ | ✓ | 829 |
| **14.2.4** | Verifique se os componentes de terceiros vêm de repositórios predefinidos, confiáveis ​​e mantidos continuamente. ([C2](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 829 |
| **14.2.5** | Verifique se um catálogo de inventário é mantido de todas as bibliotecas de terceiros em uso. ([C2](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ |  |
| **14.2.6** | Verifique se a superfície de ataque é reduzida por sandbox ou encapsulamento de bibliotecas de terceiros para expor apenas o comportamento necessário no aplicativo. ([C2](https://owasp.org/www-project-proactive-controls/#div-numbering)) |  | ✓ | ✓ | 265 |

## V14.3 Requisitos de divulgação de segurança não intencionais

As configurações de produção devem ser reforçadas para proteger contra ataques comuns, como consoles de depuração, aumentar o nível de ataques de Cross-site Scripting (XSS) e Inclusão de arquivo remoto (RFI) e para eliminar "vulnerabilidades" de descoberta de informações triviais que são indesejáveis marca registrada de muitos relatórios de testes de penetração. Muitos desses problemas raramente são classificados como um risco significativo, mas estão encadeados com outras vulnerabilidades. Se esses problemas não estiverem presentes por padrão, ele aumentará o nível antes que a maioria dos ataques seja bem-sucedida.

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **14.3.1** | Verifique se as mensagens de erro do servidor da web ou de aplicativos e da estrutura estão configuradas para fornecer respostas personalizadas e acionáveis ​​ao usuário para eliminar quaisquer divulgações de segurança não intencionais. | ✓ | ✓ | ✓ | 209 |
| **14.3.2** | Verifique se os modos de depuração do servidor da web ou do aplicativo e da estrutura do aplicativo estão desativados na produção para eliminar recursos de depuração, consoles de desenvolvedor e divulgações de segurança não intencionais. | ✓ | ✓ | ✓ | 497 |
| **14,3.3** | Verifique se os cabeçalhos HTTP ou qualquer parte da resposta HTTP não expõem informações detalhadas da versão dos componentes do sistema. | ✓ | ✓ | ✓ | 200 |

## 

## Requisitos de cabeçalhos de segurança HTTP V14.4

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **14,4.1** | Verifique se cada resposta HTTP contém um cabeçalho Content-Type. Os tipos de conteúdo text / \*, / + xml e application / xml também devem especificar um conjunto de caracteres seguro (por exemplo, UTF-8, ISO-8859-1). | ✓ | ✓ | ✓ | 173 |
| **14,4.2** | Verifique se todas as respostas da API contêm um Content-Disposition: attachment; filename = cabeçalho "api.json" (ou outro nome de arquivo apropriado para o tipo de conteúdo). | ✓ | ✓ | ✓ | 116 |
| **14,4.3** | Verifique se um cabeçalho de resposta da Política de Segurança de Conteúdo (CSP) está em vigor para ajudar a mitigar o impacto de ataques XSS como HTML, DOM, JSON e vulnerabilidades de injeção de JavaScript. | ✓ | ✓ | ✓ | 1021 |
| **14,4.4** | Verifique se todas as respostas contêm um cabeçalho X-Content-Type-Options: nosniff. | ✓ | ✓ | ✓ | 116 |
| **14,4.5** | Verifique se um cabeçalho Strict-Transport-Security está incluído em todas as respostas e para todos os subdomínios, como Strict-Transport-Security: max-age = 15724800; includeSubdomains. | ✓ | ✓ | ✓ | 523 |
| **14,4.6** | Verifique se um cabeçalho "Referrer-Policy" adequado está incluído, como "no-referrer" ou "same-origin". | ✓ | ✓ | ✓ | 116 |
| **14,4.7** | Verifique se o conteúdo de um aplicativo da web não pode ser incorporado em um site de terceiros por padrão e se a incorporação dos recursos exatos só é permitida quando necessário, usando a Política de Segurança de Conteúdo: ancestrais de quadros e resposta de Opções de X-Frame cabeçalhos. | ✓ | ✓ | ✓ | 346 |

