

Relying Party Trust anlegen – per Metadaten und von Hand



WISSEN

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

> [Active Directory Federation Services](#)

BERATUNG

Neue Anwendungen anbinden, Altlasten aufräumen, Trusts dokumentieren – mit jemandem, der das schon oft gemacht hat.

> [Consulting zu ADFS](#)

SCHULUNG

Trusts im Labor anlegen, absichtlich falsch konfigurieren und wieder geradeziehen.

> [ADFS-Schulungen](#)

Irgendwann kommt die Mail. Ein Fachbereich hat eine neue Anwendung gekauft, der Anbieter schreibt „unterstützt SAML 2.0“, und im Anhang liegt eine XML-Datei oder ein Link auf Metadaten. Dazu der Satz: „Bitte bis Freitag im ADFS einrichten.“ Du sollst also einen ADFS Relying Party Trust anlegen – die Vertrauensstellung, mit der deine Farm einer Anwendung Tokens ausstellt. Das dauert im besten Fall zehn Minuten. Im schlechtesten Fall zehn Minuten plus drei Tage, in denen du herausfindest, dass im Bezeichner ein Schrägstrich zu viel steht.

Dieser Beitrag zeigt dir alle drei Wege: den Import über eine Metadaten-URL, den Import aus einer Metadaten-Datei und die vollständig manuelle Anlage mit Bezeichner, Endpunkten, Signaturalgorithmus SHA-256 und Verschlüsselungszertifikat. Du bekommst jeweils den Weg in der ADFS-Verwaltung und das passende PowerShell-Kommando, eine Tabelle mit den Feldern und ihren typischen Fehlern – und einen eigenen Kasten zur Überwachung der Metadaten. Die Grundlagen zu ADFS, Farm und Protokollen setzen wir voraus; die findest du auf der Übersichtsseite [Active Directory Federation Services](#).

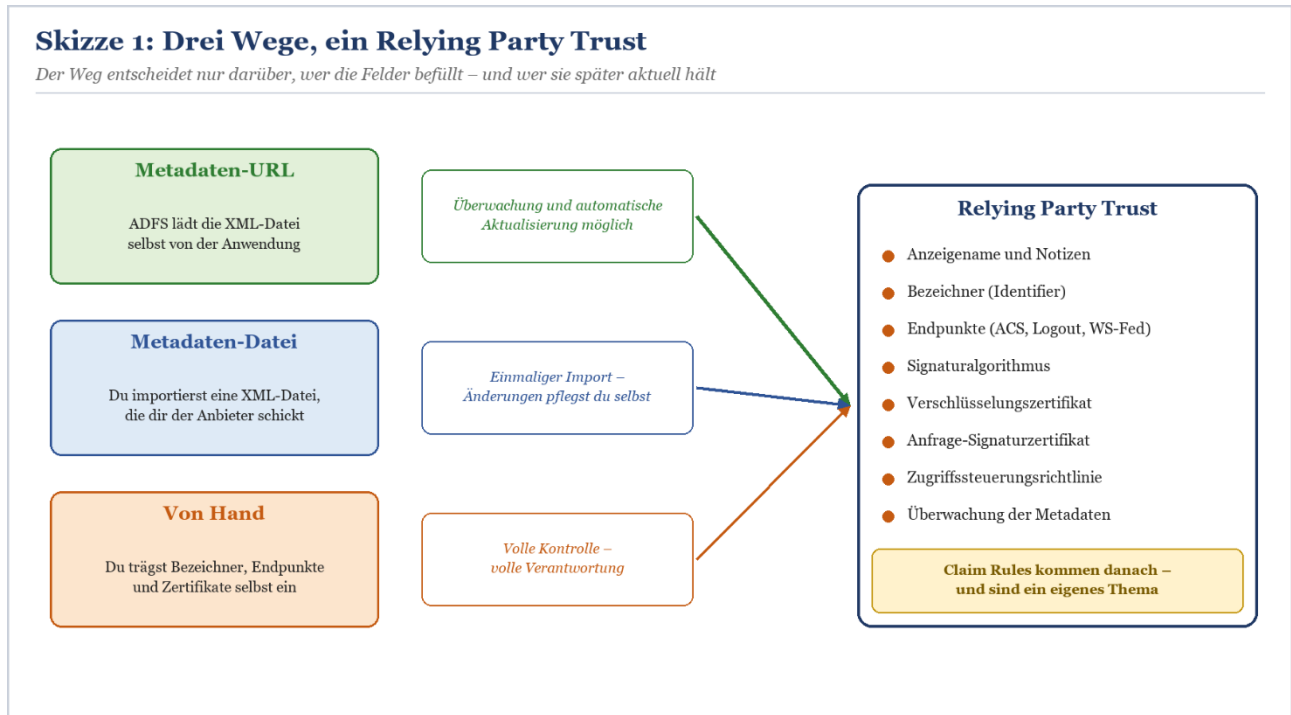
WICHTIG – Was hier nicht drinsteht

Die Claim Rules – also welche Attribute im Token landen und wie sie umgeformt werden – behandeln wir hier bewusst nicht. Der Trust ist das Gefäß, die Regeln sind der Inhalt. Für den Inhalt gibt es den [ADFS Claim Rules Leitfaden](#). Dort steht alles von der LDAP-Regel bis zur NameID-Transformation.

Bevor du klickst: Was ein Relying Party Trust wirklich speichert

Ein Relying Party Trust ist in ADFS nichts Mystisches. Es ist ein Datensatz in der Konfigurationsdatenbank, der beschreibt, wem die Farm Tokens ausstellt, wohin sie diese schickt,

wie sie sie signiert und ob sie sie verschlüsselt. Dazu kommen die Zugriffssteuerungsrichtlinie, die Claim Rules und – optional – eine Metadaten-URL, unter der ADFS regelmäßig nachsieht, ob sich auf der Gegenseite etwas geändert hat. Egal, welchen der drei Wege du gehst: Am Ende steht immer dasselbe Objekt. Der Weg entscheidet nur, wer die Felder befüllt und wer sie später aktuell hält.



