Introdução

Autenticação é o processo de verificação de que um indivíduo, entidade ou site é quem afirma ser. A autenticação no contexto de aplicativos da Web é comumente executada enviando um nome de usuário ou ID e um ou mais itens de informações particulares que apenas um determinado usuário deve saber.

O Gerenciamento de Sessão é um processo pelo qual um servidor mantém o estado de uma entidade interagindo com ele. Isso é necessário para um servidor lembrar como reagir a solicitações subsequentes durante uma transação. As sessões são mantidas no servidor por um identificador de sessão que pode ser passado e encaminhado entre o cliente e o servidor ao transmitir e receber solicitações. As sessões devem ser únicas por usuário e computacionalmente muito difíceis de prever.

Orientações Gerais de Autenticação

IDs de usuário

Certifique-se de que seus nomes de usuário / userids sejam insensíveis a maiúsculas e minúsculas. O usuário 'smith' e o usuário 'Smith' devem ser o mesmo usuário. Os nomes de usuários também devem ser exclusivos. Para nomes de usuários de aplicativos de alta segurança, eles podem ser atribuídos e secretos em vez de dados públicos definidos pelo usuário.

Endereço de email como um ID do usuário

Para obter informações sobre como validar endereços de e-mail, visite a discussão por e-mail de validação de entrada .

Solução de autenticação e contas sensíveis

- Você **não** permitir login com contas sensíveis (ou seja, contas que podem ser usados internamente dentro da solução como um back-end / meio-ware / DB) para qualquer interface do usuário front-end
- Você **não** usar a mesma solução de autenticação (por exemplo, IDP / AD) usado internamente para acesso sem garantia (por exemplo, acesso público / DMZ)

Implementar controles de força adequados da senha

Uma preocupação importante ao usar senhas para autenticação é a força da senha. Uma política de senha "forte" dificulta ou dificulta a adivinhação da senha por meios manuais ou automatizados. As seguintes características definem uma senha forte:

Comprimento da senha

Senhas mais longas fornecem uma combinação maior de caracteres e, consequentemente, dificultam a adivinhação de um invasor.

- **O** comprimento **mínimo** das senhas deve ser **imposto** pelo aplicativo.
  - Senhas **menores que 10 caracteres** são consideradas fracas (NIST SP800-132).

Embora a aplicação de tamanho mínimo possa causar problemas na memorização de senhas entre alguns usuários, os aplicativos devem incentivá-los a definir *senhas* (frases ou combinações de palavras) que podem ser muito mais longas do que as senhas comuns e muito mais fáceis de lembrar.

- **O** comprimento **máximo da** senha não deve ser definido **muito baixo** , pois isso impedirá que os usuários criem senhas. O comprimento máximo típico é de 128 caracteres.
    - Frases secretas com menos de 20 caracteres são geralmente consideradas fracas se consistirem apenas de caracteres latinos minúsculos.

Complexidade de senha

Os aplicativos devem impor regras de complexidade de senha para desencorajar senhas fáceis de adivinhar. Os mecanismos de senha devem permitir que praticamente qualquer caractere que o usuário possa digitar seja parte de sua senha, incluindo o caractere de espaço. As senhas devem, obviamente, diferenciar maiúsculas de minúsculas para aumentar sua complexidade. Ocasionalmente, encontramos sistemas em que as senhas não diferenciam maiúsculas de minúsculas, frequentemente devido a problemas de sistema herdados, como mainframes antigos que não possuíam senhas que diferenciam maiúsculas de minúsculas.

O mecanismo de alteração de senha deve exigir um nível mínimo de complexidade que faça sentido para o aplicativo e sua população de usuários. Por exemplo:

- A senha deve atender a pelo menos 3 das 4 regras de complexidade a seguir
    - pelo menos 1 caractere maiúsculo (AZ)
    - pelo menos 1 caractere minúsculo (az)
    - pelo menos 1 dígito (0-9)
    - pelo menos 1 caractere especial (pontuação) - não se esqueça de tratar o espaço como caracteres especiais também
- pelo menos 10 caracteres
- no máximo 128 caracteres
- não mais de 2 caracteres idênticos em uma linha (por exemplo, 111 não permitido)

Leitura adicional:

- Seus requisitos de complexidade de senha são inúteis - Apresentação do OWASP AppSecUSA 2014 para uma discussão mais aprofundada das regras de complexidade de senhas herdadas
- PathWell: Histologia de topologia de senha

Topologias de Senha

- Banir topologias de senha comumente usadas

- Forçar vários usuários a usar diferentes topologias de senha
- Exigir uma alteração de topologia mínima entre senhas antigas e novas

informação adicional

- Certifique-se de que todos os caracteres digitados pelo usuário estejam realmente incluídos na senha. Vimos sistemas que truncam a senha com um comprimento menor do que o fornecido pelo usuário (por exemplo, truncado em 15 caracteres quando eles inseriram 20).
- Como os aplicativos exigem políticas de senhas mais complexas, eles precisam ser muito claros sobre o que são essas políticas. A política necessária precisa ser explicitamente declarada na página de alteração de senha
- Se a nova senha não estiver em conformidade com a política de complexidade, a mensagem de erro deverá descrever TODAS as regras de complexidade que a nova senha não atende, não apenas a primeira regra que não está em conformidade.

