

ADFS-Ereignisprotokolle lesen – Admin-Log, Debug-Tracing und Auditing

ADFS · Protokolle & Diagnose

Was ADFS dir sagen will – wenn du zuhörst

Admin-Log, Debug-Tracing, Sicherheitsüberwachung und die eine GUID, die alles zusammenhält.

boddenberg.de

Ereignisanzeige – AD FS/Admin

● Fehler	07:58:12	AD FS
● Fehler	07:58:11	AD FS
● Warnung	07:41:03	AD FS
● Information	07:30:00	AD FS
● Information	07:12:47	AD FS

ActivityID:
{00000000-0000-0000-4b1d-0080000000a7}
Exception: MSIS... (Details im Debug-Log)

WISSEN

Grundlagen, Architektur und alle Praxisbeiträge rund um ADFS an einem Ort.

› [Active Directory Federation Services](#)

BERATUNG

Die Logs schweigen, die Anwender nicht? Wir lesen mit dir mit – und richten das Auditing gleich dauerhaft sauber ein.

› [Consulting zu ADFS](#)

SCHULUNG

Im Labor Fehler provozieren, Debug-Tracing laufen lassen und die Activity ID durch die Farm jagen.

› [ADFS-Schulungen](#)

ADFS ist ein erstaunlich gesprächiger Dienst. Er erzählt dir ziemlich genau, warum eine Anmeldung gescheitert ist, welches Zertifikat ihm Sorgen macht und wer sich um 3:12 Uhr nachts mit dem vierzigsten falschen Kennwort versucht hat. Das Problem: Er erzählt es an drei verschiedenen Stellen, zwei davon sind standardmäßig stumm geschaltet, und die dritte steht auf jedem Knoten der Farm separat. Das ist ungefähr so, als würde der Hausmeister seine Mängelliste in drei verschlossene Schubladen in drei verschiedenen Gebäuden legen – und sich dann wundern, dass niemand reagiert.

Wer „adfs debug log aktivieren“ in die Suchmaschine tippt, hat meistens schon ein Problem, und zwar ein akutes. Dieser Beitrag hilft dir in genau dieser Lage: Er zeigt, welche Protokolle es gibt, wie du das Debug-Tracing gezielt ein- und – mindestens genauso wichtig – wieder abschaltest, wie die Sicherheitsüberwachung mit Audit Levels und der Richtlinie „Anwendung generiert“ zusammenspielt und wie du eine einzelne Anmeldung über die Activity ID quer durch Web Application Proxy und Farm verfolgst. Die Grundlagen zu Farm, Relying Parties und Proxy findest du auf der Übersichtsseite [Active Directory Federation Services](#).

FAKTEN – ADFS-Protokolle auf einen Blick

AD FS/Admin ist standardmäßig aktiv und liefert Fehler, Warnungen und Informationen auf hohem Niveau.

AD FS Tracing/Debug ist standardmäßig aus, weil es in kurzer Zeit sehr viele Einträge erzeugt und Leistung kostet.

Die Sicherheitsüberwachung des ADFS-Dienstkontos ist ebenfalls aus. Sie braucht drei Voraussetzungen: das Benutzerrecht „Generieren von Sicherheitsüberwachungen“, die Unterkategorie „Anwendung generiert“ und die Erfolgs- und Fehlerüberwachung in den ADFS-Eigenschaften.

Seit Windows Server 2016 steuert Set-AdfsProperties -AuditLevel den Umfang: None, Basic (Standard, höchstens fünf Ereignisse je Anfrage) oder Verbose.

Alle Ereignisse einer Anfrage tragen dieselbe Activity ID – im Admin-Log, im Debug-Log und auf dem Proxy. Dieselbe ID zeigt die Fehlerseite im Browser.

Welche Protokolle ADFS schreibt – und welche davon überhaupt laufen

Microsoft unterscheidet beim Troubleshooting zwei primäre Protokolle: das Admin-Log und das Tracelog, in der Ereignisanzeige als „AD FS Tracing/Debug“ geführt. Dazu kommt als dritte Quelle das Windows-Sicherheitsprotokoll, in das ADFS seine Überwachungsereignisse schreibt – sofern man es lässt. Jedes der drei Protokolle hat einen eigenen Charakter: Das Admin-Log ist der Pförtner, der dir sagt, dass etwas schiefging. Das Debug-Log ist der Kollege, der dir jede Einzelheit erzählt, ob du willst oder nicht. Und das Sicherheitsprotokoll ist der Notar, der festhält, wer wann was bekommen hat.

Skizze 1: Drei Protokolle, drei Charaktere

Wo ADFS schreibt, was es schreibt – und ob es das von Haus aus tut



Skizze 1: Admin-Log, Debug-Log und Sicherheitsprotokoll im Vergleich

Protokoll	Pfad in der Ereignisanzeige	Standard	Inhalt	Wann du es liest
AD FS/Admin	Anwendungs- und Dienstprotokolle › AD FS › Admin	aktiv	Fehler, Warnungen, Informationen, fehlgeschlagene Anfragen mit Exception	immer zuerst
AD FS Tracing/Debug	Anwendungs- und Dienstprotokolle › AD FS Tracing › Debug (erst nach Einblenden sichtbar)	inaktiv	jeder Verarbeitungsschritt einer Anfrage, Regeln, Protokolldetails	wenn das Admin-Log nicht reicht – gezielt und befristet
Sicherheit	Windows-Protokolle › Sicherheit, Quelle „AD FS Auditing“	inaktiv	Anmeldungen, ausgestellte und verweigte Token, Kennwortänderungen, Abmeldungen	Forensik, Revision, Angriffserkennung, Connect Health
Web Application Proxy/Admin	Anwendungs- und Dienstprotokolle › Microsoft › Windows › Web Application Proxy	aktiv (auf dem WAP)	veröffentlichte Anwendungen, Backend-Verbindungen, Konfiguration	wenn nur der externe Zugriff scheitert

Tabelle 1: Die Protokolle rund um eine ADFS-Anmeldung

Das Admin-Log: der Pförtner

Das Admin-Log ist ab Werk eingeschaltet und für den Alltag die wichtigste Quelle. Hier landen Warnungen zu ablaufenden Zertifikaten, Probleme beim Dienststart, Fehler beim Abruf von Partner-Metadaten und vor allem fehlgeschlagene Anfragen. Für Fehler bei passiven Anfragen, also bei Anmeldungen über den Browser, ist Ereignis 364 der bekannte Sammelbehälter. Der Kopf des Ereignisses ist dabei fast egal – die eigentliche Ursache steht weiter unten in der Exception, oft mit einer MSIS-Fehlernummer und einem Satz, der überraschend konkret sagt, was ADFS gestört hat.

