

Cinco estratégias simples para proteger APIs

Por Scott Morrison, CA Technologies





Sumário

- 3 O que são APIs e elas compensam o risco?
- 8 Três vetores de ataque contra os quais se precaver
- 13 Cinco estratégias simples de mitigação que você pode ter negligenciado
- 19 Conclusão

O que são APIs e elas compensam o risco?





Como você torna a empresa flexível?

As APIs são uma tecnologia emergente para integrar aplicativos usando a tecnologia da web. Essa abordagem está cada vez mais popular porque ela se baseia em técnicas bastante conhecidas e aproveita a infraestrutura existente.

Mas é um erro pensar que podemos proteger APIs usando os mesmos métodos e a mesma tecnologia que usamos para proteger a web convencional centrada no navegador. Embora seja verdade que as APIs compartilham muitas das mesmas ameaças que infestam a web, elas são fundamentalmente diferentes e têm um perfil de risco totalmente exclusivo que você precisa gerenciar.

Este livreto eletrônico fornece uma visão geral desses novos riscos e oferece cinco soluções simples para contra-atacar as ameaças comuns. Ao adotar uma arquitetura de API segura desde o início, as organizações podem seguir uma estratégia de API com mais segurança e colher os benefícios da integração ágil prometida por essa empolgante nova tecnologia.

O que é uma API?

Desde o início da era da computação, os desenvolvedores lutam para fazer com que os aplicativos se comuniquem. Protocolos especializados, como COM+, CORBA e, até mesmo, SOAP, surgiram ao longo dos anos, mas nenhum era suficiente para atender às necessidades de escalabilidade, simplicidade e funcionalidade entre linguagens.

Mas, na verdade, a resposta estava o tempo todo bem na nossa frente. A World Wide Web foi o primeiro sistema realmente expansível e distribuído que uniu plataformas díspares com um protocolo que era simples de entender e enganosamente poderoso. A ideia crucial era utilizar esse sucesso para integrar aplicativos além do par navegador/servidor web para os quais foi criado.

As APIs são a tecnologia por trás dessa abordagem. As APIs permitem que os desenvolvedores criem uma arquitetura aberta para o compartilhamento de funcionalidades e dados entre aplicativos. **Elas são como janelas que permitem ver dentro de um aplicativo** — um caminho direto para as funcionalidades e os dados principais que residem no núcleo do aplicativo.

As APIs fornecem aos desenvolvedores do lado do cliente — desenvolvedores legítimos e potenciais decodificadores de sistema — um acesso muito mais refinado a um aplicativo do que um aplicativo típico da web. Isso ocorre porque o limite da granularidade de chamadas para camadas do back-end se move das camadas internas relativamente seguras (que residem com segurança em uma DMZ) totalmente para fora, até o aplicativo cliente que reside na internet.