Implementar mecanismo de recuperação de senha segura

É comum que um aplicativo tenha um mecanismo que forneça um meio para um usuário obter acesso a sua conta caso ele esqueça sua senha. Por favor, consulte Esqueceu a senha da folha de fraude para obter detalhes sobre esse recurso.

Armazene senhas de forma segura

É fundamental que um aplicativo armazene uma senha usando a técnica criptográfica correta. Por favor, consulte a folha de dicas do Armazenamento de senhas para obter detalhes sobre esse recurso.

Transmitir senhas somente por TLS ou outro transporte forte

Veja: Folha de fraude de proteção de camada de transporte

A página de login e todas as páginas autenticadas subsequentes devem ser acessadas exclusivamente por TLS ou outro transporte forte. A página de login inicial, chamada de "página de entrada de login", deve ser veiculada por TLS ou outro transporte forte. A falha em utilizar o TLS ou outro transporte forte para a página de entrada de login permite que um invasor modifique a ação do formulário de login, fazendo com que as credenciais do usuário sejam postadas em um local arbitrário. A falha em utilizar o TLS ou outro transporte forte para páginas autenticadas após o login permite que um invasor visualize a ID de sessão não criptografada e comprometa a sessão autenticada do usuário.

Exigir nova autenticação para recursos confidenciais

Para mitigar o CSRF e o seqüestro de sessão, é importante exigir as credenciais atuais de uma conta antes de atualizar informações confidenciais da conta, como a senha do usuário, o email do usuário ou antes de transações confidenciais, como enviar uma compra para um novo endereço. Sem essa contramedida, um invasor poderá executar transações confidenciais por meio de um ataque CSRF ou XSS sem precisar conhecer as credenciais atuais do usuário. Além disso, um invasor pode obter acesso físico temporário ao navegador do usuário ou roubar sua ID de sessão para assumir a sessão do usuário.

Considere a autenticação de transação forte

Alguns aplicativos devem usar um segundo fator para verificar se um usuário pode executar operações confidenciais. Para obter mais informações, consulte a folha de dicas de autorização de transação .

Autenticação de cliente TLS

A Autenticação de Cliente TLS, também conhecida como autenticação TLS bidirecional, consiste em ambos, navegador e servidor, enviando seus respectivos certificados TLS durante o processo de handshake TLS. Assim como você pode validar a autenticidade de um servidor usando o certificado e perguntando a uma Autoridade de Certificação (CA) bem conhecida se o certificado é válido, o servidor pode autenticar o usuário recebendo um certificado do cliente e validando contra uma CA de terceiros ou sua própria CA. Para fazer isso, o servidor deve fornecer ao usuário um certificado gerado especificamente para ele, atribuindo valores ao assunto para que eles possam ser usados para determinar qual usuário o certificado deve validar. O usuário instala o certificado em um navegador e agora o usa para o site.

É uma boa ideia fazer isso quando:

- É aceitável (ou mesmo preferido) que o usuário tenha acesso somente ao site a partir de um único computador / navegador.
- O usuário não se assusta facilmente com o processo de instalação de certificados TLS em seu navegador ou alguém, provavelmente do suporte de TI, fará isso para o usuário.
- O site requer uma etapa extra de segurança.
- Também é bom usar quando o site é para uma intranet de uma empresa ou organização.

Geralmente, não é uma boa ideia usar esse método para sites amplamente e publicamente disponíveis que terão um usuário médio. Por exemplo, não seria uma boa ideia implementar isso em um site como o Facebook. Embora esta técnica possa impedir que o usuário tenha que digitar uma senha (portanto, protegendo contra um keylogger médio de roubá-la), ainda é considerado uma boa ideia considerar o uso de uma senha e autenticação de cliente TLS combinadas.

Para obter mais informações, consulte: Handshake TLS autenticado pelo cliente.

Autenticação e Mensagens de Erro

Mensagens de erro implementadas incorretamente no caso da funcionalidade de autenticação podem ser usadas para fins de identificação de usuário e senha. Um aplicativo deve responder (HTTP e HTML) de maneira genérica.

Respostas de autenticação

Um aplicativo deve responder com uma mensagem de erro genérica, independentemente de o ID do usuário ou a senha estar incorreta. Ele também não deve indicar o status de uma conta existente.