Wichtig für Farmen: Jeder Knoten führt sein eigenes Admin-Log. Es gibt keine zentrale Sammelstelle in ADFS selbst. Steht ein Load Balancer davor, weißt du zunächst nicht, welcher Knoten die fehlgeschlagene Anfrage bearbeitet hat. Also suchst du auf allen – oder du hast die Logs vorher in ein zentrales System weitergeleitet, was ohnehin die erwachsene Lösung ist. Wie die Lastverteilung die Fehlersuche beeinflusst, beschreibt der Beitrag [ADFS hinter dem Load Balancer – Health Probe, SNI und Persistenz](#).

```
# Welche ADFS- und WAP-Protokolle gibt es auf diesem Server, und wie groß sind sie?
Get-WinEvent -ListLog "AD FS*", "*Web Application Proxy*" -ErrorAction SilentlyContinue |
    Select-Object LogName, IsEnabled, RecordCount, MaximumSizeInBytes

# Die letzten 20 Fehler aus dem Admin-Log
Get-WinEvent -LogName "AD FS/Admin" -MaxEvents 200 |
    Where-Object LevelDisplayName -eq "Fehler" |
    Select-Object -First 20 TimeCreated, Id, Message
```

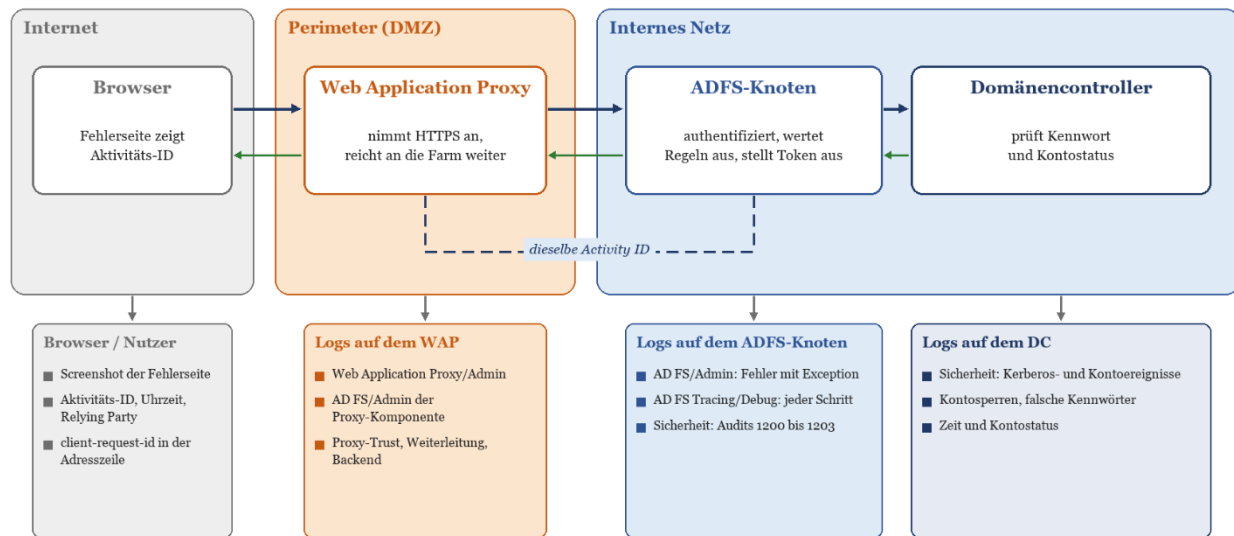
Bestandsaufnahme der Protokolle. Auf englischen Systemen lautet der Level-Name „Error“.

Die Logs auf dem Web Application Proxy

Der Web Application Proxy hat zwei Rollen in einem Gehäuse: Er veröffentlicht Anwendungen, und er spielt den ADFS-Proxy für externe Anmeldungen. Entsprechend findest du auf ihm sowohl das Protokoll des Web Application Proxy als auch ein AD FS-Protokoll für die Proxy-Komponente. Dort tauchen die Klassiker auf: Der Proxy kann seine Konfiguration nicht von der Farm laden, das Proxy-Trust-Zertifikat ist nicht mehr gültig, das Backend antwortet nicht. Wenn intern alles läuft und extern nichts, ist der WAP die erste Adresse. Die Architektur dahinter erklärt der Beitrag [Web Application Proxy: Architektur und Ablöse](#), die Reparatur eines kaputten Vertrauens steht in [WAP-Proxy-Vertrauen erneuern – wenn der Web Application Proxy plötzlich nicht mehr will](#).

Skizze 2: Der Weg einer Anmeldung durch WAP- und ADFS-Logs

Eine externe Anmeldung, vier Stationen – und an jeder bleibt eine Spur zurück



Skizze 2: Der Weg einer externen Anmeldung durch WAP- und ADFS-Logs

Die Skizze zeigt den Grundgedanken, der die Fehlersuche abkürzt: Jede Station hinterlässt eine Spur, und die Spuren lassen sich über die Activity ID verbinden. Kommt die Anfrage im Proxy-Log an, aber auf keinem ADFS-Knoten, liegt das Problem zwischen DMZ und internem Netz – Firewall, Name, Zertifikat. Taucht sie auf einem Knoten mit einer Exception auf, ist der Proxy unschuldig, und du kannst dich auf Regeln, Trusts und Konten konzentrieren.

