MODÈLES DE SÉCURITÉ ET D'AUTHENTIFICATION · MODULE 2

Authentification intégrée avec IIS et le navigateur

Guide de la leçon · Guide du lab / de l'examen

LearnWisej

Guide de la leçon

Choisir la frontière de site IIS, les fournisseurs de l'authentification Windows et l'ordre des challenges, la négociation du navigateur et la politique intranet, les SPN, les preuves Kerberos, la protection étendue et les proxys, et diagnostiquer une défaillance couche par couche. Ce que cela signifie pour un développeur qui sécurise les chemins de connexion, d'identité et d'autorisation d'une application Wisej.NET en production — et comment aborder ce module.

Objectif en production

Choisir la frontière de site IIS, les fournisseurs de l'authentification Windows et l'ordre des challenges, la négociation du navigateur et la politique intranet, les SPN, les preuves Kerberos, la protection étendue et les proxys, et diagnostiquer une défaillance couche par couche.

1. Le problème en production

Le personnel ouvre la route intranet d'AccessOps et s'attend à être connecté silencieusement avec son compte Windows, tandis que les partenaires atteignent une page de connexion publique qui ne doit jamais recevoir de challenge pour des identifiants Windows. Quand la connexion silencieuse échoue, le symptôme est une invite, une boucle de 401 ou une page d'erreur, et la cause se trouve dans IIS, le navigateur, l'annuaire ou un proxy, avant qu'aucun code Wisej.NET ne s'exécute. Ce module vous apprend à configurer délibérément cette frontière de l'hôte et à distinguer une défaillance de l'hôte d'un refus applicatif.

2. Concepts clés

Concept	Pourquoi c'est important
frontière de site	Une application IIS ou une liaison distincte par point d'entrée, de sorte que la route intranet réservée à Windows et la route publique en Anonymous portent chacune exactement une politique d'authentification visible.
challenge Negotiate	Un HTTP 401 portant `WWW-Authenticate: Negotiate` et/ou `NTLM` ; un navigateur éligible réessaie avec un en-tête `Authorization`, et Negotiate choisit Kerberos ou se replie sur NTLM.
preuve Kerberos	Un ticket de service pour le nom d'hôte canonique `HTTP/` sur le client, corrélé avec l'enregistrement de connexion du serveur ; l'en-tête de réponse seul ne prouve jamais quel fournisseur s'est exécuté.
authentification en mode noyau	Avec `useKernelMode`, c'est le pilote HTTP.sys qui effectue l'échange Windows ; le compte qui déchiffre les tickets est donc normalement le compte machine, sauf si `useAppPoolCredentials` en décide autrement.
SPN	Le nom principal de service (`HTTP/accessops.contoso.test`) qui indique au domaine pour quel compte un ticket est chiffré ; un SPN manquant ou en double pousse silencieusement les clients vers NTLM ou dans une boucle.
protection étendue	Des vérifications de liaison de canal et de liaison de service qui rattachent l'échange Windows à la connexion TLS et au SPN attendu, ce qui compte dès que TLS se termine sur un proxy.
principal de l'hôte dans Wisej.NET	Sur ASP.NET Core, l'utilisateur Windows n'atteint l'application que sous la forme d'`Application.User`, le `HttpContext.User` de la requête capturé par Wisej.NET ; il prouve qui s'est connecté, pas ce qu'il peut lire.

3. Le fil de la leçon

Une application par point d'entrée

AccessOps a deux portes d'entrée. La route intranet est destinée au personnel du domaine : dans IIS, vous installez le service de rôle Authentification Windows, vous l'activez sur cette application et vous y désactivez Anonymous, si bien que chaque requête reçoit un challenge avant que l'ASP.NET Core Module ne la transmette à l'application. La route publique sert la page de connexion des partenaires et, au Module 5, le callback OpenID Connect. Le navigateur du partenaire doit atteindre ces pages avant que l'application ne sache qui est le partenaire ; cette application garde donc le transport Anonymous activé et l'application Wisej.NET contrôle elle-même l'accès. Partager une seule application reviendrait soit à envoyer un challenge au callback externe, soit à ouvrir les écrans intranet.