## V14.5 Validar Requisitos de Cabeçalho de Solicitação HTTP

| # | Descrição | L1 | L2 | L3 | CWE |
| --- | --- | --- | --- | --- | --- |
| **14.5.1** | Verifique se o servidor de aplicativos aceita apenas os métodos HTTP em uso pelo aplicativo / API, incluindo OPÇÕES pré-voo e logs / alertas em quaisquer solicitações que não sejam válidas para o contexto do aplicativo. | ✓ | ✓ | ✓ | 749 |
| **14.5.2** | Verifique se o cabeçalho Origin fornecido não é usado para autenticação ou decisões de controle de acesso, pois o cabeçalho Origin pode ser facilmente alterado por um invasor. | ✓ | ✓ | ✓ | 346 |
| **14.5.3** | Verifique se o cabeçalho de compartilhamento de recursos de origem cruzada (CORS) Access-Control-Allow-Origin usa uma lista de permissões restrita de domínios e subdomínios confiáveis ​​para correspondência e não oferece suporte à origem "nula". | ✓ | ✓ | ✓ | 346 |
| **14.5.4** | Verifique se os cabeçalhos HTTP adicionados por um proxy confiável ou dispositivos SSO, como um token do portador, são autenticados pelo aplicativo. |  | ✓ | ✓ | 306 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Web Security Testing Guide 4.1: Teste de violação de verbo HTTP](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/03-Testing_for_HTTP_Verb_Tampering.html)
* Adicionar Content-Disposition às respostas da API ajuda a evitar muitos ataques baseados em mal-entendidos sobre o tipo de MIME entre o cliente e o servidor, e a opção "nome do arquivo" ajuda especificamente a prevenir [Ataques de download de arquivo refletido.](https://www.blackhat.com/docs/eu-14/materials/eu-14-Hafif-Reflected-File-Download-A-New-Web-Attack-Vector.pdf)
* [Folha de referências da política de segurança de conteúdo](https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html)
* [Explorando a configuração incorreta do CORS para BitCoins e Bounties](https://portswigger.net/blog/exploiting-cors-misconfigurations-for-bitcoins-and-bounties)
* [OWASP Web Security Testing Guide 4.1: Teste de Gerenciamento de Configuração e Implantação](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/02-Configuration_and_Deployment_Management_Testing/README.html)
* [Sandboxing de componentes de terceiros](https://cheatsheetseries.owasp.org/cheatsheets/Third_Party_Javascript_Management_Cheat_Sheet.html#sandboxing-content)