WARNUNG – Die Uhr lügt nicht, aber sie geht vielleicht falsch

Wenn du Ereignisse über mehrere Server hinweg nach Uhrzeit vergleichst, müssen die Uhren stimmen. Ein WAP außerhalb der Domäne, der seine Zeit von irgendwoher bezieht, macht jede zeitliche Korrelation zum Ratespiel – und produziert nebenbei selbst Anmeldefehler. Die Activity ID ist deshalb das bessere Suchkriterium als die Uhrzeit.

ADFS Debug Log aktivieren – und wieder abschalten

Das Debug-Log ist der Grund, warum die meisten hier gelandet sind. Microsoft beschreibt es selbst als das nützlichste Protokoll bei der Fehlersuche – und schaltet es trotzdem standardmäßig ab, weil es in kurzer Zeit so viele Einträge erzeugt, dass es die Leistung des Systems beeinträchtigen kann. Das ist keine Übervorsicht. Eine Farm, die Microsoft 365 für ein paar tausend Leute bedient, schreibt im Debug-Modus in einer Viertelstunde mehr, als ein Mensch in einer Woche liest. Deshalb gilt die Regel: Debug-Tracing ist eine Taschenlampe, kein Flutlicht.

Per Ereignisanzeige einschalten

Der Weg über die Oberfläche ist schnell erklärt, hat aber einen kleinen Haken: Das Protokoll ist zunächst unsichtbar.

- Ereignisanzeige öffnen und „Anwendungs- und Dienstprotokolle“ aufklappen.
- Im Menü „Ansicht“ die Option „Analyse- und Debugprotokolle einblenden“ aktivieren. Erst jetzt erscheinen zusätzliche Knoten.
- „AD FS Tracing“ aufklappen, mit der rechten Maustaste auf „Debug“ klicken und „Protokoll aktivieren“ wählen.
- Fehler reproduzieren – möglichst genau einmal, mit notierter Uhrzeit und Activity ID.
- Wieder rechte Maustaste auf „Debug“ und „Protokoll deaktivieren“. Danach lesen oder exportieren.

Per wevtutil einschalten – reproduzierbar und skriptbar

Wer Änderungen lieber per Befehl dokumentiert, nimmt wevtutil. Das hat zwei Vorteile: Der Befehl steht im Change-Ticket, und das Ausschalten lässt sich gleich mit einplanen. Der Schalter /q:true unterdrückt die Rückfrage, die Windows beim Aktivieren eines Debug-Kanals sonst stellt.

```
# Debug-Log vergrößern (hier 200 MB) und einschalten
wevtutil sl "AD FS Tracing/Debug" /ms:209715200
wevtutil sl "AD FS Tracing/Debug" /e:true /q:true

# ... Fehler reproduzieren, Uhrzeit und Activity ID notieren ...

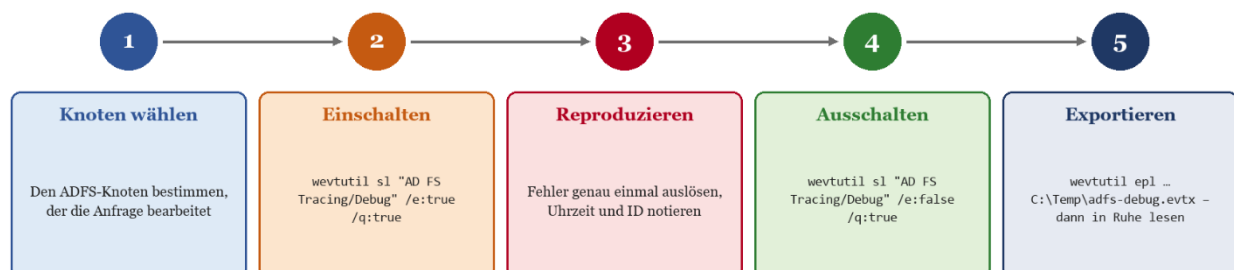
# Wieder ausschalten und als Datei sichern
wevtutil sl "AD FS Tracing/Debug" /e:false /q:true
wevtutil epl "AD FS Tracing/Debug" C:\Temp\adfs-debug-sts01.evtx

# Status prüfen
wevtutil gl "AD FS Tracing/Debug"
```

Ein- und Ausschalten des Debug-Tracings auf einem ADFS-Knoten

Skizze 3: Debug-Tracing – rein, reproduzieren, raus

Ein Debug-Log ist eine Taschenlampe, kein Flutlicht. Fünf Schritte, dann ist wieder Ruhe.



Vergessenes Debug-Tracing kostet Leistung und läuft voll – und dann fehlen genau die Einträge, die du morgen brauchst. Schritt 4 gehört in denselben Change wie Schritt 2.

Skizze 3: Der Lebenszyklus eines Debug-Tracings in fünf Schritten

WICHTIG – Debug-Kanäle laufen voll statt im Kreis

Analyse- und Debugprotokolle verhalten sich anders als das Admin-Log: Sie sind auf sequenzielles Schreiben ausgelegt. Läuft das Protokoll voll, kommen keine neuen Einträge mehr dazu – und genau der Fehler, den du reproduzieren wolltest, fehlt. Vergrößere das Protokoll vor dem Test und exportiere den Inhalt, bevor du es erneut aktivierst, denn beim erneuten Einschalten kann der alte Inhalt verworfen werden.