Gardez les routes dans des applications IIS distinctes ou dans des liaisons distinctes avec leurs propres pools d'applications. Dans le lab, la route intranet est une application HTTPS isolée avec un pool dédié, dont vous consignez d'abord la liaison, le framework cible et l'état d'authentification. Inventoriez les autres modes d'IIS plutôt que de les ajouter : Basic envoie un identifiant facilement décodable et exige HTTPS

partout, Digest est un service de rôle hérité, et le mappage des certificats clients exige toujours une politique de confiance, de révocation et de mappage des comptes.

Lire le challenge et prouver le fournisseur

La première requête reçoit un 401 avec `WWW-Authenticate: Negotiate` et, s'il est configuré, `NTLM`. Un navigateur qui considère l'hôte comme éligible réessaie avec un en-tête `Authorization`, IIS valide l'échange, et la requête suivante renvoie 200. La fixture F01 du lab consigne exactement cela : un challenge Negotiate suivi d'un 200 est un challenge de l'hôte suivi d'une authentification, rien de plus. Negotiate est une enveloppe qui peut transporter Kerberos ou se replier sur NTLM selon le client, le SPN et la configuration du fournisseur ; un en-tête Negotiate seul ne prouve donc pas Kerberos.

Pour affirmer Kerberos, vous corrélez deux preuves : `klist` sur le client jetable montre un ticket de service pour `HTTP/accessops.contoso.test`, et l'enregistrement de connexion du serveur au même moment indique Kerberos comme package d'authentification. Sans les deux, notez le fournisseur comme indéterminé ; un lab en groupe de travail montre des challenges mais ne peut jamais étayer une affirmation Kerberos. Consignez les en-têtes uniquement par schéma et par statut, car les valeurs `Authorization` sont des identifiants et ne doivent finir ni dans une trace ni dans un ticket.

Fournisseurs, mode noyau, SPN et navigateur

La configuration délimitée du lab définit `<anonymousAuthentication enabled="false" />` et `<windowsAuthentication enabled="true" useKernelMode="true">`, avec la liste des fournisseurs vidée puis reconstruite avec Negotiate, puis NTLM. Gardez Negotiate en premier quand vous testez Kerberos, et testez le repli que vous voulez plutôt que de le déduire de l'ordre. Les attributs de persistance de connexion `authPersistNonNTLM` et `authPersistSingleRequest` décident combien de temps une connexion authentifiée reste fiable. D'après le pack et la documentation IIS de Microsoft, le mode noyau déchiffre normalement les tickets avec le compte machine ; quand le SPN HTTP est enregistré pour une identité de pool personnalisée, `useAppPoolCredentials` doit correspondre à cette topologie. Interrogez le SPN avec `setspn -Q` et ne modifiez les SPN qu'avec l'équipe de l'annuaire.

La protection étendue ajoute `tokenChecking`, `flags` et une liste de SPN sous `extendedProtection` ; décidez d'abord si TLS se termine sur IIS ou sur un intermédiaire de confiance, puis testez précisément ce chemin. Le navigateur a aussi son mot à dire : Edge n'envoie des identifiants silencieux qu'aux hôtes qu'il considère comme intranet ou que les administrateurs listent dans `AuthServerAllowlist` ; une machine jointe au domaine ne garantit donc pas la connexion silencieuse pour chaque FQDN, profil ou état VPN. NTLM et certaines configurations Negotiate exigent aussi une affinité de connexion, et avec le handler Negotiate de Kestrel (le package `Microsoft.AspNetCore.Authentication.Negotiate`, qui ne fait pas partie du framework partagé), Microsoft exige une affinité de connexion persistante un-à-un derrière un proxy. Un test direct contre le serveur est votre référence ; vérifiez le chemin par le proxy séparément.

