Attaque des contrôles d'accès

Présentation de STI

Cassandre Wojciechowski Gwendoline Dössegger Gabriel Roch

Haute École d'Ingénierie et de Gestion du Canton de Vaud

20 novembre 2020

Sommaire

Sommaire

Exemple ou note

Attaque

Sécurisation

Qu'est-ce que les contrôles d'accès (access controls) ?

- Vérification du niveau d'accréditation (vertical)
- Vérification de l'identité de l'utilisateur (horizontal)
- Restriction de l'accès en fonction des vérifications ci-dessus

Section 1

Vulnérabilités communes

Sommaire

Vulnérabilités communes

- Vulnérabilités verticales
- Vulnérabilités horizontales
- Vulnérabilités dépendant du contexte

Vulnérabilités verticales

Vulnérabilité

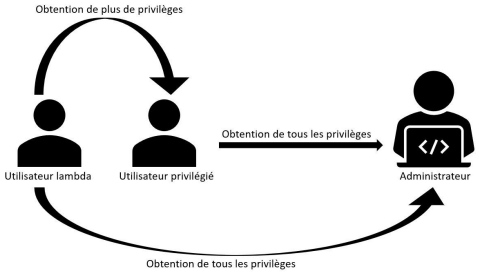
Utiliser des fonctions de l'application quand notre rôle ne le permet normalement pas.

Type d'attaque

Escalade de privilèges verticale — *vertical privilege esalation*

Exemple d'attaque

Devenir administrateur quand on est un utilisateur lambda.



Vulnérabilités horizontales

Vulnérabilité

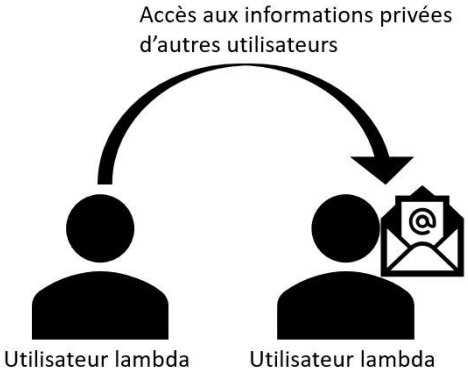
Accéder aux ressources d'autres utilisateurs du même niveau.

Type d'attaque

Escalade de privilèges horizontale — *horizontal privilege escalation*

Exemple d'attaque

Un utilisateur lambda peut lire les emails d'un autre utilisateur lambda



Vulnérabilités dépendant du contexte

Vulnérabilité

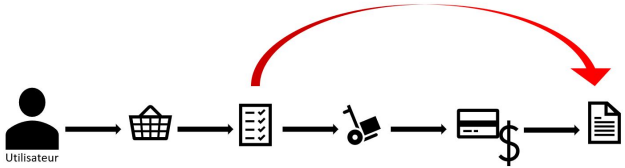
Le contrôle d'accès ne prend pas en compte l'état actuel de l'application

Type d'attaque

Accès hors du flux d'exécution normal — *business logic exploitation*

Exemple d'attaque

Accéder à une page de paiement en ligne sans passer par l'étape de calcul des frais de port.
Sauter des étapes de vérification normalement obligatoires.



Contrôles d'accès par connaissance d'URL

- Certains sites masquent l'URL de la page d'admin pour les utilisateurs
- Le contrôle d'accès n'est pas toujours fait sur les pages d'admin

Contrôles d'accès par connaissance d'URL

- Certains sites masquent l'URL de la page d'admin pour les utilisateurs
- Le contrôle d'accès n'est pas toujours fait sur les pages d'admin

L'URL n'est pas secret

Le fait que l'URL ne soit pas affichée n'empêche pas l'attaquant d'y accéder, il va pouvoir la trouver autre part (essayer des URLs habituelles, outils de bruteforce, dans le code source, sur internet, ...)
D'autres moyens peuvent également être utilisés car une URL n'est jamais traitée comme confidentielle par les logiciels et serveurs.

Contrôles d'accès par connaissance d'URL

- Certains sites masquent l'URL de la page d'admin pour les utilisateurs
- Le contrôle d'accès n'est pas toujours fait sur les pages d'admin

Sécurisation

- Ne pas se baser sur l'ignorance des utilisateurs pour les URLs et les identifiants des documents

L'URL n'est pas secret

Le fait que l'URL ne soit pas affichée n'empêche pas l'attaquant d'y accéder, il va pouvoir la trouver autre part (essayer des URLs habituelles, outils de bruteforce, dans le code source, sur internet, ...) D'autres moyens peuvent également être utilisés car une URL n'est jamais traitée comme confidentielle par les logiciels et serveurs.