Skizze 1: Metadaten-URL, Metadaten-Datei oder manuelle Eingabe – alle drei Wege führen zum selben Trust-Objekt.

Claims aware oder nicht – die erste Frage im Assistenten

Der Assistent „Add Relying Party Trust“ (in der deutschen Konsole die Vertrauensstellung der vertrauenden Seite) fragt dich zuerst, ob die Anwendung „Claims aware“ ist. Für jede SAML- oder WS-Federation-Anwendung lautet die Antwort Ja. „Non claims aware“ brauchst du nur für Anwendungen, die du über den Web Application Proxy mit Vorauthentifizierung veröffentlichst und die selbst mit Kerberos oder gar nicht authentifizieren – das Thema vertieft [Web Application Proxy: Architektur und Ablöse](#). Moderne Anwendungen mit OAuth 2.0 oder OpenID Connect legst du dagegen nicht als klassischen Trust an, sondern über Application Groups; wie das geht, steht in [OAuth 2.0 und OpenID Connect mit ADFS Application Groups](#).

Die Checkliste für das Gespräch mit dem Anbieter

Bevor du auch nur die Konsole öffnest, brauchst du vom Anbieter oder vom Entwicklungsteam ein paar Angaben. Wer die nicht hat, rät – und Raten ist bei Föderation ein teures Hobby.

- Den Entity-Identifier der Anwendung – exakt so, wie er in der SAML-Anfrage als Issuer steht.
- Die Assertion Consumer Service URL (ACS), an die ADFS die Antwort schicken soll, samt Binding (fast immer POST).
- Optional die Logout-URL, falls die Anwendung Single Logout unterstützt.
- Ob die Anwendung ihre Anfragen signiert – und wenn ja, mit welchem Zertifikat.
- Ob sie verschlüsselte Assertions erwartet – und wenn ja, das öffentliche Verschlüsselungszertifikat.
- Welche Claims sie braucht und in welchem NameID-Format. Das ist Futter für die Claim Rules, nicht für den Trust.

- Ob es eine Metadaten-URL gibt, die dauerhaft erreichbar ist.

Weg	Wann sinnvoll	Vorteil	Haken
Metadaten-URL	SaaS-Anwendungen und Produkte, die Metadaten veröffentlichen	Schnell, fehlerarm, Überwachung und automatische Aktualisierung möglich	ADFS muss die URL erreichen – dauerhaft, nicht nur beim Import
Metadaten-Datei	URL nicht erreichbar, Anbieter verschickt XML, Testumgebungen	Genauso fehlerarm beim Import, keine ausgehende Verbindung nötig	Einmaliger Stand – neue Zertifikate des Anbieters pflegst du selbst nach
Von Hand	Eigenentwicklungen, ältere Produkte ohne Metadaten, Sonderfälle	Du weißt genau, was drinsteht	Jeder Tippfehler gehört dir, jede Zertifikatserneuerung auch

Tabelle 1: Die drei Wege im Vergleich.

Weg 1 und 2: Relying Party Trust per Metadaten anlegen

Metadaten sind eine XML-Datei, in der die Anwendung sich selbst beschreibt: ihren Bezeichner, ihre Endpunkte, ihre Zertifikate und manchmal die Claims, die sie erwartet. ADFS liest diese Datei und füllt daraus den Trust. Das ist der bequemste und in der Regel auch der fehlerärmste Weg – vorausgesetzt, die Metadaten stimmen. Und genau das prüfst du nach dem Import trotzdem.

Über die Metadaten-URL in der ADFS-Verwaltung

- ADFS-Verwaltung öffnen, unter „Relying Party Trusts“ die Aktion „Add Relying Party Trust“ wählen.
- „Claims aware“ auswählen und starten.
- Unter „Select Data Source“ die Option „Import data about the relying party published online or on a local network“ wählen und die Metadaten-URL der Anwendung eintragen, etwa <https://reisekosten.contoso.de/saml/metadata>.
- Anzeigenamen vergeben und unter Notizen festhalten, wem die Anwendung gehört und wo das Ticket liegt.
- Zugriffssteuerungsrichtlinie wählen – zum Start meist „Permit everyone“ oder eine Richtlinie mit Gruppe.
- Auf der Seite „Ready to Add Trust“ die Reiter durchsehen: Bezeichner, Endpunkte, Verschlüsselung, Signatur.
- Abschließen. Die Konsole bietet dir danach an, direkt die Claim-Issuance-Policy zu bearbeiten.

Die PowerShell-Variante ist kürzer und hat einen entscheidenden Vorteil: Du kannst sie ins Änderungsticket kopieren, und dein Nachfolger weiß in zwei Jahren noch, was du getan hast.

```
# Listing 1: Relying Party Trust über die Metadaten-URL anlegen
Add-AdfsRelyingPartyTrust -Name "Reisekosten SaaS" `
  -MetadataUrl "https://reisekosten.contoso.de/saml/metadata" `
  -MonitoringEnabled $true `
  -AutoUpdateEnabled $false `
  -AccessControlPolicyName "Permit everyone" `
  -Notes "Fachbereich Finanzen, Änderung CHG-0815, angelegt 30.09.2026"
```

Listing 1: Import über die Metadaten-URL mit Überwachung, aber zunächst ohne automatische Aktualisierung.

TIPP – Den Namen der Zugriffsrichtlinie nicht raten

Die mitgelieferten Richtlinien tragen feste Namen wie „Permit everyone“. Ob deine Farm weitere, eigene Richtlinien hat und wie sie exakt heißen, zeigt dir `Get-AdfsAccessControlPolicy | Select-Object Name`. Ein Tippfehler im Namen lässt das Kommando scheitern – das ist immerhin ein ehrlicher Fehler.

Warum Access Control Policies der bessere Weg sind als handgeschriebene Autorisierungsregeln, erklärt [Access Control Policies in ADFS – Zugriffsregeln ohne Claim-Rule-Akrobatik](#).