# Apêndice A: Glossário

* **Randomização do Layout do Espaço de Endereço** (ASLR) - Uma técnica para dificultar a exploração de bugs de corrupção de memória.
* **Lista de permissão** - Uma lista de dados ou operações permitidas, por exemplo, uma lista de caracteres que têm permissão para realizar a validação de entrada.
* **Segurança do aplicativo** - A segurança em nível de aplicativo se concentra na análise de componentes que compõem a camada de aplicativo do Modelo de Referência de Interconexão de Sistemas Abertos (Modelo OSI), em vez de focar, por exemplo, no sistema operacional subjacente ou nas redes conectadas.
* **Verificação de segurança do aplicativo** - A avaliação técnica de um aplicativo contra o OWASP ASVS.
* **Relatório de verificação de segurança do aplicativo** - Um relatório que documenta os resultados gerais e a análise de apoio produzida pelo verificador para uma aplicação específica.
* **Autenticação** - A verificação da identidade reivindicada de um usuário do aplicativo.
* **Verificação Automatizada** - O uso de ferramentas automatizadas (ferramentas de análise dinâmica, ferramentas de análise estática ou ambas) que usam assinaturas de vulnerabilidade para localizar problemas.
* **Teste de caixa preta** - É um método de teste de software que examina a funcionalidade de um aplicativo sem examinar suas estruturas internas ou funcionamento.
* **Componente** - uma unidade de código independente, com disco associado e interfaces de rede que se comunicam com outros componentes.
* **Cross-Site Scripting** (XSS) - Uma vulnerabilidade de segurança normalmente encontrada em aplicativos da web, permitindo a injeção de scripts do lado do cliente no conteúdo.
* **Módulo criptográfico** - Hardware, software e / ou firmware que implementa algoritmos criptográficos e / ou gera chaves criptográficas.
* **Enumeração de fraqueza comum**(CWE) - Uma lista desenvolvida pela comunidade de pontos fracos comuns de segurança de software. Ele serve como uma linguagem comum, uma medida de medição para ferramentas de segurança de software e como uma linha de base para os esforços de identificação, mitigação e prevenção de fraquezas.
* **Verificação de Design** - A avaliação técnica da arquitetura de segurança de um aplicativo.
* **Teste de segurança de aplicativo dinâmico** (DAST) - As tecnologias são projetadas para detectar condições indicativas de uma vulnerabilidade de segurança em um aplicativo em seu estado de execução.
* **Verificação Dinâmica** - O uso de ferramentas automatizadas que usam assinaturas de vulnerabilidade para localizar problemas durante a execução de um aplicativo.
* **Fast IDentity Online** (FIDO) - Um conjunto de padrões de autenticação que permite que uma variedade de métodos de autenticação diferentes sejam usados, incluindo biometria, Módulos de plataforma confiáveis ​​(TPMs), tokens de segurança USB, etc.
* **Identificador Globalmente Único** (GUID) - um número de referência exclusivo usado como um identificador no software.
* **Protocolo de Transferência de Hipertexto**(HTTPS) - Um protocolo de aplicação para sistemas de informação hipermídia distribuídos e colaborativos. É a base da comunicação de dados para a World Wide Web.
* **Chaves codificadas** - Chaves criptográficas que são armazenadas no sistema de arquivos, seja em código, comentários ou arquivos.
* **Módulo de Segurança de Hardware** (HSM) - Componente de hardware capaz de armazenar chaves criptográficas e outros segredos de maneira protegida.
* **Linguagem de consulta do Hibernate** (HQL) - Uma linguagem de consulta que é semelhante em aparência ao SQL usado pela biblioteca Hibernate ORM.
* **Validação de entrada** - A canonização e validação de entrada de usuário não confiável.
* **Código malicioso**- Código introduzido em um aplicativo durante seu desenvolvimento sem o conhecimento do proprietário do aplicativo, o que contorna a política de segurança pretendida do aplicativo. Não é o mesmo que malware, como vírus ou worm!
* **Malware** - Código executável que é introduzido em um aplicativo durante o tempo de execução sem o conhecimento do usuário ou administrador do aplicativo.
* **Abra o Projeto de Segurança de Aplicativos da Web**(OWASP) - O Open Web Application Security Project (OWASP) é uma comunidade mundial gratuita e aberta com foco na melhoria da segurança do software aplicativo. Nossa missão é tornar a segurança do aplicativo "visível", para que as pessoas e organizações possam tomar decisões informadas sobre os riscos à segurança do aplicativo. Ver:<https://www.owasp.org/>
* **Senha de uso único** (OTP) - Uma senha gerada exclusivamente para ser usada em uma única ocasião.
* **Mapeamento Objeto Relacional** (ORM) - Um sistema usado para permitir que um banco de dados relacional / baseado em tabela seja referenciado e consultado em um programa de aplicativo usando um modelo de objeto compatível com o aplicativo.
* **Função 2 de derivação de chave baseada em senha** (PBKDF2) - Um algoritmo unilateral especial usado para criar uma chave criptográfica forte a partir de um texto de entrada (como uma senha) e um valor salt aleatório adicional e pode, portanto, ser usado para tornar mais difícil quebrar uma senha offline se o valor resultante é armazenado em vez da senha original.