Diagnostiquer couche par couche

Travaillez de l'extérieur vers l'intérieur. Un 401 avant `Program.Main` relève du challenge de l'hôte : comparez ce que le navigateur a vu avec les champs du journal IIS `sc-status`, `sc-substatus` et `sc-win32-status`, puis vérifiez le SPN, la politique du navigateur et le proxy. Un 500.19 juste après avoir placé des paramètres d'authentification dans `web.config` est une erreur de configuration d'IIS, généralement une section verrouillée à une portée parente : lisez le fichier et la section signalés, appliquez les mêmes

valeurs dans l'emplacement de niveau serveur ou via le Gestionnaire IIS, et ne touchez pas au code. Une page qui se charge avec une identité Windows valide mais refuse un document est un refus applicatif, et c'est le résultat attendu pour un compte sans l'octroi Reader. Une page qui fonctionne alors qu'un partage de fichiers ou un appel SQL échoue relève du second saut, le sujet du Module 3.

Une fois qu'IIS a authentifié la requête, Wisej.NET capture `HttpContext.User`, et `Application.User` le renvoie, sauf si la session a affecté son propre principal. `Application.UserIdentity` n'existe que dans le build .NET Framework. Le démarrage intranet d'AccessOps traite donc une identité Windows manquante comme une défaillance de l'hôte et recherche l'octroi séparément :

```
static void Main()
{
    // IIS answered the challenge; Wisej captured HttpContext.User.
    if (!Application.IsAuthenticated ||
        Application.User.Identity is not WindowsIdentity windows)
    {
        new HostFaultPage().Show();   // host layer, not an app deny
        return;
    }

    // The Windows account is evidence; the grant is the permission.
    var grant = AccessGrants.FindForAccount(windows.User?.Value);
    new DocumentsPage(grant).Show(); // DocumentService denies a null grant
}
```

Le lab prouve deux fois cette séparation : sur IIS dans un domaine Windows, un second compte sans Reader est refusé après une authentification réussie auprès de l'hôte ; sur n'importe quelle machine de développement, le PolicyHarness refuse une identité authentifiée dont l'octroi ne permet pas la lecture, à zéro lecture du repository. Sauvegardez les paramètres IIS d'origine pour que le retour arrière tienne en une étape.

Réflexe de sécurité

Gardez les identités séparées et décidez sur le serveur. Le principal de l'hôte, le principal applicatif Wisej.NET et le jeton Windows effectif sont trois faits différents, et aucun n'est une permission. Chaque lecture protégée vérifie l'octroi actuel, le tenant et l'enregistrement à la couche service avant que le repository ne s'exécute, et un bouton masqué, un claim de rôle ou un onglet déconnecté ne prouvent rien tant que cette vérification n'a pas refusé.

Objectifs d'apprentissage

À la fin de ce module, vous saurez :

- ✓ Choisir le mode d'authentification IIS de chaque point d'entrée d'AccessOps — intranet réservé à Windows, transport Anonymous pour la connexion publique et le callback — et garder les deux dans des applications ou des liaisons distinctes.

- ✓ Lire un challenge 401 Negotiate ou NTLM et prouver si Kerberos a été utilisé à partir de preuves corrélées (ticket et enregistrement de connexion du serveur) plutôt qu'à partir de l'en-tête de réponse.
- ✓ Configurer les fournisseurs de l'authentification Windows, le mode noyau, le SPN HTTP et la protection étendue, et expliquer comment la politique intranet du navigateur et l'affinité de connexion du proxy influent sur la connexion silencieuse.
- ✓ Diagnostiquer une défaillance couche par couche : challenge de l'hôte, verrou de configuration IIS (500.19), refus applicatif et accès au second saut.

Guide du lab / de l'examen

Ce que vous allez construire dans le lab pratique, l'approche suggérée et les livrables attendus.

Lab pratique

Publiez la route intranet d'AccessOps dans une application IIS HTTPS isolée avec un pool d'applications dédié, consignez la liaison, le framework cible et l'état d'authentification, puis appliquez les paramètres d'emplacement délimités windows-auth pour activer l'authentification Windows et désactiver Anonymous. Depuis un client du domaine, ouvrez le FQDN canonique en tant que CONTOSO\reader1, consignez le schéma du challenge, le résultat de la requête suivante et l'identité applicative masquée, interrogez le SPN HTTP et comparez le ticket du client avec les preuves de connexion du serveur, puis connectez-vous avec un second compte sans Reader et capturez un refus applicatif après une authentification réussie auprès de l'hôte, avant de restaurer les paramètres d'origine. Le challenge, le fournisseur négocié et les preuves SPN ne peuvent être vérifiés que dans l'environnement de lab avec IIS et un domaine Windows (un groupe de travail montre des challenges mais ne permet pas d'affirmer Kerberos : consignez-y donc un résultat indéterminé) ; sur une simple machine de développement, le PolicyHarness prouve localement qu'une identité authentifiée sans l'octroi Reader est toujours refusée, à zéro lecture du repository.