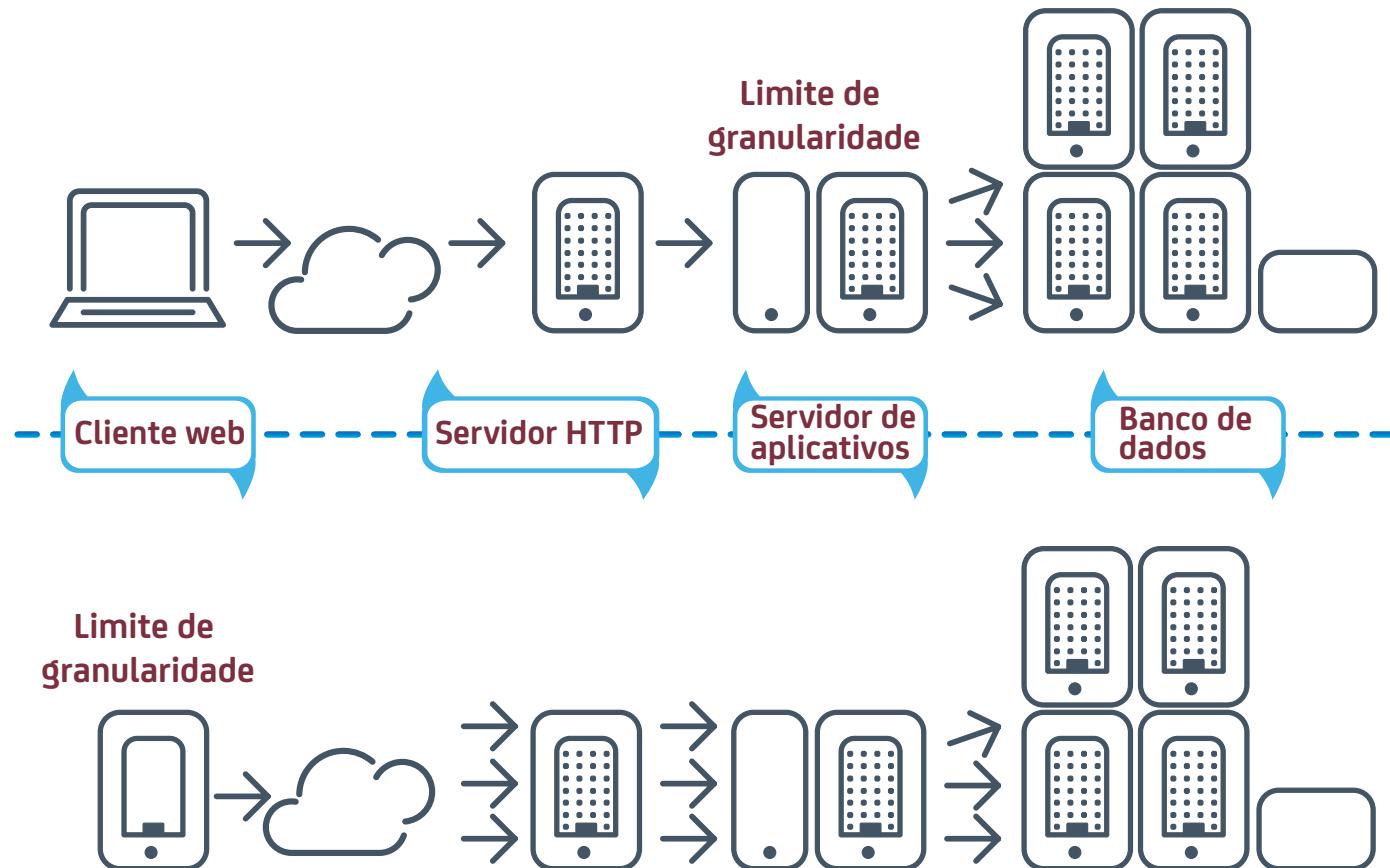
Eu sei, parece ciência aeroespacial. Mas um limite de granularidade simplesmente descreve quanto de um sistema de back-end um aplicativo chamador pode acessar. Infelizmente, a ironia é que as mesmas coisas que tornam as APIs bem-sucedidas também as tornam um alvo perfeito para hackers.



Como as APIs aumentam o risco da organização?

O problema com as APIs é que elas normalmente fornecem um roteiro que descreve a implementação subjacente de um aplicativo — detalhes que, de outra forma, estariam enterrados sob camadas da funcionalidade do aplicativo web. Isso pode dar dicas valiosas aos hackers que levariam a vetores de ataque que, de outra forma, não seriam percebidos. As APIs tendem a ser extremamente claras e autodocumentadas na sua melhor forma, fornecendo uma visão dos objetos internos e até da infraestrutura do banco de dados interno — consistindo em inteligência valiosa para os hackers.

Mas a maior visibilidade não é o único risco introduzido pelas APIs. O aumento do número de chamadas potenciais também aumenta a superfície de ataque, o que significa que um hacker tem mais oportunidades para explorar. Os riscos aumentam com a oportunidade.





Velhos hábitos são difíceis de eliminar

As APIs podem representar um risco maior para a empresa, mas os benefícios potenciais que representam para uma organização podem ofuscar os perigos inerentes. A maior ameaça pode estar na maneira como implementamos as APIs.