* **Informação pessoalmente identificável** (PII) - são informações que podem ser usadas sozinhas ou com outras informações para identificar, contatar ou localizar uma única pessoa, ou para identificar um indivíduo no contexto.
* **Executável independente de posição** (PIE) - Um corpo de código de máquina que, sendo colocado em algum lugar da memória primária, é executado corretamente, independentemente de seu endereço absoluto.
* **Infraestrutura de chave pública**(PKI) - Um arranjo que vincula as chaves públicas às respectivas identidades de entidades. A vinculação é estabelecida por meio de um processo de registro e emissão de certificados em e por uma autoridade de certificação (CA).
* **Rede telefónica pública comutada** (PSTN) - A rede de telefonia tradicional, incluindo telefones fixos e celulares.
* **Parte Confiante**(RP) - geralmente um aplicativo que depende de um usuário ter se autenticado em um provedor de autenticação separado. O aplicativo depende de algum tipo de token ou conjunto de asserções assinadas fornecido por esse provedor de autenticação para confiar que o usuário é quem diz ser.
* **Teste de segurança de aplicativo estático**(SAST) - Um conjunto de tecnologias projetado para analisar o código-fonte do aplicativo, código de byte e binários para codificação e condições de design que são indicativas de vulnerabilidades de segurança. As soluções SAST analisam um aplicativo de "dentro para fora" em um estado de não execução.
* **Ciclo de vida de desenvolvimento de software** (SDLC) - O processo passo a passo pelo qual o software é desenvolvido, desde os requisitos iniciais até a implantação e manutenção.
* **Arquitetura de Segurança** - Uma abstração do design de um aplicativo que identifica e descreve onde e como os controles de segurança são usados, e também identifica e descreve a localização e a sensibilidade dos dados do usuário e do aplicativo.
* **Configuração de Segurança** - A configuração de tempo de execução de um aplicativo que afeta como os controles de segurança são usados.
* **Controle de segurança** - Uma função ou componente que executa uma verificação de segurança (por exemplo, uma verificação de controle de acesso) ou quando chamado resulta em um efeito de segurança (por exemplo, gerando um registro de auditoria).
* **Falsificação de solicitação do lado do servidor** (SSRF) - Um ataque que abusa da funcionalidade do servidor para ler ou atualizar recursos internos, fornecendo ou modificando uma URL para a qual o código em execução no servidor lerá ou enviará dados.
* **Autenticação de logon único**(SSO) - Isso ocorre quando um usuário efetua login em um aplicativo e, em seguida, é automaticamente conectado a outros aplicativos sem ter que se autenticar novamente. Por exemplo, quando você faz login no Google, ao acessar outros serviços do Google, como YouTube, Google Docs e Gmail, você será automaticamente conectado.
* **Injeção SQL** (SQLi) - Uma técnica de injeção de código usada para atacar aplicativos baseados em dados, nos quais instruções SQL maliciosas são inseridas em um ponto de entrada.
* **SVG** - Gráficos vetoriais escaláveis
* **OTP baseado em tempo** - Um método de geração de OTP em que a hora atual atua como parte do algoritmo para gerar a senha.
* **Modelagem de Ameaças** - Uma técnica que consiste no desenvolvimento de arquiteturas de segurança cada vez mais refinadas para identificar agentes de ameaça, zonas de segurança, controles de segurança e ativos técnicos e comerciais importantes.
* **Segurança da Camada de Transporte** (TLS) - protocolos criptográficos que fornecem segurança de comunicação em uma conexão de rede
* **Módulo de plataforma confiável** (TPM) - Um tipo de HSM que geralmente é conectado a um componente de hardware maior, como uma placa-mãe, e atua como a "raiz de confiança" desse sistema.
* **Autenticação de dois fatores** (2FA) - Isso adiciona um segundo nível de autenticação a um login de conta.
* **2º Fator Universal** (U2F) - Um dos padrões criados pela FIDO especificamente para permitir que uma chave de segurança USB ou NFC seja usada como um segundo fator de autenticação.
* **URI / URL / fragmentos de URL**- Um Uniform Resource Identifier é uma sequência de caracteres usada para identificar um nome ou um recurso da web. Um Localizador Uniforme de Recursos é freqüentemente usado como referência a um recurso.
* **Verificador** - A pessoa ou equipe que está analisando um aplicativo em relação aos requisitos do OWASP ASVS.
* **O que você vê é o que você obtém** (WYSIWYG) - Um tipo de editor de conteúdo avançado que mostra como o conteúdo realmente aparecerá quando renderizado, em vez de mostrar a codificação usada para controlar a renderização.
* **Certificado X.509** - Um certificado X.509 é um certificado digital que usa o padrão internacionalmente aceito de infraestrutura de chave pública (PKI) X.509 para verificar se uma chave pública pertence ao usuário, computador ou identidade de serviço contida no certificado.
* **XML eXternal Entity**(XXE) - Um tipo de entidade XML que pode acessar conteúdo local ou remoto por meio de um identificador de sistema declarado. Isso pode levar a vários ataques de injeção.