Découvrir une URL d'administration

Attaque

- Commentaires dans le code source,
- Affichage à l'écran,
- Historique des navigateurs, favoris,
- Envoi du lien par e-mail (ou autres outils) par certains utilisateurs,
- Logs (clients, serveurs, proxys),
- Scripts de génération des menus.

Demo

Accès direct à l'API

Sur certains sites, les API ne font pas les mêmes contrôles d'accès que pour les pages "standards"

Attaque

- Analyse des requêtes faites par le navigateur
- Test des URLs standards
- Test les ressources devinables

Sécurisation

- L'API doit être sécurisée de la même manière que les pages standards de modification, car les mêmes risques s'appliquent
- Les fonctions privilégiées doivent vérifier les privilèges de l'utilisateur.

Contrôle basé sur les identifiants

Certains sites permettent l'accès à des fonctionnalités en fonction de l'ID d'une ressource (statique ou dynamique).

- Les IDs peuvent être devinés.
- Les GUIDs n'améliorent que peu la sécurité (prévisibles).
  - `3F2504E0-4F89-11D3-9A0C-0305E82C3301`
- Les IDs peuvent être affichés ailleurs.

Contrôle basé sur les identifiants

Attaque

- Deviner les identifiants
- Les identifiants sont les mêmes quelle que soit l'action
- Accès à une ressource connue avec un compte moins privilégié (et même sans s'identifier)

Sécurisation

- Vérifier les droits d'accès à la ressource de l'utilisateur.
- Vérifier que l'utilisateur a le droit de faire l'action demandée

Contrôle basé sur la requête

- Les contrôles d'accès peuvent ne pas être effectués par l'application mais par l'infrastructure (par ex. Apache)
- Apache peut valider une requête avec les critères suivants
  - L'URL demandée (`/admin`, `/image`)
  - Le type de fichier demandé
  - La méthode HTTP utilisée (`GET`, `POST`, `HEAD`, ...)
  - Les identifiants de l'utilisateur
  - Les cookies (admin=true)

Methode HTTP particulière

Une requête `HEAD` doit renvoyer les mêmes en-têtes que `GET`, mais sans le corps du message. Pour cela, les mêmes scripts que pour GET sont généralement executés.

Methode HTTP non-standard

Les requêtes HTTP non-standards (`DELETE`, `TOTO`, ...) peuvent également être traitées par les mêmes scripts.

Contrôle basé sur la requête

Attaque

- Est-ce qu'une requête HEAD fonctionne ?
- Est-ce qu'une méthode HTTP non-valide fonctionne ?
- Est-ce que les privilèges sont contrôle de manière identique pour l'API que pour l'application ?
- Ce type de contrôle peut également être mis en place pour les pages dynamiques du site.

Sécurisation

- Faire attention lors de la configuration
- Pour les fichiers statiques : passer par un script qui renvoie le fichier après authentification (le fichier original étant innacessible en HTTP)

Demo

Fonctions en plusieurs étapes

Exemple de réception de matériel

1. enregistrement de la facture,
2. sélection des comptes débiteurs,
3. mise à jour du stock,
4. validation du paiement.

Fonctions en plusieurs étapes

- Contrôles d'accès à chaque étape.
- Vérification de toutes les étapes précedentes à chaque fois.

Attaque

- Validation finale directe
- Bypass d'une étape
- Modification du header `REFERER`

Sécurisation

- Ne pas faire confiance aux utilisateurs pour utiliser les fonctionnalités comme elles ont été prévues
- Tout revérifier à chaque étape
- Le header `REFERER` doit être considéré comme non-sécurisé

Demo

Contrôles d'accès basés sur des paramètres

L'information concernant le rôle ou le niveau d'accès de l'utilisateur est transmis par :

- des cookies
- un champ masqué dans le formulaire
- un paramètre de la requête
- une information dans l'URL

Ces techniques ne sont pas sécurisées car un attaquant peut modifier ces champs et usurper l'identité de l'administrateur.