Uma API bem-arquitetada permite que as organizações forneçam poderosas ferramentas web diretamente a seus funcionários e clientes. Os bons desenvolvedores de APIs entendem o perfil de ameaça daquilo que estão criando. Infelizmente, muitos desenvolvedores de APIs vêm de um histórico de design da web e podem trazer alguns maus hábitos com eles. É importante reconhecer que, apesar de suas raízes comuns e do compartilhamento da infraestrutura, o design da web e o design de APIs têm objetivos diversos e exigem abordagens diferentes.

Há três vetores de ataque principais em que os hackers se concentram mais frequentemente nas APIs. O entendimento desses vetores ajudará você a criar APIs mais seguras.

Três vetores de ataque contra os quais se precaver





Explicação sobre os vetores

Os vetores de ataque mais comuns podem ser separados em três categorias:

Parâmetros

Ataques a parâmetros exploram os dados enviados para uma API, incluindo URL, parâmetros de consulta, cabeçalhos HTTP e/ou conteúdo de postagens.

Identidades

Ataques a identidades exploram falhas na autenticação, na autorização e no acompanhamento de uma sessão. Especificamente, muitos desses ataques são o resultado de práticas de migração inadequadas do mundo da web para o desenvolvimento de APIs.

Intermediários

Esses ataques interceptam transações legítimas e exploram dados não assinados e/ou não criptografados que estão sendo enviados entre o cliente e o servidor. Podem revelar informações confidenciais (como dados pessoais), alterar uma transação em andamento ou, até mesmo, reproduzir transações legítimas.



Vetor de ataque: *parâmetros*

Os ataques a parâmetros — sendo os mais comuns denominados injeção de SQL — tentam manipular um sistema fornecendo a ele entradas que exploram o comportamento de aplicativos e a infraestrutura que oferece suporte a eles (como bancos de dados).

Normalmente, esses ataques resultam de desenvolvedores que não verificam cuidadosamente a entrada em um aplicativo. E, em contraste com muitos aplicativos web, as APIs normalmente identificam claramente o uso subjacente de um parâmetro por seu nome, oferecendo dicas sedutoras até para o invasor casual.

Os ataques a parâmetros com certeza não são novos; a web é explorada há anos por esse vetor. Mas eles estão aumentando no mundo das APIs porque muitos fornecedores negligenciam o saneamento de entradas, por estarem acostumados a ter isso aplicado automaticamente por muitas estruturas da web. O mesmo risco está associado a APIs, e as mesmas precauções precisam ser tomadas para mitigar essa ameaça.

Vetor de ataque: *identidades*

Todos nós estamos familiarizados com a ideia de identidade de usuário. Mas as APIs também introduzem o conceito de identidade de aplicativo: uma chave que identifica exclusivamente o aplicativo que está chamando uma API. Essa chave, feliz ou infelizmente, é normalmente chamada de chave da API. A chave da API é replicada em cada instância de um aplicativo. Seu objetivo é oferecer suporte ao gerenciamento básico do cliente, como à limitação de taxas, de forma que um aplicativo que se torne viral não possa monopolizar uma API em detrimento de aplicativos menos populares.

Infelizmente, as chaves da API normalmente são tratadas como credenciais confiáveis, o que não são. Normalmente, as chaves da API estão ocultas no código de um aplicativo cliente que faz uma chamada e, apesar dos melhores esforços para escondê-las por parte dos desenvolvedores, elas geralmente são fáceis de ser localizadas e exploradas. A autenticação segura de um aplicativo sempre foi, e continua sendo, um problema difícil de resolver. As APIs trazem isso à tona, mas, infelizmente, há poucas respostas fáceis para esse problema.

A conclusão é que nunca se deve tratar chaves da API como secretas.