# Apêndice B: Referências

Os seguintes projetos OWASP são provavelmente úteis para usuários / adotantes deste padrão:

## Projetos Centrais OWASP

1. OWASP Top 10 Project: <https://owasp.org/www-project-top-ten/>
2. OWASP Web Security Testing Guide: <https://owasp.org/www-project-web-security-testing-guide/>
3. Controles proativos OWASP: <https://owasp.org/www-project-proactive-controls/>
4. Estrutura de Conhecimento de Segurança OWASP: <https://owasp.org/www-project-security-knowledge-framework/>
5. OWASP Software Assurance Maturity Model (SAMM): <https://owasp.org/www-project-samm/>

## Projeto OWASP Cheat Sheet Series

[Este projeto](https://owasp.org/www-project-cheat-sheets/) tem uma série de folhas de dicas que serão relevantes para diferentes tópicos do ASVS.

Há um mapeamento para o ASVS que pode ser encontrado aqui: <https://cheatsheetseries.owasp.org/cheatsheets/IndexASVS.html>

## Projetos relacionados à segurança móvel

1. Projeto de Segurança Móvel OWASP: <https://owasp.org/www-project-mobile-security/>
2. Os 10 principais riscos do OWASP Mobile: <https://owasp.org/www-project-mobile-top-10/>
3. OWASP Mobile Security Testing Guide e Mobile Application Security Verification Standard: <https://owasp.org/www-project-mobile-security-testing-guide/>

## Projetos relacionados à Internet das Coisas da OWASP

1. Projeto OWASP Internet of Things: <https://owasp.org/www-project-internet-of-things/>

## Projetos sem servidor OWASP

1. Projeto sem servidor OWASP: <https://owasp.org/www-project-serverless-top-10/>

## Outras

Da mesma forma, os seguintes sites são provavelmente úteis para usuários / adotantes deste padrão

1. SecLists Github: <https://github.com/danielmiessler/SecLists>
2. Enumeração de fraqueza comum do MITER: <https://cwe.mitre.org/>
3. Conselho de padrões de segurança PCI: <https://www.pcisecuritystandards.org>
4. Requisitos do PCI Data Security Standard (DSS) v3.2.1 e procedimentos de avaliação de segurança: <https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf>
5. Estrutura de segurança de software PCI - Requisitos de software e procedimentos de avaliação seguros: <https://www.pcisecuritystandards.org/documents/PCI-Secure-Software-Standard-v1_0.pdf>
6. Requisitos e procedimentos de avaliação do PCI Secure Software Lifecycle (Secure SLC): <https://www.pcisecuritystandards.org/documents/PCI-Secure-SLC-Standard-v1_0.pdf>