Point de départ

Ouvrez la solution AccessOps du Module 1, avec son `DocumentService`, son repository compteur de lectures, les fixtures des tenants A et B, et le PolicyHarness autonome. Il vous faut une application IIS HTTPS isolée, deux comptes de test du domaine, Edge et l'autorisation de lire les journaux IIS ; sans domaine, exécutez les étapes IIS dans un groupe de travail et les étapes de politique dans le PolicyHarness.

Approche suggérée

- ✓ Publiez la route intranet dans sa propre application IIS et son propre pool, et sauvegardez les paramètres d'authentification, la liaison et le framework cible actuels comme référence de retour arrière avant de modifier quoi que ce soit.
- ✓ Appliquez les paramètres d'emplacement délimités (Anonymous désactivé, Windows activé, mode noyau, fournisseurs vidés puis reconstruits avec Negotiate puis NTLM) ; si IIS répond 500.19, définissez les mêmes valeurs au niveau du serveur ou dans le Gestionnaire IIS au lieu de déverrouiller des sections.
- ✓ Ouvrez `https://accessops.contoso.test` depuis un client du domaine en tant que `CONTOSO\reader1` et consignez le schéma du 401, le statut de la requête suivante et l'identité masquée qu'affiche la vue de diagnostic, jamais la valeur `Authorization`.
- ✓ Exécutez `setspn -Q HTTP/accessops.contoso.test` et `klist`, faites correspondre le ticket du client avec l'enregistrement de connexion du serveur, et notez Kerberos, NTLM ou indéterminé à partir de ces seules preuves.
- ✓ Connectez-vous avec le second compte sans Reader, capturez le refus de `DocumentService` à zéro lecture du repository, ajoutez le cas PolicyHarness correspondant, puis restaurez les paramètres sauvegardés et confirmez que le site est sain.

Livrables attendus

- ✓ Paramètres d'authentification IIS effectifs pour les routes intranet et publique
- ✓ Trace de challenge masquée avec schéma, statut et résultat de la requête suivante
- ✓ Preuves SPN et ticket, ou un résultat explicite de fournisseur indéterminé
- ✓ Utilisateur authentifié par l'hôte refusé par la politique d'AccessOps, avec un cas PolicyHarness pour la même règle
- ✓ Note de diagnostic par couche distinguant une défaillance d'IIS d'un refus applicatif, avec les paramètres de retour arrière

Questions de révision

- ✓ Le navigateur montre un challenge Negotiate suivi d'un 200, et la vue de diagnostic affiche le bon nom de compte. Que pouvez-vous affirmer au sujet de Kerberos, et quelle preuve manque encore ?
- ✓ Un partenaire signale une invite d'identifiants Windows sur la page de connexion publique. Quel paramètre ou quelle frontière d'IIS inspecteriez-vous en premier, et pourquoi la correction ne se trouve-t-elle pas dans le code Wisej.NET ?
- ✓ La connexion silencieuse fonctionne quand vous accédez directement au serveur, mais boucle à travers l'équilibreur de charge. Quelles deux propriétés du chemin par le proxy compareriez-vous, et que casserait chacune d'elles ?

Réflexe de sécurité

Gardez les identités séparées et décidez sur le serveur. Le principal de l'hôte, le principal applicatif Wisej.NET et le jeton Windows effectif sont trois faits différents, et aucun n'est une permission. Chaque lecture protégée vérifie l'octroi actuel, le tenant et l'enregistrement à la couche service avant que le repository ne s'exécute, et un bouton masqué, un claim de rôle ou un onglet déconnecté ne prouvent rien tant que cette vérification n'a pas refusé.