PATTERN DI SICUREZZA E AUTENTICAZIONE · MODULO 2

Autenticazione integrata con IIS e browser

Guida della lezione · Guida al lab / esame

LearnWisej

Guida della lezione

Scegliere il confine del sito IIS, i provider dell'autenticazione Windows e l'ordine dei challenge, la negoziazione del browser e la policy intranet, gli SPN, le prove di Kerberos, la extended protection e i proxy, e diagnosticare un errore livello per livello. Che cosa significa per uno sviluppatore che mette in sicurezza i percorsi di accesso, identità e autorizzazione di un'app Wisej.NET di produzione — e come affrontare questo modulo.

Focus di produzione

Scegliere il confine del sito IIS, i provider dell'autenticazione Windows e l'ordine dei challenge, la negoziazione del browser e la policy intranet, gli SPN, le prove di Kerberos, la extended protection e i proxy, e diagnosticare un errore livello per livello.

1. Il problema in produzione

Il personale apre la route intranet di AccessOps e si aspetta di accedere in modo silenzioso con il proprio account Windows, mentre i partner raggiungono una pagina di accesso pubblica che non deve mai ricevere un challenge per credenziali Windows. Quando l'accesso silenzioso fallisce, il sintomo è un prompt, un ciclo di 401 o una pagina di errore, e la causa sta in IIS, nel browser, nella directory o in un proxy, prima che venga eseguito qualsiasi codice Wisej.NET. Questo modulo ti insegna a configurare quel confine dell'host in modo deliberato e a distinguere un errore dell'host da un rifiuto dell'applicazione.

2. Concetti chiave

Concetto	Perché è importante
confine del sito	Un'applicazione IIS o un binding separato per ogni punto di ingresso, così che la route intranet solo Windows e la route pubblica Anonymous portino ciascuna esattamente una policy di autenticazione visibile.
challenge Negotiate	Un HTTP 401 che porta `WWW-Authenticate: Negotiate` e/o `NTLM`; un browser idoneo riprova con un'intestazione `Authorization`, e Negotiate sceglie Kerberos oppure ripiega su NTLM.
prove di Kerberos	Un ticket di servizio per il nome host canonico `HTTP/` sul client, messo in correlazione con il record di logon del server; l'intestazione della risposta da sola non dimostra mai quale provider è stato eseguito.
autenticazione in kernel mode	Con `useKernelMode` è il driver HTTP.sys a eseguire l'handshake Windows, quindi l'account che decifra i ticket è normalmente l'account macchina, a meno che `useAppPoolCredentials` non dica altrimenti.
SPN	Il service principal name (`HTTP/accessops.contoso.test`) che dice al dominio per quale account è cifrato un ticket; uno SPN mancante o duplicato spinge in silenzio i client su NTLM o in un ciclo.
extended protection	Controlli di channel binding e service binding che legano l'handshake Windows alla connessione TLS e allo SPN atteso, e che contano non appena il TLS termina su un proxy.
principal dell'host in Wisej.NET	Su ASP.NET Core l'utente Windows raggiunge l'app solo come `Application.User`, l' `HttpContext.User` della richiesta catturato da Wisej.NET; dimostra chi ha effettuato l'accesso, non che cosa può leggere.

3. Il filo della lezione

Un'applicazione per ogni punto di ingresso

AccessOps ha due porte d'ingresso. La route intranet è per il personale del dominio: in IIS installi il servizio ruolo Windows Authentication, lo abiliti su quell'applicazione e lì disabiliti Anonymous, così che ogni richiesta riceva un challenge prima che l'ASP.NET Core Module la consegni all'app. La route pubblica serve la pagina di accesso dei partner e, nel Modulo 5, il callback OpenID Connect. Il browser del partner deve raggiungere quelle pagine prima che l'app sappia chi è il partner, quindi quell'applicazione mantiene abilitato il trasporto Anonymous ed è l'app Wisej.NET stessa a imporre l'accesso. Condividere un'unica applicazione significherebbe o mandare un challenge al callback esterno o aprire le schermate intranet.