Sécurisation

- Ne pas faire confiance aux paramètres envoyés par les utilisateurs

Contrôles d'accès basés sur la géolocalisation

Un utilisateur peut modifier sa géolocalisation perçue par le serveur

- VPN
- Proxy web
- En-tête HTTP X-Forwarded-For

Sécurisation

- Ne pas partir du principe que la localisation est juste

Comparer différents comptes

Attaque

- Cartographier le site avec un logiciel comme Burp, pour détecter des URLs
- Tester les URLs obtenues à partir d'un compte privilégié avec un compte moins privilégié
- Il faut potentiellement compléter les outils automatiques avec nos connaissances

Section 2

Les bonnes pratiques

Sommaire

Les bonnes pratiques

- Ne pas laisser des fonctionnalités non utilisées

Les bonnes pratiques

- Ne pas laisser des fonctionnalités non utilisées

- Ne pas faire confiance aux utilisateurs pour utiliser les fonctionnalités comme elles ont été prévues
- Ne pas faire confiance aux utilisateurs pour ne pas détourner les données transmises par le côté client
- Il ne faut faire confiance qu'aux données provenant du côté serveur, et non du côté client. Il faut revalider les identifiants à chaque transmission de données

Les bonnes pratiques

- Ne pas laisser des fonctionnalités non utilisées

- Ne pas faire confiance aux utilisateurs pour utiliser les fonctionnalités comme elles ont été prévues
- Ne pas faire confiance aux utilisateurs pour ne pas détourner les données transmises par le côté client
- Il ne faut faire confiance qu'aux données provenant du côté serveur, et non du côté client. Il faut revalider les identifiants à chaque transmission de données

- Evaluer et documenter les contrôles d'accès pour chaque partie de l'application (pour les fonctionnalités et les ressources)

Les bonnes pratiques

- Ne pas laisser des fonctionnalités non utilisées

- Ne pas faire confiance aux utilisateurs pour utiliser les fonctionnalités comme elles ont été prévues
- Ne pas faire confiance aux utilisateurs pour ne pas détourner les données transmises par le côté client
- Il ne faut faire confiance qu'aux données provenant du côté serveur, et non du côté client. Il faut revalider les identifiants à chaque transmission de données

- Evaluer et documenter les contrôles d'accès pour chaque partie de l'application (pour les fonctionnalités et les ressources)

- Toutes les décisions d'autorisation doivent être prises à partir de la session de l'utilisateur

Les bonnes pratiques

- Utiliser un composant central à l'application pour vérifier tous les contrôles d'accès
- Utiliser ce composant central pour valider toutes les requêtes client
  - Plus grande clarté des contrôles d'accès
  - Meilleure maintenabilité (plus efficace et sûr)
  - Plus adaptable
  - Moins d'erreurs et d'omissions

Les bonnes pratiques

- Utiliser un composant central à l'application pour vérifier tous les contrôles d'accès
- Utiliser ce composant central pour valider toutes les requêtes client
  - Plus grande clarté des contrôles d'accès
  - Meilleure maintenabilité (plus efficace et sûr)
  - Plus adaptable
  - Moins d'erreurs et d'omissions

- Utiliser des techniques de programmation pour forcer le contrôle d'accès à être effectué et éviter que le développeur passe outre

Les bonnes pratiques

- Utiliser un composant central à l'application pour vérifier tous les contrôles d'accès
- Utiliser ce composant central pour valider toutes les requêtes client
  - Plus grande clarté des contrôles d'accès
  - Meilleure maintenabilité (plus efficace et sûr)
  - Plus adaptable
  - Moins d'erreurs et d'omissions

- Utiliser des techniques de programmation pour forcer le contrôle d'accès à être effectué et éviter que le développeur passe outre

- Pour les parties sensibles de l'application, effectuer des contrôles supplémentaires, par exemple basés sur l'adresse IP

Les bonnes pratiques

- Accès à des fichiers statiques :
  - Accès indirect en passant un nom de fichier à une page dynamique côté serveur qui va implémenter un contrôle d'accès et retourner le fichier (pas rediriger dessus, car cela ne mettrait en place aucun contrôle)
  - Utiliser l'authentification HTTP et d'autres fonctionnalités du serveur d'application pour contrôler l'authentification (cela risque de faire une vérification différente de celle du composant central, **il faut donc s'assurer que cela soit consistant**)

Les bonnes pratiques

- Accès à des fichiers statiques :
  - Accès indirect en passant un nom de fichier à une page dynamique côté serveur qui va implémenter un contrôle d'accès et retourner le fichier (pas rediriger dessus, car cela ne mettrait en place aucun contrôle)
  - Utiliser l'authentification HTTP et d'autres fonctionnalités du serveur d'application pour contrôler l'authentification (cela risque de faire une vérification différente de celle du composant central, **il faut donc s'assurer que cela soit consistant**)

- Pour des actions critiques, il faut ré-authentifier l'utilisateur à chaque transaction et utiliser un système d'authentification multi-facteurs