Das Debug-Log lesen, ohne darin zu ertrinken

Die Ereignisanzeige ist für Debug-Logs ein mühsames Werkzeug. Besser filterst du gezielt per PowerShell. Eine Besonderheit solltest du kennen: Get-WinEvent liest Analyse- und Debugkanäle nur mit dem Schalter -Oldest, also in chronologischer Reihenfolge. Ohne ihn gibt es eine Fehlermeldung statt Ereignisse.

```
$id = "00000000-0000-0000-4b1d-0080000000a7" # Activity ID von der Fehlerseite
$xp = "[System[Correlation[@ActivityID='{ $id }']]]"

# Aus dem exportierten Debug-Log
Get-WinEvent -Path C:\Temp\adfs-debug-sts01.evtx -FilterXPath $xp -Oldest |
    Select-Object TimeCreated, Id, Message | Format-List

# Direkt aus dem (deaktivierten) Debug-Kanal
Get-WinEvent -LogName "AD FS Tracing/Debug" -FilterXPath $xp -Oldest
```

Nur die Einträge einer einzigen Anfrage aus dem Debug-Log holen

TIPP — Nur auf einem Knoten tracen

Debug-Tracing auf allen Knoten einer Farm gleichzeitig ist der sichere Weg, viele Daten und wenig Erkenntnis zu sammeln. Ermittle über das Admin-Log zuerst, welcher Knoten die Anfrage bearbeitet hat. Wenn dein Load Balancer es erlaubt, nimmst du die übrigen Knoten für die Dauer des Tests aus der Verteilung oder pinnst den Testclient per hosts-Datei auf den einen Knoten. Dann landet die Reproduktion garantiert dort, wo du zuhörst.

Die Stufe darunter: WIF- und WCF-Tracing

Für ganz hartnäckige Fälle, etwa bei Problemen mit Zertifikatsprüfungen oder WS-Trust-Clients, reicht selbst das Debug-Log nicht. Dann lassen sich zusätzlich Meldungen von Windows Identity Foundation und Windows Communication Foundation einschalten. Das geschieht in der Datei Microsoft.IdentityServer.ServiceHost.Exe.Config im Verzeichnis C:\Windows\ADFS: Bei den Einträgen „Microsoft.IdentityModel“ und „System.ServiceModel“ wird der Wert switchValue von „Off“ auf eine Stufe wie „Verbose“ gesetzt. Nach einem Neustart des ADFS-Dienstes erscheinen die Meldungen im Tracelog.

WARNUNG — Konfigurationsdatei ist kein Spielplatz

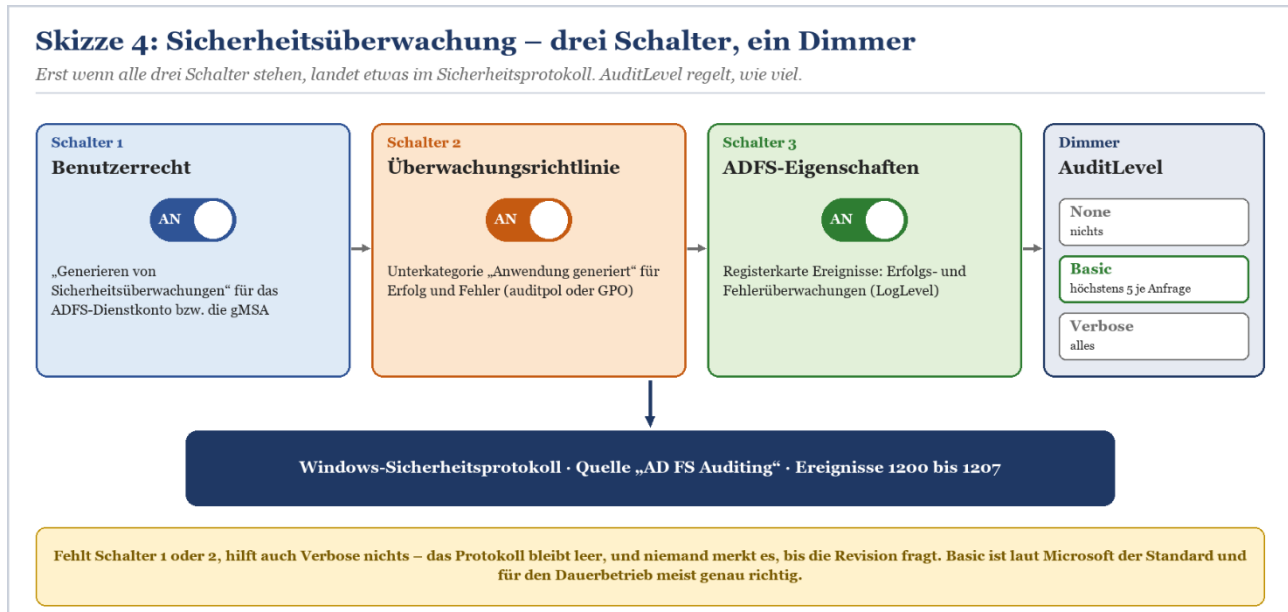
Sichere die Datei vor jeder Änderung, ändere sie nur auf dem betroffenen Knoten, und dreh die Werte nach dem Test zurück. Ein Tippfehler im XML verhindert den Dienststart – und dann hast du statt eines Anmeldeproblems einen Knoten weniger. Der Dienstneustart unterbricht laufende Anmeldungen auf diesem Knoten, also lieber außerhalb der Hauptzeit oder mit dem Knoten außerhalb der Lastverteilung.