# Apêndice C: Requisitos de verificação da Internet das coisas

Esta seção estava originalmente no branch principal, mas com o trabalho que a equipe OWASP IoT fez, não faz sentido manter dois tópicos diferentes sobre o assunto. Para a versão 4.0, estamos movendo isso para o Apêndice e exortamos todos os que precisam, em vez de usar o principal[Projeto OWASP IoT](https://owasp.org/www-project-internet-of-things/)

## Objetivo de Controle

Dispositivos incorporados / IoT devem:

* Tenha o mesmo nível de controles de segurança dentro do dispositivo que os encontrados no servidor, reforçando os controles de segurança em um ambiente confiável.
* Os dados confidenciais armazenados no dispositivo devem ser feitos de maneira segura, usando armazenamento com suporte de hardware, como elementos seguros.
* Todos os dados confidenciais transmitidos do dispositivo devem utilizar a segurança da camada de transporte.

## Requisitos de verificação de segurança

| # | Descrição | L1 | L2 | L3 | Desde |
| --- | --- | --- | --- | --- | --- |
| **C.1** | Verifique se as interfaces de depuração da camada de aplicativo, como USB, UART e outras variantes seriais, estão desabilitadas ou protegidas por uma senha complexa. | ✓ | ✓ | ✓ | 4,0 |
| **C.2** | Verifique se as chaves criptográficas e os certificados são exclusivos para cada dispositivo individual. | ✓ | ✓ | ✓ | 4,0 |
| **C.3** | Verifique se os controles de proteção de memória, como ASLR e DEP, estão ativados pelo sistema operacional incorporado / IoT, se aplicável. | ✓ | ✓ | ✓ | 4,0 |
| **C.4** | Verifique se as interfaces de depuração no chip, como JTAG ou SWD, estão desabilitadas ou se o mecanismo de proteção disponível está habilitado e configurado apropriadamente. | ✓ | ✓ | ✓ | 4,0 |
| **C.5** | Verifique se a execução confiável está implementada e habilitada, se disponível no dispositivo SoC ou CPU. | ✓ | ✓ | ✓ | 4,0 |
| **C.6** | Verifique se os dados confidenciais, chaves privadas e certificados são armazenados com segurança em um Elemento Seguro, TPM, TEE (Trusted Execution Environment) ou protegidos por criptografia forte. | ✓ | ✓ | ✓ | 4,0 |
| **C.7** | Verifique se os aplicativos de firmware protegem os dados em trânsito usando a segurança da camada de transporte. | ✓ | ✓ | ✓ | 4,0 |
| **C.8** | Verifique se os aplicativos de firmware validam a assinatura digital das conexões do servidor. | ✓ | ✓ | ✓ | 4,0 |
| **C.9** | Verifique se as comunicações sem fio são autenticadas mutuamente. | ✓ | ✓ | ✓ | 4,0 |
| **C.10** | Verifique se as comunicações sem fio são enviadas por um canal criptografado. | ✓ | ✓ | ✓ | 4,0 |
| **C.11** | Verifique se qualquer uso de funções C proibidas é substituído pelas funções equivalentes seguras apropriadas. | ✓ | ✓ | ✓ | 4,0 |
| **C.12** | Verifique se cada firmware mantém uma lista de materiais de software catalogando componentes de terceiros, controle de versão e vulnerabilidades publicadas. | ✓ | ✓ | ✓ | 4,0 |
| **C.13** | Verifique se todo o código, incluindo binários de terceiros, bibliotecas e estruturas são revisados ​​para credenciais codificadas permanentemente (backdoors). | ✓ | ✓ | ✓ | 4,0 |
| **C.14** | Verifique se os componentes do aplicativo e do firmware não são suscetíveis à injeção de comando do sistema operacional invocando wrappers de comando shell, scripts ou se os controles de segurança evitam a injeção de comando do sistema operacional. | ✓ | ✓ | ✓ | 4,0 |