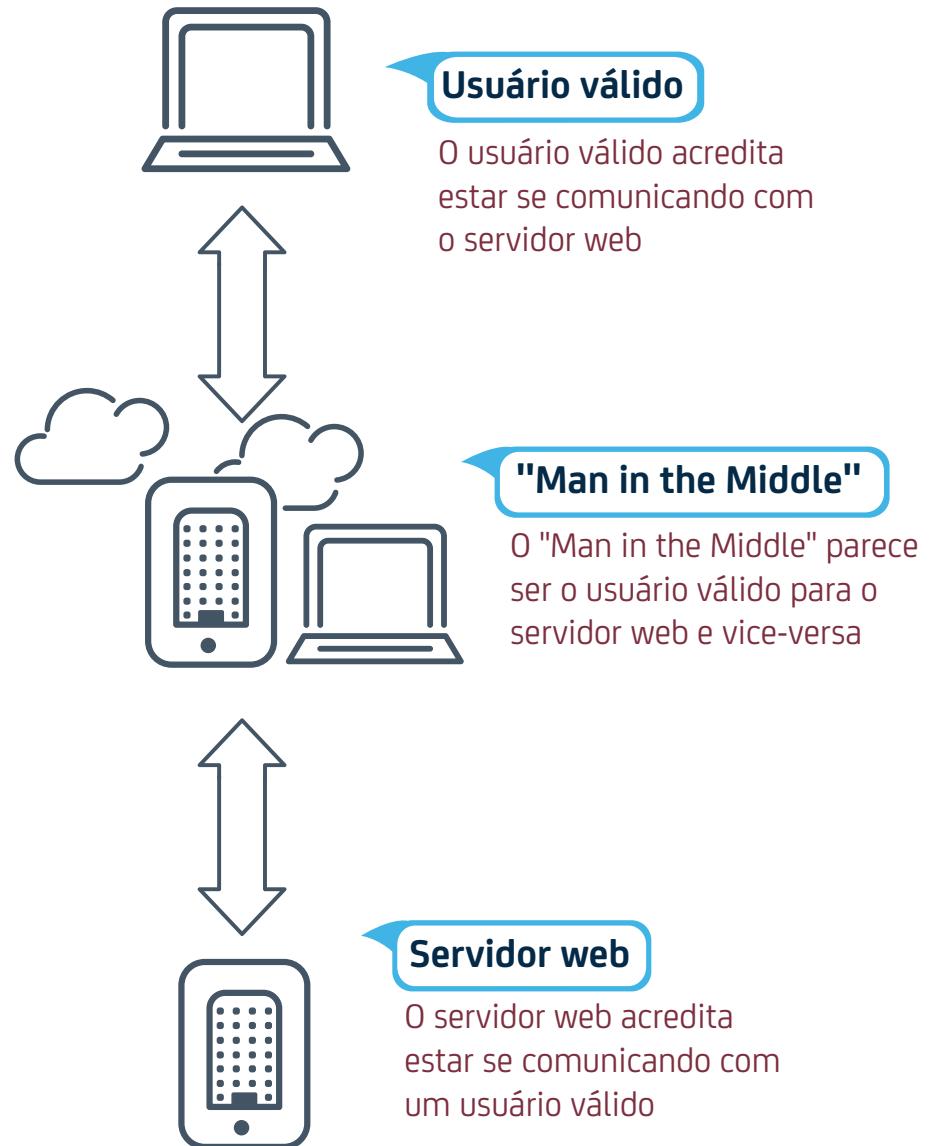


vetor de ataque: *intermediários*

Um ataque de intermediário descreve uma situação em que um invasor se coloca entre o remetente e o receptor de informações.

Ele pode fazer isso de maneira transparente ou pode se colocar explicitamente como uma das partes, mas, nos dois casos, usa essa posição para explorar a troca de dados não assinados ou não criptografados.

APIs que não são configuradas corretamente usando SSL/TLS são altamente vulneráveis a essa forma de ataque. Infelizmente, a cultura no design da web é deixar a maioria das comunicações desprotegida, principalmente por causa dos desafios históricos do dimensionamento de cargas SSL. E mesmo quando o SSL/TLS é aplicado, ele é configurado inadequadamente ou é vulnerável a ataques de downgrade que os tornam ineficazes. No mundo das APIs, há mais em jogo, e a proteção do transporte é essencial para garantir a segurança de dados, sessões e acessos à funcionalidade.