Werkzeug	Gut für	Aufwand	Risiko
Admin-Log	Erstdiagnose, Zertifikate, Dienststatus, Exception zur Activity ID	keiner, läuft immer	keins
Debug-Log	Ablauf einer einzelnen Anfrage, Regelauswertung, Protokolldetails	Ein- und Ausschalten pro Knoten	Leistung, volles Protokoll
WIF-/WCF-Tracing	Zertifikatsketten, WS-Trust, Transportprobleme	Datei ändern, Dienst neu starten	Dienststart scheitert bei Fehlern im XML
Sicherheitsprotokoll	wer, wann, welches Ergebnis; Angriffserkennung	einmalig drei Schalter	Protokollgröße, Aufbewahrung

Tabelle 2: Welches Werkzeug für welche Frage

Sicherheitsüberwachung: Audit Levels und die Richtlinie „Anwendung generiert“

Das Admin-Log sagt dir, dass etwas schiefging. Das Sicherheitsprotokoll sagt dir, wem. Für Forensik, Revision und die Erkennung von Angriffen ist es deshalb unverzichtbar – und für Microsoft Entra Connect Health für ADFS ebenfalls, denn dessen Auswertungen zu Anmeldungen und fehlgeschlagenen Versuchen speisen sich aus den Überwachungsereignissen. Die schlechte Nachricht: Die Sicherheitsüberwachung des ADFS-Dienstkontos ist standardmäßig aus, und sie einzuschalten braucht drei Schalter an drei verschiedenen Stellen. Fehlt einer, bleibt das Protokoll leer. Ohne Fehlermeldung, ohne Warnung, einfach still. Das merkt man typischerweise genau dann, wenn man es dringend braucht.



Skizze 4: Drei Schalter für die Sicherheitsüberwachung, AuditLevel als Dimmer

Schalter 1: Das Benutzerrecht für das Dienstkonto

Das ADFS-Dienstkonto braucht das Benutzerrecht „Generieren von Sicherheitsüberwachungen“ (Generate security audits). Du findest es in der lokalen Sicherheitsrichtlinie unter Sicherheitseinstellungen > Lokale Richtlinien > Zuweisen von Benutzerrechten. Bei einer Neuinstallation wird es in der Regel gesetzt. Gefährlich wird es, wenn eine Gruppenrichtlinie dieses Recht zentral verwaltet und das ADFS-Konto dort fehlt – dann überschreibt die GPO bei der nächsten Aktualisierung die lokale Zuweisung, und das Auditing ist weg. Besonders gern passiert das nach einer Umstellung auf ein gruppenverwaltetes Dienstkonto, siehe [gMSA als ADFS-Dienstkonto einrichten und umstellen](#). Läuft ADFS auf einem Domänencontroller, was du ohnehin vermeiden solltest, gehört die Zuweisung in die Default Domain Controllers Policy.

Schalter 2: Die Unterkategorie „Anwendung generiert“

Windows muss Ereignisse, die eine Anwendung selbst erzeugt, überhaupt ins Sicherheitsprotokoll durchlassen. Zuständig ist die Unterkategorie „Anwendung generiert“ (Application Generated) im Bereich Objektzugriff der erweiterten Überwachungsrichtlinie. Microsoft dokumentiert dafür einen einzigen Befehl:

```
auditpol.exe /set /subcategory:"Application Generated" /success:enable /failure:enable

# Kontrolle
```

```
auditpol.exe /get /subcategory:"Application Generated"
```

Überwachung „Anwendung generiert“ für Erfolg und Fehler einschalten. Der englische Name funktioniert auch auf deutschen Systemen.

Auch hier lauert die Gruppenrichtlinie: Gibt es in deiner Domäne eine GPO mit erweiterter Überwachungsrichtlinienkonfiguration, die die ADFS-Server erreicht, gewinnt sie gegen den lokalen auditpol-Befehl. Dann trägst du „Anwendung generiert überwachen“ in dieser GPO ein – am besten in einer eigenen, die nur auf die Organisationseinheit der ADFS-Server zielt.

Schalter 3: Erfolgs- und Fehlerüberwachung in ADFS

Zum Schluss muss ADFS selbst überwachen wollen. In der ADFS-Verwaltung öffnest du „Eigenschaften des Verbunddiensts bearbeiten“ und setzt auf der Registerkarte „Ereignisse“ die Häkchen bei Erfolgs- und Fehlerüberwachungen. Per PowerShell ist das die Eigenschaft LogLevel. Vorsicht: LogLevel ist ein Array, und Set-AdfsProperties ersetzt es komplett. Wer nur die beiden Audit-Werte übergibt, wirft dabei Fehler, Warnungen und Informationen aus dem Admin-Log. Also erst lesen, dann ergänzen:

```
# Aktuellen Stand anzeigen
Get-AdfsProperties | Select-Object AuditLevel, LogLevel

# Erfolgs- und Fehlerüberwachung ergänzen, ohne die übrigen Stufen zu verlieren
$lvl = @((Get-AdfsProperties).LogLevel) + "SuccessAudits", "FailureAudits" | Select-Object -Unique
Set-AdfsProperties -LogLevel $lvl
```

LogLevel ergänzen statt überschreiben. Die Einstellung gilt farmweit.

Der Dimmer: AuditLevel None, Basic und Verbose

Seit Windows Server 2016 regelt Set-AdfsProperties -AuditLevel, wie gesprächig die Überwachung ist. Microsoft hat dabei die Menge pro Anmeldung kräftig reduziert: Auf der Standardstufe Basic entstehen höchstens fünf Ereignisse je Anfrage. Das ist ein Segen für jedes SIEM und für jeden, der Lizenzen nach Datenvolumen bezahlt.