Aus einer Metadaten-Datei

Kommt die Metadaten-URL nicht durch – weil sie nur mit Anmeldung erreichbar ist, weil deine ADFS-Server keinen Weg ins Internet haben oder weil der Anbieter die Datei per Mail verschickt –, importierst du sie aus einer Datei. Im Assistenten heißt die Option „Import data about the relying party from a file“, in PowerShell nimmst du -MetadataFile. Der Import ist genauso vollständig wie bei der URL. Der Unterschied kommt später: ADFS weiß nicht, woher die Datei stammt, und kann deshalb auch nicht nachsehen, ob es eine neue Version gibt.

```
# Listing 2: Relying Party Trust aus einer Metadaten-Datei anlegen
Add-AdfsRelyingPartyTrust -Name "Reisekosten SaaS" `
  -MetadataFile "C:\Temp\reisekosten-metadata.xml" `
  -AccessControlPolicyName "Permit everyone" `
  -Notes "Metadaten-Datei vom Anbieter, Stand 30.09.2026, Zertifikat gültig bis 2028"
```

Listing 2: Import aus einer Datei – der Stand der Datei gehört in die Notizen.

WARNUNG – Die Datei ist ein Schnappschuss

Eine importierte Datei veraltet in dem Moment, in dem der Anbieter sein Zertifikat tauscht. Signiert die Anwendung ihre Anfragen mit einem neuen Zertifikat oder erwartet sie die Verschlüsselung mit einem neuen Schlüssel, bricht die Anmeldung – und zwar zuverlässig an einem Montagmorgen. Trag das Ablaufdatum der Zertifikate aus der Datei in dein Wiedervorlagesystem ein. Nicht in dein Gedächtnis.

Wenn der Import scheitert

Die Fehlermeldung beim Metadaten-Import ist erfrischend unkonkret: ADFS konnte die Metadaten nicht lesen, bitte prüfe die Adresse. Dahinter steckt fast immer einer dieser Klassiker:

Ursache	Woran du sie erkennst	Lösung
Kein Weg nach draußen	Aufruf der URL im Browser auf dem ADFS-Server läuft ins Leere	Proxy- oder Firewallfreigabe für die Metadaten-URL, sonst Datei-Import
TLS-Aushandlung scheitert	Im Browser geht es, aus PowerShell mit Invoke-WebRequest nicht	.NET auf dem ADFS-Server für starke Kryptografie und TLS 1.2 konfigurieren
Mehrere Entitäten in einer Datei	Die Datei beginnt mit EntitiesDescriptor statt EntityDescriptor	ADFS erwartet genau eine Entität – den passenden Block herauslösen oder vom Anbieter einzeln anfordern
Datei nachträglich bearbeitet	Import meldet einen Signaturfehler, obwohl die Datei gut aussieht	Signierte Metadaten nicht von Hand ändern – Original neu anfordern
Bezeichner schon vergeben	Fehlermeldung, dass der Identifier bereits verwendet wird	Alten Test-Trust suchen: Get-AdfsRelyingPartyTrust -Identifier <Bezeichner>
Anmelde-Seite statt XML	Die URL liefert HTML, nicht XML	Richtige Metadaten-URL beim Anbieter erfragen – oft steht sie ganz hinten in der Admin-Doku

Tabelle 2: Warum der Metadaten-Import scheitert – und was dagegen hilft.

Weg 3: Relying Party Trust von Hand anlegen

Die manuelle Anlage ist der Weg für Eigenentwicklungen, für ältere Produkte ohne Metadaten und für Anbieter, deren Metadaten so kaputt sind, dass du ihnen lieber nicht glaubst. Du brauchst dafür

die Angaben aus der Checkliste oben. Im Assistenten wählst du „Enter data about the relying party manually“ und gehst die Seiten der Reihe nach durch.

Bezeichner: ein Zeichen zu viel, eine Anwendung zu wenig

Der Bezeichner (Identifier) ist das wichtigste Feld überhaupt. Wenn eine SAML-Anfrage eintrifft, sucht ADFS anhand des Issuers in der Anfrage den passenden Trust. Findet es keinen, ist Schluss – die Anwendung ist für deine Farm dann schlicht unbekannt, und im Admin-Log landet ein Fehler, dass die angefragte Relying Party nicht angegeben oder nicht unterstützt ist. Der Vergleich ist wörtlich: <https://reisekosten.contoso.de> und <https://reisekosten.contoso.de/> sind für ADFS zwei verschiedene Dinge. Ein Bezeichner darf außerdem nur in einem einzigen Trust vorkommen. Du kannst mehrere Bezeichner pro Trust hinterlegen – praktisch, wenn eine Anwendung in Test und Produktion unterschiedliche Issuer schickt, aber dieselben Regeln braucht.

Bei WS-Federation-Anwendungen ist der Bezeichner der Wert, den die Anwendung im Parameter wtrealm schickt, häufig eine URN wie urn:contoso:intranet oder schlicht die Basis-URL. Ein gutes Beispiel für eine solche Anwendung ist SharePoint Server, das wir in [SharePoint Server mit ADFS – SAML-Authentifizierung einrichten](#) Schritt für Schritt anbinden.

Endpunkte: wohin die Antwort gehen darf

Auf der Seite „Configure URL“ aktivierst du entweder die Unterstützung für WS-Federation Passive und trägst die Rücksprung-URL ein, oder die Unterstützung für SAML 2.0 WebSSO mit der ACS-URL. Weitere Endpunkte – ein zweiter ACS mit anderem Index, ein Logout-Endpunkt, alternative Bindings – ergänzt du nach dem Assistenten im Reiter „Endpoints“. ADFS schickt eine SAML-Antwort nur an eine hier hinterlegte Adresse. Fordert die Anwendung eine andere an, verweigert ADFS die Auslieferung. Das ist kein Bug, das ist der Sinn der Sache: Sonst könnte jeder mit passender Anfrage Tokens an sich selbst umleiten.

Signaturalgorithmus: SHA-256, ohne Diskussion

ADFS signiert jede SAML-Assertion. Mit welchem Hash-Verfahren, stellst du nach der Anlage in den Eigenschaften des Trusts im Reiter „Advanced“ unter „Secure hash algorithm“ ein. Neue Trusts legt ADFS mit SHA-256 an – und dabei soll es bleiben. SHA-1 ist nur noch der Kompatibilitätsmodus für Anwendungen, deren Hersteller seit Jahren nicht mehr ans Telefon geht. Wenn ein Anbieter heute noch SHA-1 verlangt, ist das kein Konfigurationswunsch, sondern ein Befund für dein Risikoregister.