Cinco estratégias simples de mitigação

que permitirão que uma
organização publique APIs
de maneira mais segura

Embora as APIs sejam suscetíveis a vários tipos de ataques, a aplicação de apenas cinco estratégias simples de mitigação permitirá que uma organização publique APIs com segurança.



Estratégia 1:
validar os parâmetros

Estratégia 2:
aplicar detecção explícita
de ameaças

Estratégia 3:
ativar o SSL em todos os lugares

Estratégia 4:
aplicar autenticação
e autorização rigorosas

Estratégia 5:
usar soluções comprovadas



Estratégia 1: validar os parâmetros

A primeira etapa de qualquer implementação resiliente de APIs é sanear todos os dados de entrada para confirmar se são válidos e não provocarão danos. A única defesa mais efetiva contra a manipulação de parâmetros e os ataques de injeção é validar todos os dados de entrada em um esquema rígido — efetivamente uma descrição do que são consideradas entradas permissíveis para o sistema. A validação do esquema deve ser a mais restritiva possível, usando tipagem, intervalos, conjuntos e até listas brancas explícitas sempre que possível. Considere também que os esquemas gerados automaticamente produzidos em muitas ferramentas de desenvolvimento normalmente reduzem todos os parâmetros a modelos que são muito amplos para serem efetivos na identificação de ameaças possíveis. Esquemas e listas brancas criados manualmente são preferidos porque os desenvolvedores podem restringir entradas com base em seu entendimento do modelo de dados que um aplicativo espera.

Uma opção para tipos de conteúdo com base em XML é usar a linguagem de esquema XML, que é altamente efetiva para criar modelos de conteúdo restritos e estruturas altamente limitadas. Para os tipos de dados JSON cada vez mais comuns, há várias linguagens de descrição de esquema JSON. Embora não seja tão sofisticado como o XML, o JSON é muito mais simples de compor e de compreender, oferecendo uma transparência que realmente simplifica a proteção.

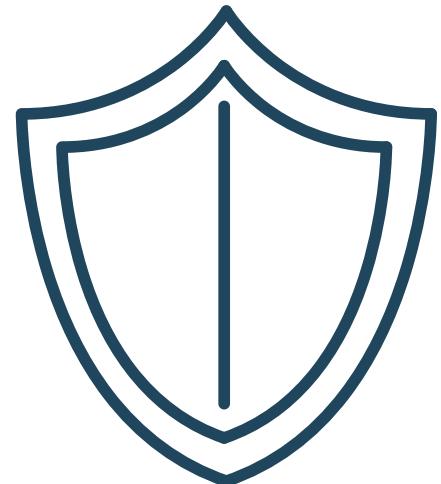
Estratégia 1:
validar os parâmetros

Estratégia 2:
aplicar detecção explícita
de ameaças

Estratégia 3:
ativar o SSL em todos os lugares

Estratégia 4:
aplicar autenticação
e autorização rigorosas

Estratégia 5:
usar soluções comprovadas



Estratégia 2: aplicar detecção explícita de ameaças

A boa validação do esquema pode proteger contra muitos ataques de injeção, mas considere também verificar explicitamente assinaturas de ataques comuns. Os ataques de injeção de SQL ou de injeção de script normalmente se traem por seguir padrões que são fáceis de localizar por meio da verificação de entrada bruta.

Considere também que os ataques podem tomar outras formas, como DoS (Denial of Service - Negação de Serviço). Utilize a infraestrutura de rede para localizar e mitigar ataques de DoS em nível de rede, mas verifique também ataques de DoS que exploram parâmetros. Mensagens muito grandes, estruturas de dados com aninhamento pesado ou estruturas de dados extremamente complexas podem resultar em um ataque de negação de serviço efetivo que consome desnecessariamente recursos de um servidor de API afetado.