Les bonnes pratiques

- Accès à des fichiers statiques :
  - Accès indirect en passant un nom de fichier à une page dynamique côté serveur qui va implémenter un contrôle d'accès et retourner le fichier (pas rediriger dessus, car cela ne mettrait en place aucun contrôle)
  - Utiliser l'authentification HTTP et d'autres fonctionnalités du serveur d'application pour contrôler l'authentification (cela risque de faire une vérification différente de celle du composant central, **il faut donc s'assurer que cela soit consistant**)

- Pour des actions critiques, il faut ré-authentifier l'utilisateur à chaque transaction et utiliser un système d'authentification multi-facteurs

- Logger toutes les actions effectuées quand des données sensibles sont concernées

Les bonnes pratiques

Application multi-tiers

- Mise en place de contrôles à chaque couche (si une couche est compromise, les autres ne le sont pas forcement)
- Le serveur de l'application peut contrôler les URLs selon le rôle de l'utilisateur
- L'application peut utiliser un compte de base de données séparé avec des privilèges limités pour chaque type d'utilisateur (privilèges en lecture-seule) et spécifier précisément quelles tables sont accessibles
- Utiliser un compte système avec des privilèges limités pour chaque composant

Modèle de contrôle d'accès

	Application Server	Application Roles	Database Privileges												
User type	**URL path**	**User role**	Search	Create Application	Edit Application	Purge Application	View Applications	Policy Updates	Rate Adjustment	View User Accounts	Create Users	View Company Ac	Edit Company Ac	Create Company	View Audit Log	Delegate privilege
Administrator	/*	Site Administrator	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
		Support	✓		✓		✓	✓		✓	✓	✓	✓	✓		
Site Supervisor	/admin/* /myQuotes/* /help/*	Back Office – New business		✓			✓									
		Back Office – Referrals		✓	✓	✓		✓	✓							
		Back Office – Helpdesk	✓				✓			✓		✓			✓	✓
Company Administrator	/myQuotes/* /help/*	Customer – Administrator		✓	✓	✓	✓				✓	✓	✓			✓
		Customer – New Business		✓		✓	✓									
		Customer – Support	✓				✓			✓						
Normal User	/myQuotes/dash.jsp /myQuotes/apply.jsp /myQuotes/search.jsp /help/*	User – Applications	✓	✓			✓									
		User – Referrals														
		User – Helpdesk														
		Unregistered (Read Only)	✓				✓									
Audit	(none)	Syslog Server Account													✓	

Modèle de contrôle d'accès

- Via des techniques de programmation
  - une matrice de droits est stockée dans la base de données
  - le programme se charge d'appliquer les contrôles (avec un algorithme aussi complexe que nécessaire)

Modèle de contrôle d'accès

- Via des techniques de programmation
  - une matrice de droits est stockée dans la base de données
  - le programme se charge d'appliquer les contrôles (avec un algorithme aussi complexe que nécessaire)
- A la discrétion de l'administrateur (discretionary access control DAC)
  - l'administrateur peut donner explicitement des privilèges à d'autres utilisateurs pour des ressources auxquelles ils ont accès
  - modèle fermé : white list
  - modèle ouvert : black list

Modèle de contrôle d'accès

- Via des techniques de programmation
  - une matrice de droits est stockée dans la base de données
  - le programme se charge d'appliquer les contrôles (avec un algorithme aussi complexe que nécessaire)
- A la discrétion de l'administrateur (discretionary access control DAC)
  - l'administrateur peut donner explicitement des privilèges à d'autres utilisateurs pour des ressources auxquelles ils ont accès
  - modèle fermé : white list
  - modèle ouvert : black list
- Basés sur des rôles (role-based access control RBAC)
  - chaque rôle donne accès à certains privilèges, pas trop de rôles, ni trop peu, il faut que cela reste gérable et sécurisé

Modèle de contrôle d'accès

- Via des techniques de programmation
  - une matrice de droits est stockée dans la base de données
  - le programme se charge d'appliquer les contrôles (avec un algorithme aussi complexe que nécessaire)
- A la discrétion de l'administrateur (discretionary access control DAC)
  - l'administrateur peut donner explicitement des privilèges à d'autres utilisateurs pour des ressources auxquelles ils ont accès
  - modèle fermé : white list
  - modèle ouvert : black list
- Basés sur des rôles (role-based access control RBAC)
  - chaque rôle donne accès à certains privilèges, pas trop de rôles, ni trop peu, il faut que cela reste gérable et sécurisé
- Via un composant externe
  - utilisation d'un compte de base de données différent pour les groupes d'utilisateurs afin de limiter leurs droits

Merci de votre attention

Ne faites pas confiance aux données en provenance du navigateur !