WARNUNG — Die Tücke im Parameterwert

In PowerShell gibst du den Algorithmus nicht als „SHA256“ an, sondern als vollständige URI aus dem XML-Signaturstandard, die auf xmldsig-more#rsa-sha256 endet. Die URI beginnt mit http, nicht mit https – sie ist ein Bezeichner, keine Webadresse, und wer aus Gewohnheit ein s ergänzt, erzeugt einen ungültigen Wert. Tippen musst du sie nicht: .NET bringt sie als Konstante mit, [System.Security.Cryptography.Xml.SignedXml]::XmlDsigRSASHA256Url, wie in Listing 3. Für SHA-1 gäbe es entsprechend XmlDsigRSASHA1Url – aber das willst du ja nicht.

Verschlüsselungszertifikat: optional, aber nicht beliebig

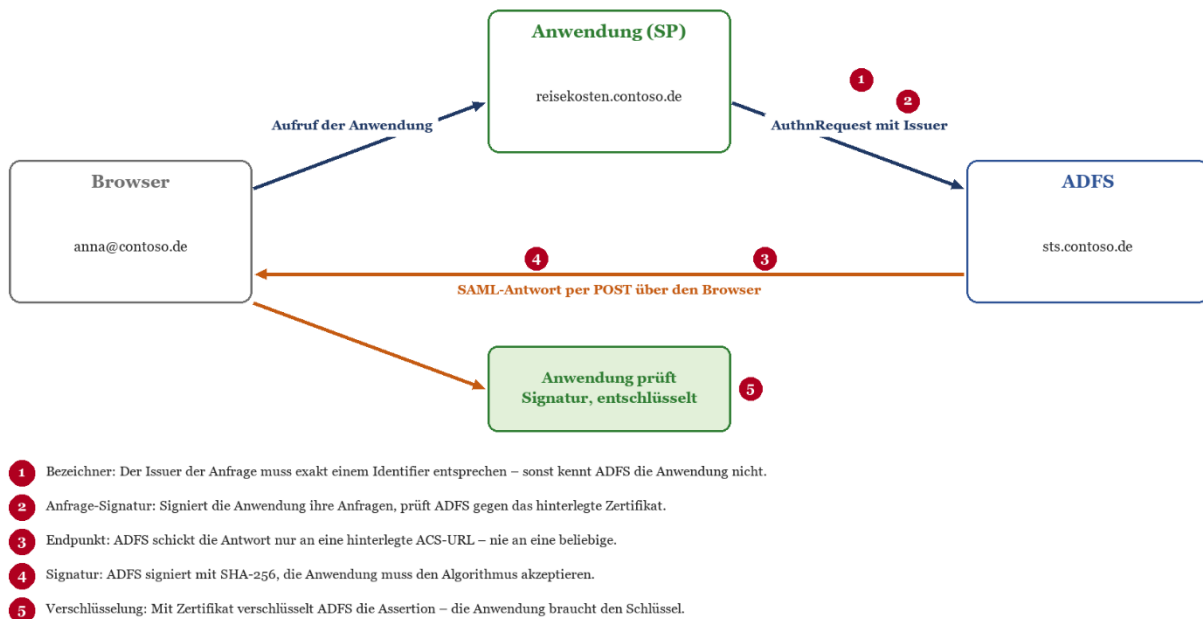
Auf der Seite „Configure Certificate“ kannst du ein Verschlüsselungszertifikat der Anwendung hinterlegen. Tust du das, verschlüsselt ADFS die Assertion mit dem öffentlichen Schlüssel – nur die Anwendung kann sie lesen. Das ist sinnvoll, wenn Claims mit sensiblen Inhalten über den Browser wandern. Es ist aber auch eine zusätzliche Fehlerquelle: Hinterlegst du ein Zertifikat, das

die Anwendung gar nicht zum Entschlüsseln konfiguriert hat, bekommt sie ein Token, mit dem sie nichts anfangen kann. Und weil ADFS die Sperrprüfung für dieses Zertifikat vornimmt, kann eine nicht erreichbare Sperrliste des Anbieters die Anmeldung ebenfalls stoppen. Das Zertifikat gibt es als .cer-Datei mit dem öffentlichen Teil – einen privaten Schlüssel brauchst du dafür nie. Wer dir eine .pfx mit Kennwort schickt, hat die Aufgabe nicht verstanden.

Signiert die Anwendung ihre Anfragen, hinterlegst du ihr Signaturzertifikat nach der Anlage im Reiter „Signature“. Damit prüft ADFS, ob eine Anfrage wirklich von der Anwendung stammt.

Skizze 2: Wo welches Feld im SAML-Ablauf greift

SP-initiierte Anmeldung an einer Anwendung – jede Zahl ist ein Feld, das du richtig setzen musst



Skizze 2: An welcher Stelle im SAML-Ablauf Bezeichner, Endpunkt, Signatur und Verschlüsselung greifen.