AuditLevel	Befehl	Wirkung	Einsatz
None	Set-AdfsProperties -AuditLevel None	keine Überwachungsereignisse	praktisch nie sinnvoll
Basic	Set-AdfsProperties -AuditLevel Basic	höchstens fünf Ereignisse je Anfrage (Standard)	Dauerbetrieb
Verbose	Set-AdfsProperties -AuditLevel Verbose	alle Ereignisse, sehr viel Information je Anfrage	befristet für Analysen, danach zurück auf Basic

Tabelle 3: Die drei Audit Levels laut Microsoft-Dokumentation

Welche Ereignisse du im Sicherheitsprotokoll erwarten kannst, dokumentiert Microsoft für die Überwachung ab Windows Server 2016 so:

Ereignis-ID	Bedeutung	Praxisnutzen
1200	Token für eine Anwendung erfolgreich ausgestellt (bei WS-Federation und SAML-P mit SSO-Artefakt, etwa dem SSO-Cookie)	wer hat wann welche Anwendung genutzt
1201	Ausstellung eines Tokens fehlgeschlagen	Regeln, Richtlinien, Zugriff verweigert
1202	Frische Anmeldedaten erfolgreich geprüft (WS-Trust, WS-Federation, SAML-P, OAuth Authorize)	echte Anmeldungen mit Kennwort
1203	Prüfung frischer Anmeldedaten fehlgeschlagen	falsche Kennwörter, Passwort-Spray, Sperren
1204 / 1205	Kennwortänderung erfolgreich / fehlgeschlagen	Seite „Kennwort aktualisieren“
1206 / 1207	Abmeldung erfolgreich / fehlgeschlagen	Sign-out-Probleme bei SAML-

Tabelle 4: Ereignistypen der ADFS-Überwachung ab Windows Server 2016

FAKTEN – Was im Sicherheitsprotokoll steht – und was nicht

Die Überwachungsereignisse stammen von der Quelle „AD FS Auditing“ und landen im normalen Windows-Sicherheitsprotokoll des jeweiligen Knotens.

Ereignis 1203 ist der wichtigste Einzelwert für die Angriffserkennung: Viele 1203 gegen viele Konten von wenigen Absendern sehen nach Passwort-Spray aus.

Die Activity ID steckt im Ereignistext. Damit lässt sich ein Audit-Eintrag direkt der Fehlermeldung im Admin-Log zuordnen.

Das Sicherheitsprotokoll läuft im Kreis. Ohne Weiterleitung oder ausreichende Größe überschreiben ein paar geschäftige Tage deine Beweise.

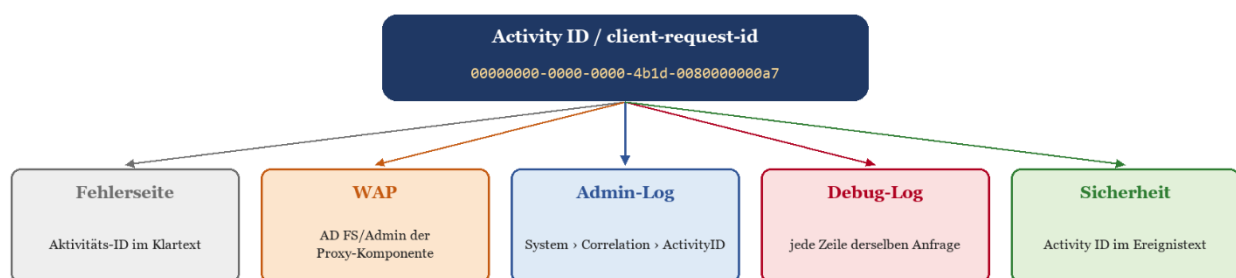
Wie du aus den 1203-Ereignissen ein brauchbares Frühwarnsystem baust, steht in [Passwort-Spray und Brute Force auf ADFS erkennen](#). Die Gegenmaßnahme dazu beschreibt [Extranet Smart Lockout – ADFS gegen Passwort-Spray schützen](#), und wie Auditing, Health Checks und Connect Health im Betrieb zusammenspielen, zeigt [ADFS überwachen – Health Checks, Diagnose-Toolbox und Entra Connect Health](#).

Korrelation: Activity ID, client-request-id und Correlation ID

Jetzt kommt der Teil, der aus drei Protokollen auf fünf Servern eine zusammenhängende Geschichte macht. ADFS vergibt für jede Anfrage eine eindeutige GUID, die Activity ID. Sie bleibt für die gesamte Dauer der Anfrage gleich und wird mit jedem Ereignis protokolliert – im Admin-Log, im Debug-Log, und zwar laut Microsoft auch über Maschinengrenzen hinweg, etwa auf dem Proxy. Scheitert die Anfrage, zeigt die Fehlerseite im Browser genau diese ID an. Das ist der rote Faden, und er ist bereits gesponnen. Du musst ihn nur aufnehmen.

Skizze 5: Eine GUID, viele Protokolle – Korrelation über die Activity ID

Dieselbe Kennung taucht im Browser, auf dem WAP und auf dem ADFS-Knoten auf



Filter, der in der Ereignisanzeige funktioniert (XPath, Admin- und Debug-Log):

```
*[System[Correlation[@ActivityID='{00000000-0000-0000-4b1d-0080000000a7}']]
```

Im Sicherheitsprotokoll steckt die ID im Ereignistext – dort hilft eine Textsuche über die Meldung.

Skizze 5: Die Activity ID verbindet Fehlerseite, Proxy, Admin-Log, Debug-Log und Sicherheitsprotokoll

client-request-id: wie die ID von Knoten zu Knoten wandert

Eine Browser-Anmeldung besteht aus mehreren Umleitungen, und hinter einem Load Balancer landen die einzelnen Schritte nicht zwangsläufig auf demselben Knoten. Damit die Kennung trotzdem erhalten bleibt, hängt ADFS bei Umleitungen an sich selbst die ID als Parameter client-

request-id an die Adresse. Gesteuert wird das über Set-AdfsProperties - SendClientRequestIdAsQueryStringParameter, Standardwert laut Dokumentation \$true. Praktischer Nebeneffekt: Wenn eine Anwenderin oder ein Anwender die Fehlerseite schon weggeklickt hat, steht die ID manchmal noch im Browserverlauf oder in den Entwicklertools in der Adresszeile.