Tieni le route in applicazioni IIS separate o in binding separati, ciascuno con il proprio application pool. Nel lab la route intranet è un'applicazione HTTPS isolata con un pool dedicato, di cui registri per prima cosa binding, target framework e stato dell'autenticazione. Fai un inventario delle altre modalità di IIS invece di aggiungerle: Basic invia una credenziale facilmente decodificabile e richiede HTTPS ovunque, Digest è un servizio ruolo legacy, e la mappatura dei certificati client richiede comunque una policy di fiducia, di revoca e di mappatura degli account.

Leggere il challenge e dimostrare il provider

La prima richiesta riceve un 401 con `WWW-Authenticate: Negotiate` e, se configurato, `NTLM`. Un browser che considera l'host idoneo riprova con un'intestazione `Authorization`, IIS convalida lo scambio e la richiesta successiva restituisce 200. La fixture F01 del lab registra esattamente questo: un challenge Negotiate seguito da 200 è un challenge dell'host seguito da un'autenticazione, e niente di più. Negotiate è un involucro che può trasportare Kerberos o ripiegare su NTLM a seconda del client, dello SPN e della configurazione del provider, quindi un'intestazione Negotiate da sola non dimostra Kerberos.

Per affermare che si tratta di Kerberos metti in correlazione due prove: `klist` sul client usa e getta mostra un ticket di servizio per `HTTP/accessops.contoso.test`, e il record di logon del server per lo stesso istante indica Kerberos come pacchetto di autenticazione. Senza entrambe, annota il provider come indeterminato; un lab in workgroup mostra i challenge ma non può mai sostenere un'affermazione su Kerberos. Registra le intestazioni solo per schema e stato, perché i valori di `Authorization` sono credenziali e non devono finire in una traccia o in un ticket.

Provider, kernel mode, SPN e il browser

La configurazione circoscritta del lab imposta `<anonymousAuthentication enabled="false" />` e `<windowsAuthentication enabled="true" useKernelMode="true">` con l'elenco dei provider svuotato e ricostruito come Negotiate, poi NTLM. Tieni Negotiate per primo quando provi Kerberos, e prova il ripiego che intendi invece di dedurlo dall'ordine. Gli attributi di persistenza della connessione `authPersistNonNTLM` e `authPersistSingleRequest` decidono per quanto tempo ci si fida di una connessione autenticata. Secondo il pacchetto e la documentazione Microsoft di IIS, il kernel mode decifra normalmente i ticket con l'account macchina; quando lo SPN HTTP è registrato su un'identità di pool personalizzata, `useAppPoolCredentials` deve corrispondere a quella topologia. Interroga lo SPN con `setspn -Q` e modifica gli SPN solo insieme al team della directory.

La extended protection aggiunge `tokenChecking`, `flags` e un elenco di SPN sotto `extendedProtection`; decidi prima se il TLS termina su IIS o su un intermediario attendibile, poi prova lungo esattamente quel percorso. Anche il browser ha voce in capitolo: Edge invia credenziali in modo silenzioso solo agli host che considera intranet o che gli amministratori elencano in `AuthServerAllowlist`, quindi una macchina aggiunta al dominio non garantisce l'accesso silenzioso per ogni FQDN, profilo o stato della VPN. NTLM e alcune configurazioni Negotiate richiedono anche l'affinità di connessione, e con l'handler Negotiate di Kestrel (il pacchetto `Microsoft.AspNetCore.Authentication.Negotiate`, che non fa parte dello shared framework) Microsoft richiede dietro un proxy un'affinità uno a uno delle connessioni persistenti. Una prova diretta contro il server è il tuo riferimento; verifica separatamente il percorso attraverso il proxy.