Der manuelle Trust in PowerShell

Hier der komplette manuelle Weg für eine SAML-Anwendung mit Signaturprüfung und verschlüsselten Assertions. Die SAML-Endpunkte baust du vorher mit New-AdfsSamlEndpoint als Objekte, die Zertifikate lädst du aus den .cer-Dateien des Anbieters.

```
# Listing 3: Relying Party Trust vollständig von Hand anlegen
$acs = New-AdfsSamlEndpoint -Binding "POST" -Protocol "SAMLAssertionConsumer" `
    -Uri "https://zeiterfassung.contoso.local/saml/acs" -Index 0 -IsDefault $true
$slo = New-AdfsSamlEndpoint -Binding "Redirect" -Protocol "SAMLLogout" `
    -Uri "https://zeiterfassung.contoso.local/saml/logout"

Add-Type -AssemblyName System.Security
$sha256 = [System.Security.Cryptography.Xml.SignedXml]::XmlDsigRSA256Url

$enc = New-Object System.Security.Cryptography.X509Certificates.X509Certificate2 `
    "C:\Temp\zeiterfassung-encryption.cer"
$sig = New-Object System.Security.Cryptography.X509Certificates.X509Certificate2 `
    "C:\Temp\zeiterfassung-signing.cer"

Add-AdfsRelyingPartyTrust -Name "Zeiterfassung" `
    -Identifier "https://zeiterfassung.contoso.local/saml" `
    -ProtocolProfile "SAML" `
    -SamlEndpoint $acs, $slo `
    -SignatureAlgorithm $sha256 `
    -EncryptionCertificate $enc `
    -RequestSigningCertificate $sig `
```

```
-SamlResponseSignature "AssertionOnly" `
-AccessControlPolicyName "Permit everyone" `
-Notes "Eigenentwicklung, Team Personal-IT, Änderung CHG-0816"
```

Listing 3: Manuelle Anlage mit ACS- und Logout-Endpoint, SHA-256, Verschlüsselungs- und Anfrage-Signaturzertifikat.

Für eine WS-Federation-Anwendung ist es noch kürzer: -Identifier mit dem wtrealm-Wert, -WSFedEndpoint mit der Rücksprung-URL, fertig. Bestehende Trusts änderst du mit Set-AdfsRelyingPartyTrust -TargetName – etwa um einen alten Trust von SHA-1 auf SHA-256 umzustellen. Vorher klärst du allerdings, ob die Anwendung das auch kann.

Die Felder und ihre typischen Fehler

Die folgende Tabelle ist die, die du ausdrucken und neben den Bildschirm legen solltest. Entstanden ist sie aus vielen Fehlersuchen, in denen am Ende jemand sagte: „Ach, da.“

Feld	Wo in der Konsole	PowerShell	Typischer Fehler	Symptom
Anzeigename	Specify Display Name	-Name	Nichtssagend („SAML-Test2“)	Nach zwei Jahren weiß niemand, ob man ihn löschen darf
Bezeichner	Configure Identifiers / Reiter Identifiers	-Identifier	Schrägstrich, http statt https, Groß- und Kleinschreibung	ADFS kennt die Anwendung nicht, Fehler im Admin-Log
ACS-Endpoint	Configure URL / Reiter Endpoints	-SamlEndpoint	Test-URL in Produktion, falsches Binding, falscher Index	Antwort geht ins Leere oder wird verweigert
Logout-Endpoint	Reiter Endpoints	-SamlEndpoint (SAMLLogout)	Fehlt, obwohl die Anwendung Single Logout auslöst	Abmeldung endet auf einer ADFS-Fehlerseite
WS-Fed-URL	Configure URL	-WSFedEndpoint	Rücksprung-URL ohne Pfad der Anwendung	Anmeldung klappt, Anwender landet auf der Startseite des Servers
Signaturalgorithmus	Reiter Advanced	-SignatureAlgorithm	SHA-1 aus alter Anleitung übernommen, URI mit https	Anwendung lehnt Signatur ab oder Kommando scheitert
Antwortsignatur	nur PowerShell	-SamlResponseSignature	Anwendung erwartet signierte Nachricht, ADFS signiert nur die Assertion	Anwendung meldet fehlende Signatur
Verschlüsselungszertifikat	Configure Certificate / Reiter Encryption	-EncryptionCertificate	Hinterlegt, obwohl die Anwendung nicht entschlüsselt; abgelaufen	Token unlesbar, Anmeldung scheitert in der Anwendung
Anfrage-Signaturzertifikat	Reiter	-	Altes Zertifikat	ADFS verwirft

	Signature	RequestSigningCertificate	nach Tausch beim Anbieter	die Anfrage schon vor der Anmeldung
Zugriffsrichtlinie	Choose Access Control Policy	- AccessControlPolicyName	Zu Testzwecken „Permit everyone“ – und dabei geblieben	Jeder im Verzeichnis kommt rein, auch wer nicht soll
Metadaten-URL	Reiter Monitoring	-MetadataUrl	Überwachung aktiv, aber URL aus dem Netz der Farm nicht erreichbar	Status bleibt veraltet, niemand merkt es
Notizen	Specify Display Name / Reiter Notes	-Notes	Leer	Die Inventur vor der Migration wird zur Archäologie

Tabelle 3: Die Felder eines Relying Party Trusts mit ihren Lieblingsfehlern.

FAKTEN — Relying Party Trust in Zahlen und Namen

Cmdlets im ADFS-Modul: Add-, Get-, Set-, Update-, Enable-, Disable- und Remove-AdfsRelyingPartyTrust.

Pflichtangaben bei manueller Anlage: Name und mindestens ein Bezeichner. Alles andere ist technisch optional – fachlich brauchst du mindestens einen Endpunkt.

Protokollprofil: SAML, WsFederation oder WsFed-SAML; ohne Angabe gilt WsFed-SAML.

Antwortsignatur: AssertionOnly, MessageOnly oder MessageAndAssertion.

Überwachung: MonitoringEnabled prüft die Metadaten-URL, AutoUpdateEnabled übernimmt Änderungen; das Intervall legst du farmweit mit Set-AdfsProperties -MonitoringInterval in Minuten fest.

Mit aktiver automatischer Aktualisierung sind die Felder, die aus Metadaten überschrieben werden können, in der Konsole schreibgeschützt.

Nach der Anlage: prüfen, testen, überwachen

Der Assistent ist fertig, die Konsole zeigt einen neuen Eintrag, der Fachbereich fragt, ob es schon geht. Es geht noch nicht. Zuerst kommen die Claim Rules – ohne sie stellt ADFS ein Token ohne nützlichen Inhalt aus. Wie du sie baust, steht im [ADFS Claim Rules Leitfadens](#). Danach prüfst du den Trust selbst, und zwar nicht in der Konsole, sondern mit einem Kommando, dessen Ausgabe du ins Ticket kopieren kannst.

```
# Listing 4: Relying Party Trust nach der Anlage prüfen
Get-AdfsRelyingPartyTrust -Name "Zeiterfassung" |
    Format-List Name, Enabled, Identifier, ProtocolProfile, SamlEndpoints, WsFedEndpoint,
        SignatureAlgorithm, SamlResponseSignature, EncryptClaims,
        EncryptionCertificate, RequestSigningCertificate,
        AccessControlPolicyName, MetadataUrl, MonitoringEnabled, AutoUpdateEnabled
```

Listing 4: Die Eigenschaften, die du nach jeder Anlage gegen das Datenblatt des Anbieters hältst.