Exemplos de resposta incorretos

- "Login para o usuário foo: senha inválida"
- "Login com falha, ID de usuário inválido"
- "Login falhou; conta desativada"
- "Login falhou; este usuário não está ativo"

Exemplo de Resposta Correta

- "Login falhou; userID ou senha inválida"

A resposta correta não indica se o ID do usuário ou a senha é o parâmetro incorreto e, portanto, inferir um ID do usuário válido.

Códigos de erro e URLs

O aplicativo pode retornar um código de erro HTTP diferente, dependendo da resposta da tentativa de autenticação. Pode responder com um 200 para um resultado positivo e um 403 para um resultado negativo. Mesmo que uma página de erro genérica seja mostrada para um usuário, o código de resposta HTTP pode diferir, o que pode vazar informações sobre se a conta é válida ou não.

Evitar ataques de força bruta

Se um invasor conseguir adivinhar senhas sem que a conta seja desativada devido a tentativas de autenticação com falha, o invasor terá a oportunidade de continuar com um ataque de força bruta até que a conta seja comprometida. Automatizar ataques de força bruta / adivinhação de senha em aplicativos da Web é um desafio trivial. Mecanismos de bloqueio de senha devem ser empregados para bloquear uma conta se mais de um número predefinido de tentativas de login malsucedidas forem feitas. Mecanismos de bloqueio de senha têm uma fraqueza lógica. Um invasor que realiza um grande número de tentativas de autenticação em nomes de contas conhecidos pode produzir um resultado que bloqueia blocos inteiros de contas de usuário. Dado que a intenção de um sistema de bloqueio de senha é proteger contra ataques de força bruta, uma estratégia sensata é bloquear as contas por um período de tempo (por exemplo, 20 minutos).

Além disso, a autenticação multifator é um impedimento muito poderoso ao tentar evitar ataques de força bruta, já que as credenciais são um alvo em movimento. Quando o multifator é implementado e ativo, o bloqueio de conta pode não ser mais necessário.

Registro e Monitoramento

Ativar o registro e o monitoramento de funções de autenticação para detectar ataques / falhas em tempo real

- Garantir que todas as falhas sejam registradas e revisadas
- Garantir que todas as falhas de senha sejam registradas e revisadas
- Garantir que todos os bloqueios de conta sejam registrados e revisados

Uso de protocolos de autenticação que não exigem senha

Embora a autenticação por meio de uma combinação de usuário / senha e o uso de autenticação multifator seja considerada geralmente segura, há casos de uso em que não é considerada a melhor opção ou até mesmo segura. Exemplos disso são aplicativos de terceiros que desejam se conectar ao aplicativo da Web, seja de um dispositivo móvel, de outro site, de uma área de trabalho ou de outras situações. Quando isso acontece, NÃO é considerado seguro permitir que o aplicativo de terceiros armazene a combinação de usuário / senha, desde então ele estende a superfície de ataque para suas mãos, onde ela não está sob seu controle. Para isso e outros casos de uso, existem vários protocolos de autenticação que podem protegê-lo da exposição dos dados de seus usuários aos invasores.

OAuth

A Autorização Aberta (OAuth) é um protocolo que permite que um aplicativo se autentique em um servidor como usuário, sem exigir senhas ou qualquer servidor de terceiros que atue como um provedor de identidade. Ele usa um token gerado pelo servidor e fornece como a maioria dos fluxos de autorização ocorre, para que um cliente, como um aplicativo móvel, possa informar ao servidor qual usuário está usando o serviço.

A recomendação é usar e implementar o OAuth 1.0a ou o OAuth 2.0, já que a primeira versão (OAuth1.0) foi considerada vulnerável à correção de sessão.

O OAuth 2.0 depende do HTTPS para segurança e é atualmente usado e implementado por APIs de empresas como Facebook, Google, Twitter e Microsoft. OAuth1.0a é mais difícil de usar porque requer o uso de bibliotecas criptográficas para assinaturas digitais. No entanto, como o OAuth1.0a não depende de HTTPS para segurança, ele pode ser mais adequado para transações de maior risco.

OpenId

O OpenId é um protocolo baseado em HTTP que usa provedores de identidade para validar que um usuário é quem ele diz ser. É um protocolo muito simples que permite que um provedor de serviços inicie o caminho para logon único (SSO). Isso permite que o usuário reutilize uma única identidade fornecida a um provedor de identidade confiável OpenId e seja o mesmo usuário em vários sites, sem a necessidade de fornecer a senha a nenhum website, exceto pelo provedor de identidade OpenId.

Devido à sua simplicidade e que fornece proteção de senhas, o OpenId foi bem adotado. Alguns dos provedores de identidade conhecidos do OpenId são o Stack Exchange, o Google, o Facebook e o Yahoo!