Diagnosticare livello per livello

Lavora dall'esterno verso l'interno. Un 401 prima di `Program.Main` appartiene al challenge dell'host: confronta ciò che ha visto il browser con i campi del log di IIS `sc-status`, `sc-substatus` e `sc-win32-status`, poi controlla lo SPN, la policy del browser e il proxy. Un 500.19 subito dopo aver messo le impostazioni di autenticazione in `web.config` è un errore di configurazione di IIS, di solito una sezione bloccata a un livello superiore: leggi il file e la sezione segnalati, applica gli stessi valori nella location a livello di server o tramite IIS Manager, e lascia stare il codice. Una pagina che si carica con un'identità Windows valida ma rifiuta un documento è un rifiuto dell'applicazione, ed è l'esito che vuoi per un account

senza il grant Reader. Una pagina che funziona mentre fallisce una chiamata a una condivisione di file o a SQL è il secondo hop, l'argomento del Modulo 3.

Una volta che IIS ha autenticato la richiesta, Wisej.NET cattura `HttpContext.User`, e `Application.User` lo restituisce a meno che la sessione non abbia assegnato un proprio principal. `Application.UserIdentity` esiste solo nella build per .NET Framework. L'avvio intranet di AccessOps tratta quindi un'identità Windows mancante come un guasto dell'host e cerca il grant separatamente:

```
static void Main()
{
    // IIS ha risposto al challenge; Wisej ha catturato HttpContext.User.
    if (!Application.IsAuthenticated ||
        Application.User.Identity is not WindowsIdentity windows)
    {
        new HostFaultPage().Show();   // livello dell'host, non un rifiuto dell'app
        return;
    }

    // L'account Windows è una prova; il grant è il permesso.
    var grant = AccessGrants.FindForAccount(windows.User?.Value);
    new DocumentsPage(grant).Show(); // DocumentService rifiuta un grant null
}
```

Il lab dimostra questa separazione due volte: su IIS in un dominio Windows, un secondo account senza Reader viene rifiutato dopo un'autenticazione dell'host riuscita; su qualsiasi macchina di sviluppo, il PolicyHarness rifiuta un'identità autenticata il cui grant non consente la lettura, con zero letture dal repository. Salva le impostazioni originali di IIS così che il rollback sia un solo passo.

Regola di sicurezza

Tieni separate le identità e decidi sul server. Il principal dell'host, il principal applicativo di Wisej.NET e il token Windows effettivo sono tre fatti diversi, e nessuno di essi è un permesso. Ogni lettura protetta controlla il grant corrente, il tenant e il record al confine del servizio prima che il repository venga eseguito, e un pulsante nascosto, un claim di ruolo o una scheda disconnessa non dimostrano nulla finché quel controllo non rifiuta.

Obiettivi di apprendimento

Alla fine di questo modulo sarai in grado di:

- ✓ Scegliere la modalità di autenticazione IIS per ogni punto di ingresso di AccessOps — intranet solo Windows, trasporto Anonymous per l'accesso pubblico e il callback — e tenere le due in applicazioni o binding separati.
- ✓ Leggere un challenge 401 Negotiate o NTLM e dimostrare se è stato usato Kerberos a partire da prove correlate tra ticket e logon del server, invece che dall'intestazione della risposta.
- ✓ Configurare i provider dell'autenticazione Windows, il kernel mode, lo SPN HTTP e la extended protection, e spiegare in che modo la policy intranet del browser e l'affinità di connessione del

proxy influenzano l'accesso silenzioso.

- ✓ Diagnosticare un errore livello per livello: challenge dell'host, blocco della configurazione di IIS (500.19), rifiuto dell'applicazione e accesso al secondo hop.

Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.