Testen, ohne die Anwender als Testteam zu missbrauchen

Der ehrlichste Test ist eine SP-initiierte Anmeldung: Du rufst die Anwendung auf, sie leitet zu sts.contoso.de, du meldest dich an und landest wieder in der Anwendung – mit den erwarteten Attributen. Mach das mit einem Testkonto, einmal intern und einmal über den Web Application

Proxy von außen. Die IdP-initiierte Anmeldeseite von ADFS ist in neueren Versionen standardmäßig abgeschaltet; schalte sie nicht nur für einen Test dauerhaft ein. Klappt es nicht, führt der Weg über [ADFS-Anmeldung schlägt fehl – die häufigsten Ursachen und ihre Lösung](#) und bei Bedarf über das Debug-Tracing, das wir in [ADFS-Ereignisprotokolle lesen – Admin-Log, Debug-Tracing und Auditing](#) beschreiben.

TIPP – Ein Praxisfall aus dem Alltag

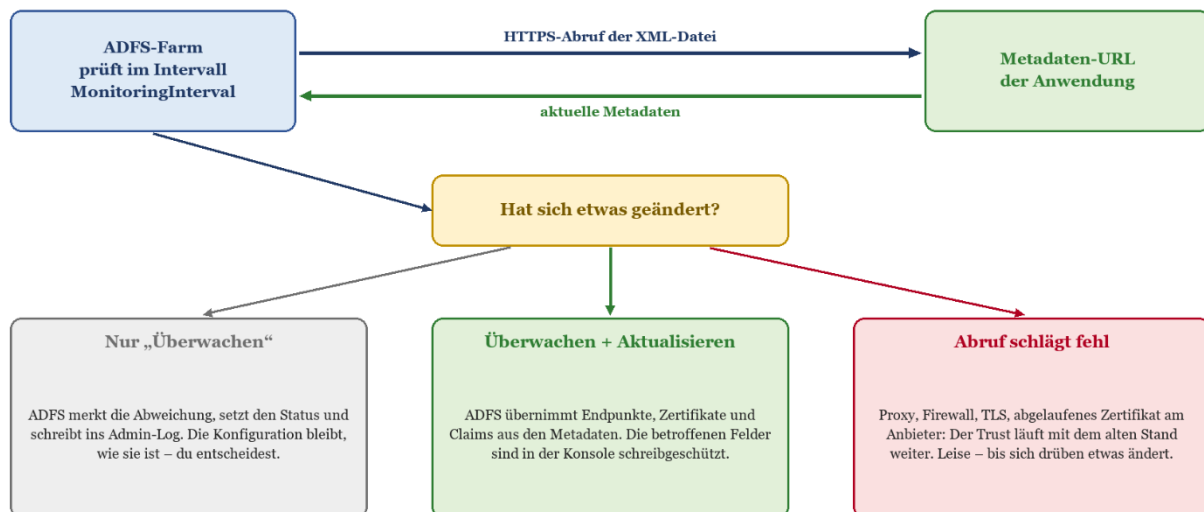
Ein Maschinenbauunternehmen bindet eine neue SaaS-Anwendung per Metadaten-URL an. Import problemlos, Test problemlos, Livegang problemlos. Sechs Wochen später: Keiner kommt mehr rein. Der Anbieter hatte sein Verschlüsselungszertifikat getauscht, die automatische Aktualisierung hatte das neue Zertifikat brav übernommen – nur die Anwendung selbst entschlüsselte in einer Region noch mit dem alten Schlüssel. Die Lösung: automatische Aktualisierung für diesen Trust ab, Überwachung an, jede Änderung mit dem Anbieter abstimmen. Die Automatik war nicht schuld. Das blinde Vertrauen in sie schon.

Metadaten überwachen und automatisch aktualisieren

ADFS kann für jeden Trust mit Metadaten-URL regelmäßig nachsehen, ob sich die Metadaten geändert haben. Dafür gibt es im Reiter „Monitoring“ zwei Häkchen, die gern verwechselt werden: „Monitor relying party“ lässt ADFS die URL abrufen und vergleichen. „Automatically update relying party“ lässt ADFS die Änderungen auch übernehmen. Das erste ist fast immer richtig. Das zweite ist eine Entscheidung, die du pro Anwendung triffst.

Skizze 3: Überwachung und automatische Aktualisierung

Zwei Häkchen, zwei sehr unterschiedliche Folgen – ADFS schaut nach und übernimmt dann auch



Skizze 3: Was bei reiner Überwachung, bei automatischer Aktualisierung und bei einem gescheiterten Abruf passiert.

FAKTEN – Monitoring der Metadaten und automatische Aktualisierung

Überwachung einschalten: Set-AdfsRelyingPartyTrust -TargetName "Reisekosten SaaS" -MonitoringEnabled \$true. Voraussetzung ist eine hinterlegte Metadaten-URL – bei einem Datei-Import gibt es nichts zu überwachen, bis du im Reiter „Monitoring“ eine URL einträgst.

Automatische Übernahme: zusätzlich -AutoUpdateEnabled \$true. ADFS übernimmt dann Endpunkte, Zertifikate und Claim-Beschreibungen aus den Metadaten; die betroffenen Felder werden schreibgeschützt. Deine Claim Rules und die Zugriffsrichtlinie fasst die Aktualisierung nicht

an.

Intervall: farmweit über Set-AdfsProperties -MonitoringInterval in Minuten. Den aktuellen Wert liest du mit (Get-AdfsProperties).MonitoringInterval.

Status je Trust: Get-AdfsRelyingPartyTrust liefert LastMonitoredTime, LastPublishedPolicyCheckSuccessful und LastUpdateTime. Ein alter Zeitstempel oder ein False ist das Signal, das du alarmieren willst.