Eine Activity ID auf allen Knoten suchen

In der Ereignisanzeige filterst du über „Aktuelles Protokoll filtern“ > XML > „Abfrage manuell bearbeiten“ mit einem XPath-Ausdruck auf System > Correlation > ActivityID. Bequemer und für Farmen alternativlos ist PowerShell. Get-AdfsFarmInformation liefert die Knoten der Farm, Invoke-Command fragt sie der Reihe nach ab:

```
$id = "00000000-0000-0000-4b1d-0080000000a7"
$xp = "[System[Correlation[@ActivityID='{ $id }']]]"
$nodes = (Get-AdfsFarmInformation).FarmNodes.FQDN

Invoke-Command -ComputerName $nodes -ArgumentList $xp -ScriptBlock {
    param($xp)
    Get-WinEvent -LogName "AD FS/Admin" -FilterXPath $xp -ErrorAction SilentlyContinue
} | Select-Object PSComputerName, TimeCreated, Id, Message | Format-List
```

Die Activity ID im Admin-Log aller Farmknoten suchen. Der Knoten mit Treffer hat die Anfrage bearbeitet.

Im Sicherheitsprotokoll funktioniert der XPath-Filter auf Correlation nicht zuverlässig, weil ADFS die ID dort im Ereignistext ablegt. Hier filterst du zuerst grob nach Quelle und Zeitraum und suchst dann im Text:

```
$id = "00000000-0000-0000-4b1d-0080000000a7"
Get-WinEvent -FilterHashtable @{
    LogName = "Security"
    ProviderName = "AD FS Auditing"
    StartTime = (Get-Date).AddHours(-2)
} | Where-Object Message -like "$id*" |
    Select-Object TimeCreated, Id, Message | Format-List
```

Überwachungseignisse zu einer Activity ID finden

Und die Correlation ID in Entra ID?

Bei föderierten Anmeldungen an Microsoft 365 gibt es eine zweite Welt mit eigenen Kennungen. Entra ID protokolliert in seinen Anmeldeprotokollen eine Correlation ID, die mit der ADFS-Activity-ID nichts zu tun hat. Verbinden lassen sich beide Seiten über Benutzer, Anwendung und Zeitpunkt. Den Entra-Teil holst du heute mit Microsoft Graph PowerShell. In älteren Anleitungen stand an dieser Stelle noch das MSONline-Modul, das ist ausgemustert und kein aktueller Weg mehr.

```
Connect-MgGraph -Scopes "AuditLog.Read.All"
Get-MgAuditLogSignIn -Filter "userPrincipalName eq 'anna.beispiel@contoso.de'" -Top 20 |
    Select-Object CreatedDateTime, AppDisplayName, CorrelationId,
    @{ n = "Fehlercode"; e = { $_.Status.ErrorCode } }
```

Entra-Anmeldeprotokoll zum Abgleich mit den ADFS-Ereignissen

Meldet Entra einen Fehler, obwohl ADFS laut Sicherheitsprotokoll ein Token ausgestellt hat (Ereignis 1200 für die Microsoft-365-Relying-Party), liegt die Ursache hinter ADFS – typischerweise beim Token-Signing-Zertifikat oder beim Federation Trust. Mehr dazu in [Microsoft-365-Federation-Trust nach Zertifikatswechsel aktualisieren](#). Den Weg aus der Föderation heraus beschreibt die Seite [Microsoft Entra ID](#).

Praxisfall: Die Anmeldung, die nur von außen scheiterte

Ein mittelständisches Unternehmen, Farm mit drei Knoten und zwei WAP, Federation Service sts.contoso.de. Seit dem Morgen kommen Außendienstler nicht mehr in eine SaaS-Anwendung, intern läuft alles. Die Fehlerseite zeigte eine Aktivitäts-ID, die immerhin ein Kollege abfotografiert hatte. Die Suche über alle drei Knoten lieferte – nichts. Kein Treffer im Admin-Log der Farm. Auf den WAP dagegen stand die ID im AD FS-Protokoll der Proxy-Komponente, mit dem Hinweis, dass der Proxy die Anfrage nicht an die Farm weiterreichen konnte. Ursache: Bei einer nächtlichen Firewall-Änderung war die Regel von der DMZ zum internen Load Balancer mit einer verwandten Regel zusammengelegt worden, die nur einen der beiden WAP umfasste. Der Load Balancer vor den WAP verteilte munter auf beide, also scheiterte ungefähr jede zweite externe Anmeldung.

Das Debug-Log wurde in diesem Fall nicht einmal gebraucht. Die Activity ID und die simple Frage „Wo taucht sie auf und wo nicht?“ reichten. Das ist die eigentliche Lektion: Nicht das ausführlichste Protokoll findet den Fehler, sondern die richtige Reihenfolge.

Deine Frage	Wo du nachsiehst	Suchkriterium
Warum zeigt ADFS eine Fehlerseite?	AD FS/Admin auf allen Knoten	Activity ID, Exception im Ereignistext
Kommt die Anfrage überhaupt bei der Farm an?	Logs auf dem WAP, danach Admin-Log der Knoten	Activity ID
Welche Regel hat den Zugriff verweigert?	Debug-Log des betroffenen Knotens	Activity ID, Reproduktion
Wer hat sich wann mit welchem Ergebnis angemeldet?	Sicherheitsprotokoll, Quelle AD FS Auditing	Ereignisse 1200 bis 1203, Benutzer
Läuft ein Passwort-Spray?	Sicherheitsprotokoll oder SIEM	Häufung von 1203 über viele Konten
Warum lehnt Microsoft 365 trotz Token ab?	Entra-Anmeldeprotokolle per Graph	Benutzer, Zeit, Correlation ID

Tabelle 5: Welche Frage du mit welchem Protokoll beantwortest

WEITER – Wenn du die Ursache gefunden hast

- › [ADFS-Anmeldung schlägt fehl – die häufigsten Ursachen und ihre Lösung](#) – die acht üblichen Verdächtigen und ihre Lösung
- › [Access Control Policies in ADFS – Zugriffsregeln ohne Claim-Rule-Akrobatik](#) – wenn das Debug-Log eine verweigernde Richtlinie zeigt
- › [ADFS Claim Rules – der Leitfaden](#) – wenn die Exception auf eine Regel zeigt