Lab pratico

Pubblica la route intranet di AccessOps in un'applicazione IIS HTTPS isolata con un application pool dedicato, registra binding, target framework e stato dell'autenticazione, poi applica le impostazioni circoscritte della location windows-auth per abilitare l'autenticazione Windows e disabilitare Anonymous. Da un client del dominio apri l'FQDN canonico come CONTOSO\reader1, registra lo schema del challenge, l'esito della richiesta successiva e l'identità applicativa oscurata, interroga lo SPN HTTP e confronta il ticket del client con le prove di logon del server, poi accedi con un secondo account senza Reader e cattura un rifiuto dell'applicazione dopo un'autenticazione dell'host riuscita, prima di ripristinare le impostazioni originali. Il challenge, il provider negoziato e le prove sullo SPN si possono verificare solo nell'ambiente di lab con IIS e un dominio Windows (un workgroup mostra i challenge ma non permette di affermare Kerberos, quindi lì registra un risultato indeterminato); su una semplice macchina di sviluppo il PolicyHarness dimostra in locale che un'identità autenticata senza il grant Reader viene comunque rifiutata con zero letture dal repository.

Punto di partenza

Apri la soluzione AccessOps del Modulo 1, con il suo `DocumentService`, il repository che conta le letture, le fixture dei tenant A e B e il PolicyHarness autonomo. Ti servono un'applicazione IIS HTTPS isolata, due account di test del dominio, Edge e il permesso di leggere i log di IIS; senza un dominio, esegui i passi IIS in un workgroup e quelli sulla policy nel PolicyHarness.

Approccio suggerito

- ✓ Pubblica la route intranet in un'applicazione e in un pool IIS dedicati, e salva le impostazioni di autenticazione correnti, il binding e il target framework come base per il rollback prima di modificare qualsiasi cosa.
- ✓ Applica le impostazioni circoscritte della location (Anonymous disattivato, Windows attivato, kernel mode, provider svuotati e ricostruiti come Negotiate poi NTLM); se IIS risponde 500.19, imposta gli stessi valori a livello di server o in IIS Manager invece di sbloccare le sezioni.
- ✓ Apri `https://accessops.contoso.test` da un client del dominio come `CONTOSO\reader1` e registra lo schema del 401, lo stato della richiesta successiva e l'identità oscurata che mostra la vista diagnostica, mai il valore di `Authorization`.
- ✓ Esegui `setspn -Q HTTP/accessops.contoso.test` e `klist`, fai corrispondere il ticket del client al record di logon del server e scrivi Kerberos, NTLM o indeterminato basandoti solo su quelle prove.
- ✓ Accedi con il secondo account senza Reader, cattura il rifiuto di `DocumentService` con zero letture dal repository, aggiungi il caso corrispondente nel PolicyHarness, poi ripristina le impostazioni salvate e conferma che il sito funziona correttamente.

Deliverable richiesti

- ✓ Impostazioni di autenticazione IIS effettive per le route intranet e pubblica
- ✓ Traccia oscurata del challenge con schema, stato ed esito della richiesta successiva
- ✓ Prove su SPN e ticket oppure un risultato esplicito di provider indeterminato
- ✓ Utente autenticato dall'host rifiutato dalla policy di AccessOps, con un caso PolicyHarness per la stessa regola
- ✓ Nota di diagnosi per livelli che separa l'errore di IIS dal rifiuto dell'applicazione, con le impostazioni di rollback

Domande di autovalutazione

- ✓ Il browser mostra un challenge Negotiate seguito da 200 e la vista diagnostica mostra il nome dell'account corretto. Che cosa puoi affermare su Kerberos, e quale prova manca ancora?
- ✓ Un partner segnala una richiesta di credenziali Windows sulla pagina di accesso pubblica. Quale impostazione o confine di IIS controlleresti per primo, e perché la correzione non sta nel codice Wisej.NET?
- ✓ L'accesso silenzioso funziona quando navighi direttamente sul server ma va in ciclo passando dal load balancer. Quali due proprietà del percorso attraverso il proxy confronteresti, e che cosa romperebbe ciascuna?

Regola di sicurezza

Tieni separate le identità e decidi sul server. Il principal dell'host, il principal applicativo di Wisej.NET e il token Windows effettivo sono tre fatti diversi, e nessuno di essi è un permesso. Ogni lettura protetta controlla il grant corrente, il tenant e il record al confine del servizio prima che il repository venga eseguito, e un pulsante nascosto, un claim di ruolo o una scheda disconnessa non dimostrano nulla finché quel controllo non rifiuta.