Sofort nachziehen: Update-AdfsRelyingPartyTrust -TargetName "Reisekosten SaaS" holt die Metadaten auf Zuruf.

Erreichbarkeit: Alle Server der Farm sollten die Metadaten-URL erreichen – über Proxy, Firewall und mit aktuellem TLS.

Faustregel: Überwachung immer an, wenn es eine URL gibt. Automatische Aktualisierung an, wenn der Anbieter Zertifikatswechsel sauber mit Vorlauf veröffentlicht; aus, wenn er gern mal spontan ist.

Die Werte aus dem Faktenkasten kannst du mit einer kleinen Abfrage für die ganze Farm auswerten und als geplante Aufgabe laufen lassen. Zwei Fragen sind dabei interessant: Welche überwachten Trusts wurden lange nicht mehr erfolgreich geprüft – und welche Zertifikate von Anwendungen laufen demnächst ab?

```
# Listing 5: Überwachung und Zertifikatsablauf aller Trusts auswerten
Get-AdfsRelyingPartyTrust | Where-Object MonitoringEnabled |
    Select-Object Name, MetadataUrl, AutoUpdateEnabled,
                    LastMonitoredTime, LastPublishedPolicyCheckSuccessful, LastUpdateTime |
    Sort-Object LastMonitoredTime | Format-Table -AutoSize

$grenze = (Get-Date).AddDays(60)
Get-AdfsRelyingPartyTrust | ForEach-Object {
    $rp = $_
    @($rp.EncryptionCertificate) + @($rp.RequestSigningCertificate) |
        Where-Object { $_ -and $_.NotAfter -lt $grenze } |
        ForEach-Object {
            [pscustomobject]@{ Trust = $rp.Name; Zertifikat = $_.Subject; Ablauf = $_.NotAfter
        }
    }
}
```

Listing 5: Überwachungsstatus und bald ablaufende Anwendungszertifikate – als Bericht für das Betriebsteam.

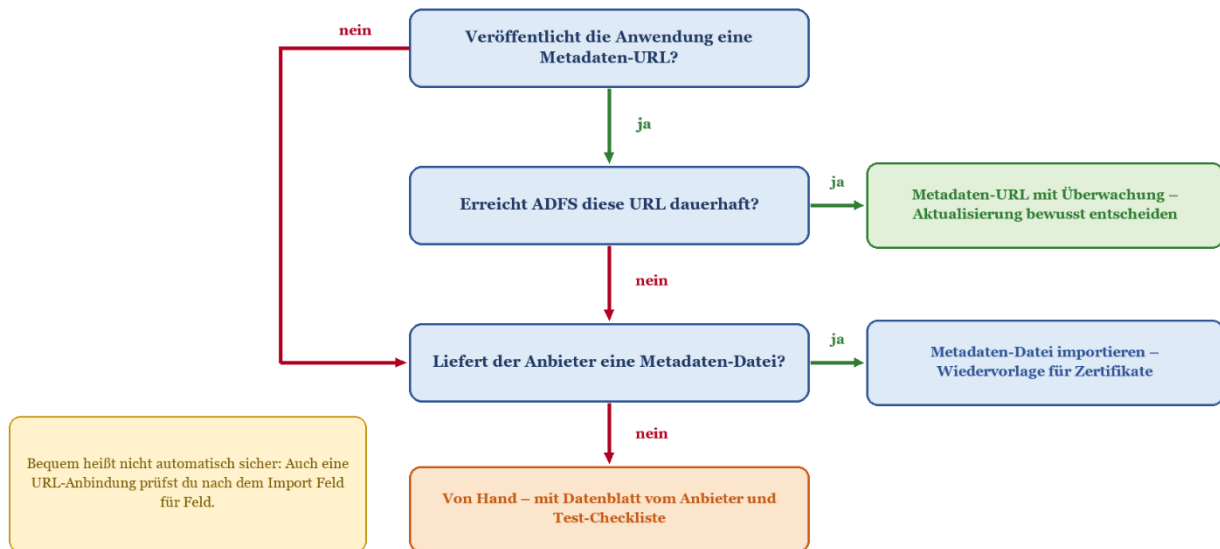
Wie sich solche Prüfungen in die Überwachung der gesamten Farm einfügen, zeigt [ADFS überwachen – Health Checks, Diagnose-Toolbox und Entra Connect Health](#). Und vergiss eines nicht: Die Metadaten-Überwachung schaut nur auf die Gegenseite. Wenn deine eigene Farm ihr Token-Signing-Zertifikat erneuert, müssen die Anwendungen die neuen Metadaten von sts.contoso.de holen – das ist das Spiegelbild dieses Themas und steht in [ADFS Token-Signing-Zertifikat erneuern – mit und ohne AutoCertificateRollover](#).

Aufräumen gehört zum Anlegen

Jeder Trust, den du heute anlegst, ist ein Eintrag in der Inventur von morgen. Ein aussagekräftiger Name, gepflegte Notizen mit Verantwortlichen und Ticketnummer, eine Metadaten-URL, wo es eine gibt – das kostet beim Anlegen zwei Minuten und spart dir bei der Bestandsaufnahme Tage. Wenn du den Ausstieg aus ADFS irgendwann planst, beginnt er genau dort: bei [Relying Parties inventarisieren – die Bestandsaufnahme vor der Entra-Migration](#). Test-Trusts, die niemand mehr braucht, deaktivierst du zuerst mit Disable-AdfsRelyingPartyTrust und entfernst sie erst nach einer Karenzzeit. Wer direkt löscht, erfährt manchmal auf unangenehme Weise, dass „Test“ im Namen nicht „ungenutzt“ bedeutet. Eine aktuelle Sicherung der Konfiguration, wie in [ADFS sichern und wiederherstellen – Rapid Restore Tool in der Praxis](#) beschrieben, ist vor größeren Aufräumaktionen Pflicht.

Skizze 4: Welcher Weg für welche Anwendung?