Aplique detecção de vírus a todo o conteúdo codificado potencialmente arriscado. As APIs envolvidas na transferência de arquivos devem decodificar anexos base64 e enviá-los para verificação de vírus no nível do servidor antes de persisti-los em um sistema de arquivos onde podem ser ativados acidentalmente.

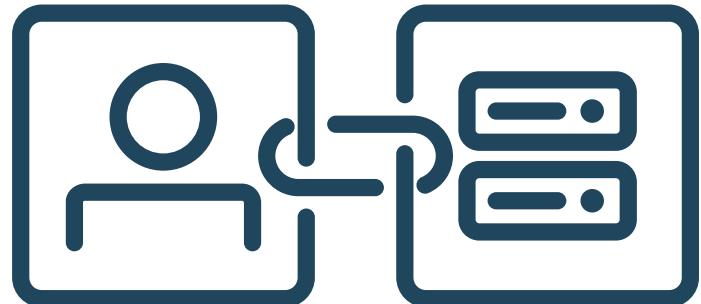
Estratégia 1:
validar os parâmetros

Estratégia 2:
aplicar detecção explícita
de ameaças

Estratégia 3:
ativar o SSL em todos os lugares

Estratégia 4:
aplicar autenticação
e autorização rigorosas

Estratégia 5:
usar soluções comprovadas



Estratégia 3: ativar o SSL em todos os lugares

Torne o SSL/TLS a regra para todas as APIs. No século 21, o SSL não é um luxo, mas um requisito básico. A adição de SSL/TLS, bem como sua aplicação correta, é uma defesa eficaz contra o risco de ataques de intermediários.

O SSL/TLS fornece integridade em todos os dados trocados entre um cliente e um servidor, incluindo tokens de acesso importantes, como os usados no OAuth. Opcionalmente, isso fornece autenticação no lado do cliente usando certificados, o que é importante em muitos ambientes.

Estratégia 1:

validar os parâmetros

Estratégia 2:

aplicar detecção explícita
de ameaças

Estratégia 3:

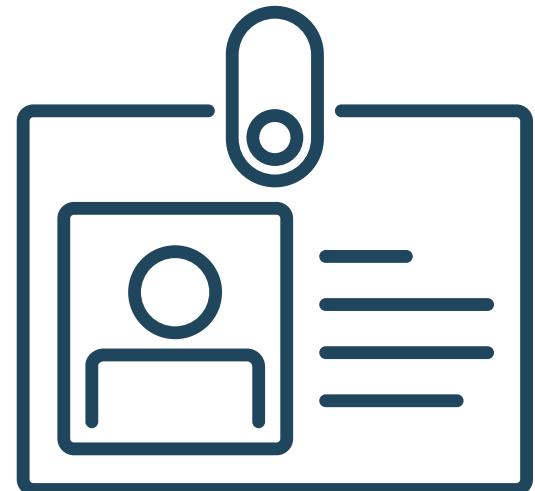
ativar o SSL em todos os lugares

Estratégia 4:

aplicar autenticação
e autorização rigorosas

Estratégia 5:

usar soluções comprovadas



Estratégia 4: aplicar autenticação e autorização rigorosas

A identidade do usuário e do aplicativo são conceitos que devem ser implementados e gerenciados separadamente. Considere a autorização com base em um contexto de identidade amplo, incluindo fatores práticos, como um endereço IP de entrada (se conhecido como fixo ou dentro de um intervalo específico), períodos de tempo de acesso, identificação de dispositivos (útil para aplicativos móveis), geolocalização, etc.

O OAuth está se tornando rapidamente um recurso utilizado para autorização de APIs centradas no usuário, mas também é uma tecnologia difícil, complexa e que passa por mudanças rapidamente. Os desenvolvedores devem acatar os casos de uso básicos e bem-comprendidos do OAuth e sempre usar bibliotecas existentes, em vez de tentar criar suas próprias.