Para ambientes não empresariais, o OpenId é considerado uma opção segura e geralmente melhor, desde que o provedor de identidade seja de confiança.

SAML

O SAML (Security Assertion Markup Language) costuma ser considerado para competir com o OpenId. A versão mais recomendada é a 2.0, já que é muito completa e fornece uma forte segurança. Como o OpenId, o SAML usa provedores de identidade, mas, diferentemente do OpenId, é baseado em XML e oferece mais flexibilidade. O SAML é baseado em redirecionamentos de navegador que enviam dados XML. Além disso, o SAML não é iniciado apenas por um provedor de serviços; Ele também pode ser iniciado a partir do provedor de identidade. Isso permite que o usuário navegue por diferentes portais enquanto ainda está sendo autenticado sem precisar fazer nada, tornando o processo transparente.

Embora o OpenId tenha conquistado a maior parte do mercado de consumo, o SAML é geralmente a escolha para aplicativos corporativos. A razão para isso é que há poucos provedores de identidade OpenID que são considerados de classe corporativa (o que significa que a maneira como eles validam a identidade do usuário não possui altos padrões exigidos para a identidade corporativa). É mais comum ver o SAML sendo usado dentro de sites de intranet, às vezes até usando um servidor da intranet como o provedor de identidade.

Nos últimos anos, aplicativos como SAP ERP e SharePoint (SharePoint usando o Active Directory Federation Services 2.0) decidiram usar a autenticação SAML 2.0 como um método geralmente preferido para implementações de logon único sempre que a federação empresarial é necessária para serviços da Web e da Web. aplicações.

Veja também: Folha de truques de segurança do SAML

FIDO

A aliança Fast Identity Online (FIDO) criou dois protocolos para facilitar a autenticação on-line: o protocolo UAF (Universal Authentication Framework) e o protocolo U2F (Universal Second Factor). Enquanto o UAF se concentra na autenticação sem senha, o U2F permite a adição de um segundo fator à autenticação baseada em senha existente. Ambos os protocolos são baseados em um modelo de desafio-resposta de criptografia de chave pública.

A UAF aproveita as tecnologias de segurança existentes presentes nos dispositivos para autenticação, incluindo sensores de impressões digitais, câmeras (biometria de face), microfones (biometria de voz), Trusted Execution Environments (TEEs), Secure Elements (SEs) e outros. O protocolo é projetado para conectar esses recursos de dispositivos a uma estrutura de autenticação comum. O UAF trabalha com aplicativos nativos e aplicativos da web.

O U2F aumenta a autenticação baseada em senha usando um token de hardware (geralmente USB) que armazena chaves de autenticação criptográfica e as utiliza para assinatura. O usuário pode usar o mesmo token como um segundo fator para vários aplicativos. O U2F trabalha com aplicativos da web. Ele fornece **proteção contra phishing** usando a URL do site para procurar a chave de autenticação armazenada.

Diretrizes Gerais do Gerenciamento de Sessões

O gerenciamento de sessões está diretamente relacionado à autenticação. As **Diretrizes Gerais de Gerenciamento de Sessão** anteriormente disponíveis nesta Folha de Fraude de Autenticação OWASP foram integradas na Folha de **Dicas** do Gerenciamento de Sessões .

Gerenciadores de Senhas

Gerenciadores de senhas são programas, plugins de navegadores ou serviços da Web que automatizam o gerenciamento de um grande número de credenciais diferentes, incluindo memorização e preenchimento, geração de senhas aleatórias em sites diferentes etc. Embora o uso de gerenciadores de senhas esteja sujeito a controvérsias e muitas organizações bloqueiem seu uso , sua contribuição para a segurança da autenticação é positiva, conforme apontado pelo National Cyber Security Center .

Os aplicativos da Web devem, pelo menos, não tornar o trabalho dos gerentes de senha mais difícil do que o necessário, observando as seguintes recomendações:

- usar formulários HTML padrão para entrada de nome de usuário e senha com `type` atributos apropriados ,
- não limite artificialmente senhas de usuários a um tamanho "razoável para humanos" e permita senhas de até 128 caracteres,
- não impede artificialmente copiar e colar nos campos de nome de usuário e senha,
- evite páginas de login baseadas em plugins (Flash, Silverlight etc)

A partir de 2017, o padrão Nível de Gerenciamento de Credenciais 1 para navegadores da Web está sendo desenvolvido, o que pode facilitar ainda mais a interação entre gerenciadores de senhas e esquemas de login complexos (por exemplo, logon único).

Autores e Editores Principais

Eoin Keary e oinkeary

Jim Manico

Timo Goosen

Pawel Krawczyk

Sven Neuhaus

Manuel Aude Morales