| **C.15** | Verifique se os aplicativos de firmware fixam a assinatura digital em servidores confiáveis. |  | ✓ | ✓ | 4,0 |
| **C.16** | Verifique a presença de resistência à violação e / ou recursos de detecção de violação. |  | ✓ | ✓ | 4,0 |
| **C.17** | Verifique se todas as tecnologias de proteção de propriedade intelectual fornecidas pelo fabricante do chip estão ativadas. |  | ✓ | ✓ | 4,0 |
| **C.18** | Verifique se os controles de segurança estão em vigor para impedir a engenharia reversa do firmware (por exemplo, remoção de símbolos de depuração detalhados). |  | ✓ | ✓ | 4,0 |
| **C.19** | Verifique se o dispositivo valida a assinatura da imagem de inicialização antes de carregar. |  | ✓ | ✓ | 4,0 |
| **C 20** | Verifique se o processo de atualização do firmware não é vulnerável a ataques de tempo de verificação versus tempo de uso. |  | ✓ | ✓ | 4,0 |
| **C.21** | Verifique se o dispositivo usa assinatura de código e valida os arquivos de atualização do firmware antes de instalar. |  | ✓ | ✓ | 4,0 |
| **C.22** | Verifique se o dispositivo não pode ser desatualizado para versões antigas (anti-rollback) de firmware válido. |  | ✓ | ✓ | 4,0 |
| **C.23** | Verifique o uso de gerador de número pseudo-aleatório criptograficamente seguro no dispositivo embutido (por exemplo, usando geradores de número aleatório fornecidos por chip). |  | ✓ | ✓ | 4,0 |
| **C.24** | Verifique se o firmware pode realizar atualizações automáticas de firmware de acordo com uma programação predefinida. |  | ✓ | ✓ | 4,0 |
| **C.25** | Verifique se o dispositivo limpa o firmware e os dados confidenciais após a detecção de adulteração ou recebimento de mensagem inválida. |  |  | ✓ | 4,0 |
| **C.26** | Verifique se apenas micro controladores que suportam a desativação de interfaces de depuração (por exemplo, JTAG, SWD) são usados. |  |  | ✓ | 4,0 |
| **C.27** | Verifique se apenas micro controladores que fornecem proteção substancial contra ataques de desbloqueio e canal lateral são usados. |  |  | ✓ | 4,0 |
| **C.28** | Verifique se os traços sensíveis não estão expostos às camadas externas da placa de circuito impresso. |  |  | ✓ | 4,0 |
| **C.29** | Verifique se a comunicação entre chips está criptografada (por exemplo, comunicação da placa principal com a placa filha). |  |  | ✓ | 4,0 |
| **C 30** | Verifique se o dispositivo usa assinatura de código e valida o código antes da execução. |  |  | ✓ | 4,0 |
| **C.31** | Verifique se as informações confidenciais mantidas na memória são sobrescritas com zeros assim que não são mais necessárias. |  |  | ✓ | 4,0 |
| **C.32** | Verifique se os aplicativos de firmware utilizam contêineres de kernel para isolamento entre aplicativos. |  |  | ✓ | 4,0 |
| **C.33** | Verifique se os sinalizadores do compilador seguro, como -fPIE, -fstack-protector-all, -Wl, -z, noexecstack, -Wl, -z, noexecheap estão configurados para compilações de firmware. |  |  | ✓ | 4,0 |
| **C.34** | Verifique se os microcontroladores estão configurados com proteção de código (se aplicável). |  |  | ✓ | 4,0 |

## Referências

Para obter mais informações, consulte também:

* [OWASP Internet of Things Top 10](https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf)
* [Projeto de Segurança de Aplicativo Integrado OWASP](https://owasp.org/www-project-embedded-application-security/)
* [Projeto OWASP Internet of Things](https://owasp.org/www-project-internet-of-things/)
* [Ferramenta de proxy TCP Trudy](https://github.com/praetorian-inc/trudy)