Fazit: Erst zuhören, dann reparieren

ADFS verschweigt selten etwas. Es verteilt seine Auskünfte nur auf mehrere Protokolle und mehrere Server, und zwei der drei wichtigsten Auskunftsstellen musst du selbst aufdrehen. Das Admin-Log liest du immer zuerst, auf allen Knoten. Das Debug-Log schaltest du gezielt auf einem Knoten ein, reproduzierst den Fehler einmal und schaltest es im selben Atemzug wieder ab. Die Sicherheitsüberwachung richtest du einmal sauber ein – Benutzerrecht, „Anwendung generiert“, Erfolgs- und Fehlerüberwachung, AuditLevel Basic – und prüfst nach jeder GPO-Änderung und jedem Kontowechsel, ob sie noch schreibt. Und die Activity ID ist der Faden, an dem du dich durch alles hindurch hängst.

Das klingt nach Handwerk, und genau das ist es. Eine Farm, die seit zehn Jahren zuverlässig läuft, verdient einen Betrieb, der ihr zuhört – ob sie noch lange bleibt oder der Umstieg auf Entra ID

schon geplant ist. Gerade für die Migration sind saubere Audit-Daten übrigens Gold wert: Sie zeigen, welche Relying Parties überhaupt noch genutzt werden. Wenn du dabei Unterstützung willst, findest du uns im [Consulting zu ADFS \(Active Directory Federation Services\)](#). Wer Fehlersuche lieber im Labor übt als am Montagmorgen in der Produktion, ist in den [ADFS-Schulungen](#) richtig. Protokolle, Auditing und Fehlersuche vertieft außerdem das Buch – mehr dazu unter [ADFS in der Praxis – das Buch im Detail](#).

WEITER — Passende Beiträge zum Betrieb der Farm

- › [ADFS überwachen – Health Checks, Diagnose-Toolbox und Entra Connect Health](#) – Health Checks, Diagnose-Toolbox und Connect Health
- › [Passwort-Spray und Brute Force auf ADFS erkennen](#) – aus 1203-Ereignissen ein Frühwarnsystem bauen
- › [WAP-Proxy-Vertrauen erneuern – wenn der Web Application Proxy plötzlich nicht mehr will](#) – wenn die Proxy-Logs vom Vertrauen erzählen
- › [ADFS-Sicherheit: Angriffe, Härtung, Golden SAML und MFA](#) – Angriffe, Härtung und warum Auditing zur Pflicht gehört

FAQ: ADFS-Ereignisprotokolle und Debug-Tracing

Wie kann ich das ADFS Debug Log aktivieren?

In der Ereignisanzeige unter „Ansicht“ die Analyse- und Debugprotokolle einblenden, dann unter „AD FS Tracing“ mit der rechten Maustaste auf „Debug“ klicken und „Protokoll aktivieren“ wählen. Per Befehl geht es mit `wevtutil sl "AD FS Tracing/Debug" /e:true /q:true`. Schalte es nach der Reproduktion des Fehlers wieder ab.

Wie schalte ich das Debug-Tracing wieder aus?

Mit „Protokoll deaktivieren“ in der Ereignisanzeige oder mit `wevtutil sl "AD FS Tracing/Debug" /e:false /q:true`. Exportiere den Inhalt danach mit `wevtutil epl` in eine EVTX-Datei, bevor du das Protokoll erneut aktivierst.

Warum sehe ich den Ordner „AD FS Tracing“ nicht?

Weil Analyse- und Debugprotokolle in der Ereignisanzeige standardmäßig ausgeblendet sind. Aktiviere im Menü „Ansicht“ die Option „Analyse- und Debugprotokolle einblenden“.

Kostet das Debug-Log Leistung?

Ja. Microsoft schaltet es genau deshalb standardmäßig ab: Es erzeugt in kurzer Zeit sehr viele Einträge. Auf einem einzelnen Knoten für wenige Minuten ist das unkritisch, dauerhaft auf der ganzen Farm nicht.

Warum bleibt das Sicherheitsprotokoll leer, obwohl ich die Überwachung in ADFS eingeschaltet habe?

Weil drei Schalter nötig sind: das Benutzerrecht „Generieren von Sicherheitsüberwachungen“ für das ADFS-Dienstkonto, die Überwachung „Anwendung generiert“ für Erfolg und Fehler und die Erfolgs- und Fehlerüberwachung in den ADFS-Eigenschaften. Häufig überschreibt eine Gruppenrichtlinie einen der ersten beiden Schalter.

Welchen AuditLevel sollte ich verwenden?

Basic, den Standard seit Windows Server 2016. Er protokolliert höchstens fünf Ereignisse je Anfrage. Verbose nur befristet für eine Analyse, None praktisch nie.

Was ist die Activity ID und wo finde ich sie?

Eine GUID, die ADFS jeder Anfrage zuordnet und mit jedem Ereignis protokolliert – im Admin-Log, im Debug-Log und auf dem Proxy. Scheitert eine Anmeldung, zeigt die ADFS-Fehlerseite diese ID an. In der Ereignisanzeige steht sie in den Details unter System > Correlation > ActivityID.

Wie finde ich heraus, welcher Farmknoten eine Anfrage bearbeitet hat?

Suche die Activity ID im Admin-Log aller Knoten, etwa per Invoke-Command mit Get-WinEvent und einem XPath-Filter auf die ActivityID. Die Knotenliste liefert Get-AdfsFarmInformation.

Ist die Correlation ID in Entra ID dieselbe wie die ADFS-Activity-ID?

Nein. Entra ID vergibt eigene Kennungen. Beide Seiten verbindest du über Benutzer, Anwendung und Zeitpunkt; die Entra-Anmeldeprotokolle liest du mit Get-MgAuditLogSignIn aus Microsoft Graph PowerShell.