Von oben nach unten – die erste passende Antwort gewinnt



Skizze 4: Entscheidungshilfe – welcher Weg für welche Anwendung.

Fazit: Zehn Minuten Arbeit, wenn du die richtigen Fragen vorher stellst

Einen ADFS Relying Party Trust anzulegen, ist kein Hexenwerk. Mit Metadaten-URL ist es ein Kommando, mit Metadaten-Datei ebenfalls, und auch der manuelle Weg ist mit Bezeichner, Endpunkten, SHA-256 und optionalem Verschlüsselungszertifikat überschaubar. Schwierig wird es nicht beim Anlegen, sondern davor und danach: davor, wenn du die Angaben des Anbieters nicht hast und rätst; danach, wenn Zertifikate auf der Gegenseite wechseln und niemand hinschaut. Deshalb gehören die Checkliste vor dem Klicken, die Prüfung mit Get-AdfsRelyingPartyTrust und eine eingeschaltete Überwachung zum Standard, nicht zur Kür.

ADFS läuft in vielen Unternehmen seit über zehn Jahren und wird dort auch noch eine Weile laufen. Jeder saubere Trust macht den Betrieb ruhiger – und einen späteren Umzug der Anwendungen nach Entra ID leichter, weil du genau weißt, was dort drinsteht. Wie dieser Umzug aussieht, beschreibt [SAML-Anwendungen von ADFS nach Entra ID migrieren](#); den größeren Rahmen findest du im Themenbereich [Microsoft Entra ID](#). Wenn du eine ganze Reihe von Anwendungen anbinden oder einen gewachsenen Bestand geradeziehen willst, unterstützen wir dich im [Consulting zu ADFS \(Active Directory Federation Services\)](#). Wer das lieber selbst im Labor übt, ist in den [ADFS-Schulungen](#) richtig. Und die Zusammenhänge zwischen Trusts, Zertifikaten und Metadaten vertieft das Buch – mehr dazu unter [ADFS in der Praxis – das Buch im Detail](#).

WEITER – Passend zum Thema

- › [ADFS Claim Rules Leitfaden](#) – was nach dem Trust in die Tokens kommt
- › [SAML-Anwendung an ADFS anbinden – Praxisbeispiel SaaS](#) – der komplette Ablauf an einem SaaS-Beispiel
- › [Access Control Policies in ADFS – Zugriffsregeln ohne Claim-Rule-Akrobatik](#) – wer die Anwendung nutzen darf
- › [ADFS-Anmeldung schlägt fehl – die häufigsten Ursachen und ihre Lösung](#) – wenn der Test nicht grün wird

› [Relying Parties inventarisieren – die Bestandsaufnahme vor der Entra-Migration](#) – der Überblick über alle Trusts der Farm

FAQ: ADFS Relying Party Trust anlegen

Wie lege ich einen Relying Party Trust in ADFS per PowerShell an?

Mit `Add-AdfsRelyingPartyTrust`. Für den Import über Metadaten reichen `-Name` und `-MetadataUrl` oder `-MetadataFile`. Für die manuelle Anlage brauchst du `-Name`, `-Identifier` und mindestens einen Endpunkt über `-SamlEndpoint` oder `-WSFedEndpoint`.

Was ist der Unterschied zwischen Metadaten-URL und Metadaten-Datei?

Der Import selbst ist gleich vollständig. Mit URL kann ADFS die Metadaten später überwachen und auf Wunsch automatisch übernehmen. Eine importierte Datei ist ein Schnappschuss, dessen Änderungen du selbst nachpflegst.

Warum schlägt der Import der Federation-Metadaten fehl?

Meist erreicht der ADFS-Server die URL nicht, die TLS-Aushandlung scheitert, die Datei enthält mehrere Entitäten, sie wurde nach dem Signieren bearbeitet oder der Bezeichner ist bereits in einem anderen Trust vergeben.

Welchen Signaturalgorithmus sollte ich für einen Relying Party Trust verwenden?

SHA-256. In PowerShell übergibst du dafür die URI aus dem XML-Signaturstandard, am einfachsten über die .NET-Konstante `[System.Security.Cryptography.Xml.SignedXml]::XmlDsigRSASHA256Url`. SHA-1 ist nur noch ein Kompatibilitätsmodus für sehr alte Anwendungen.

Brauche ich ein Verschlüsselungszertifikat?

Nein, es ist optional. Wenn du eines hinterlegst, verschlüsselt ADFS die Assertion, und die Anwendung muss sie entschlüsseln können. Hinterlege es nur, wenn die Anwendung das ausdrücklich unterstützt und erwartet.

Was bedeutet es, wenn ADFS die Relying Party nicht findet?

Der Issuer in der Anfrage der Anwendung stimmt mit keinem hinterlegten Bezeichner überein. Vergleiche beide Werte Zeichen für Zeichen – inklusive abschließendem Schrägstrich und Protokoll.

Wie oft prüft ADFS die Metadaten einer Relying Party?

So oft, wie es das farmweite `MonitoringInterval` vorgibt, angegeben in Minuten. Den aktuellen Wert zeigt dir `(Get-AdfsProperties).MonitoringInterval`. Sofort aktualisieren kannst du mit `Update-AdfsRelyingPartyTrust`.

Soll ich die automatische Aktualisierung einschalten?

Die Überwachung ja, wann immer es eine Metadaten-URL gibt. Die automatische Aktualisierung nur, wenn der Anbieter Zertifikatswechsel mit Vorlauf ankündigt und veröffentlicht – sonst übernimmt ADFS Änderungen, auf die die Anwendung selbst noch nicht vorbereitet ist.

Kann ich einen Relying Party Trust nachträglich ändern?

Ja, mit Set-AdfsRelyingPartyTrust -TargetName oder in den Eigenschaften in der ADFS-Verwaltung. Ist die automatische Aktualisierung aktiv, sind die Felder, die aus den Metadaten stammen, schreibgeschützt.

Wo konfiguriere ich die Claims, die die Anwendung bekommt?

In der Claim-Issuance-Policy des Trusts, also in den Claim Rules. Das ist ein eigenes Thema – ausführlich im [ADFS Claim Rules Leitfaden](#).