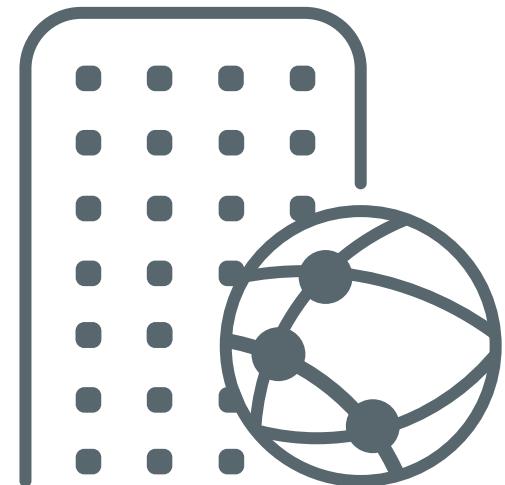
Estratégia 1:
validar os parâmetros

Estratégia 2:
aplicar detecção explícita
de ameaças

Estratégia 3:
ativar o SSL em todos os lugares

Estratégia 4:
aplicar autenticação
e autorização rigorosas

Estratégia 5:
usar soluções comprovadas



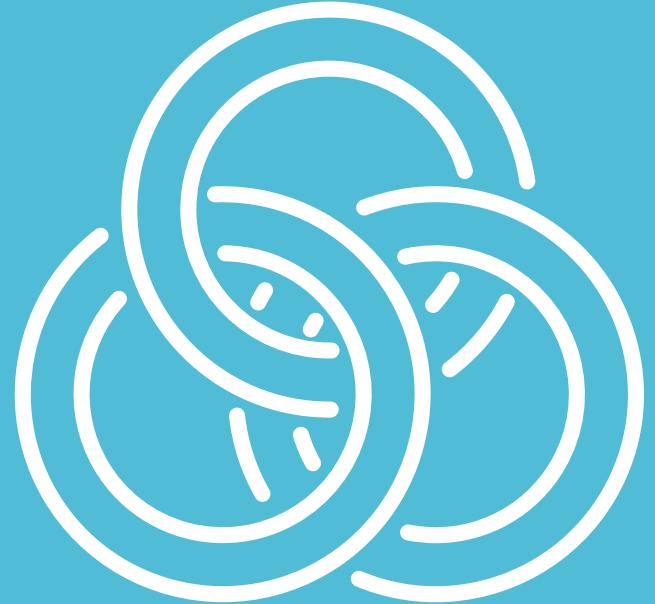
Estratégia 5: usar soluções comprovadas

A primeira regra de segurança é: não invente a sua própria. Não há razão para criar sua própria estrutura de segurança para APIs, já que existem excelentes soluções de segurança para APIs. O desafio está em aplicá-las corretamente.

Proteger arquiteturas de APIs

A melhor maneira de proteger sua API contra qualquer tipo de intrusão é separar a implementação da API e a segurança da API em camadas distintas. Essa é uma separação de funções muito lógica, concentrando a experiência no problema certo e no momento adequado.

Essa abordagem permite que um desenvolvedor de API se concentre completamente no domínio do aplicativo, garantindo que cada API seja bem-desenvolvida, e promove a integração entre diferentes aplicativos. Assim, a segurança fica por conta do domínio do especialista, que pode concentrar-se exclusivamente em identidades, ameaças e segurança de dados.



Conclusão

As APIs representam uma ótima oportunidade para a empresa integrar aplicativos de maneira fácil e rápida. Mas elas podem ser uma faca de dois gumes: prometendo agilidade e, ao mesmo tempo, aumentando o risco. Mas, se uma organização puder resolver a segurança das APIs como um desafio de arquitetura muito antes de qualquer desenvolvimento, ela poderá colher os frutos desse avanço revolucionário de maneira segura.



Saiba mais sobre os riscos e os benefícios inerentes às APIs com estes recursos:

www.ca.com/br/techinsights/security

Utilize a documentação técnica de líderes do setor e os relatórios de pesquisa e do setor para obter uma visão das principais tendências de segurança atuais.

www.ca.com/br/api

Saiba como conectar sua empresa a aplicativos móveis, plataformas na nuvem e redes de desenvolvedores com segurança por meio de APIs.