

Relying Parties inventarisieren – die Bestandsaufnahme vor der Entra-Migration



WISSEN

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

> [Active Directory Federation Services](#)

BERATUNG

Deine Farm hat 80 Relying Parties und keiner weiß, welche davon noch jemand braucht? Wir sortieren mit.

> [Consulting zu ADFS](#)

SCHULUNG

Inventar, Bewertung und Migration einer Beispielfarm im Labor durchspielen – bevor es ernst wird.

> [ADFS-Schulungen](#)

Jede ADFS-Farm, die länger als fünf Jahre läuft, hat eine Relying Party, deren Name eine Drohung ist. „Test_alt_nicht_löschen“. „Neu_2017_FINAL“. „Bitte_Rücksprache_Meier“ – wobei Herr Meier seit 2019 in Rente ist und auf dem Golfplatz nicht ans Telefon geht. Solange die Farm einfach läuft, stört das niemanden. In dem Moment, in dem die Geschäftsführung fragt, wann ihr endlich „alles nach Entra“ umzieht, wird aus der Kuriositätsammlung ein Projektrisiko. Denn bevor du eine einzige Anwendung migrierst, musst du eine unangenehm einfache Frage beantworten: Welche Anwendungen hängen eigentlich dran?

Genau darum geht es hier, und um nichts anderes. Du exportierst alle Relying Parties samt Regeln so, dass du sie im Notfall wiederherstellen kannst. Du misst, welche davon tatsächlich genutzt werden – auf den ADFS-Servern selbst und über die AD-FS-Anwendungsmigration in Entra ID, die Connect Health mit Daten füttert. Und du sortierst jede Anwendung in eine von vier Schubladen: trivial, machbar, schwierig, bleibt. Wie ADFS grundsätzlich funktioniert und wo es im Unternehmen steht, beschreibt die Übersichtsseite [Active Directory Federation Services](#). Dieser Beitrag setzt voraus, dass du eine laufende Farm hast und PowerShell nicht für eine Muschelart hältst.

FAKTEN — Bestandsaufnahme der Relying Parties in Kürze

Die vollständige Konfiguration liefert nur die Farm selbst: Get-AdfsRelyingPartyTrust zeigt alle Relying Parties, auch deaktivierte und längst vergessene, mit Bezeichnern, Endpunkten, Regeln und Zertifikaten.

Die Nutzung steht im Sicherheitsprotokoll der ADFS-Server: Bei eingeschaltetem Auditing protokolliert ADFS ab Windows Server 2016 jede Token-Ausstellung als Ereignis 1200, inklusive Benutzer und Relying Party.

Entra ID bewertet Relying Parties unter „Usage & insights“ > „AD FS application migration“.

Voraussetzung sind die Connect-Health-Agents für AD FS und eine Lizenz Entra ID P1 oder P2. Angezeigt werden nur Anwendungen mit Anmeldungen in den letzten 30 Tagen, Microsoft-eigene Relying Parties wie Microsoft 365 fehlen bewusst.

Den Migrationsstatus vergibt Entra ID in drei Stufen: „Ready to migrate“, „Needs review“ und „Additional steps required“. Neu berechnet wird er einmal täglich gegen 00:00 Uhr UTC.

Warum die Inventur vor jeder Migration steht

Man könnte meinen, das Inventar sei schnell gemacht: ADFS-Verwaltungskonsole öffnen, Ordner „Relying Party Trusts“ anklicken, Liste abschreiben, fertig. Das funktioniert genau bis zur ersten Rückfrage. Die Liste in der Konsole sagt dir, dass es eine Relying Party namens „Portal-Extern“ gibt. Sie sagt dir nicht, ob sich dort täglich 2.000 Leute anmelden oder seit vier Jahren niemand, welche Claims die Anwendung erwartet, wer im Fachbereich dafür geradesteht und ob Entra ID mit ihren Regeln überhaupt etwas anfangen kann.

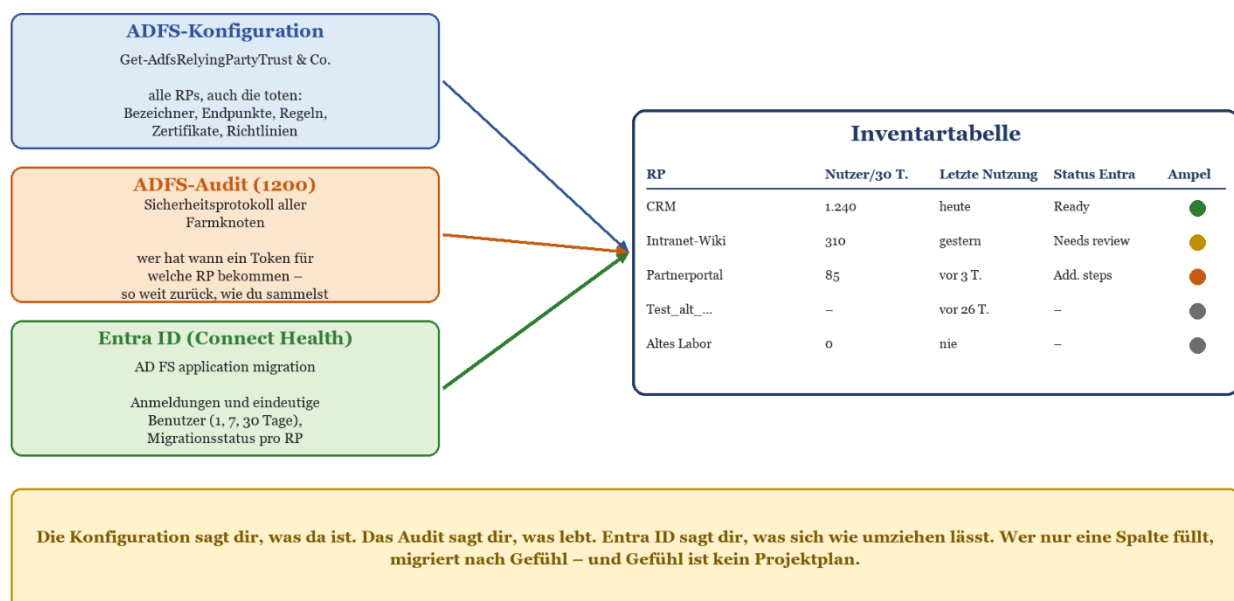
Die Migration nach Entra ID scheitert selten an der Technik. Sie scheitert an Überraschungen: an der Anwendung, die keiner auf dem Zettel hatte, an der Claim-Regel mit dem regulären Ausdruck, den ein Dienstleister 2016 eingebaut hat, an der Partnerorganisation, die über einen Claims Provider Trust hereinkommt und von deren Existenz die IT-Leitung nichts wusste. Das Inventar ist das Werkzeug, mit dem du diese Überraschungen in den Projektplan holst, bevor sie dich am Freitagnachmittag finden.

Drei Blickwinkel, drei Wahrheiten

Für eine belastbare Bestandsaufnahme brauchst du drei Blickwinkel, die sich gegenseitig ergänzen. Die Farmkonfiguration ist vollständig, aber stumm: Sie weiß nichts über Nutzung. Das Audit auf den ADFS-Servern kennt die Nutzung, aber nur so weit zurück, wie dein Sicherheitsprotokoll reicht. Entra ID liefert eine fertige Bewertung, sieht aber nur 30 Tage und blendet Microsoft-Anwendungen aus.

Skizze 1: Drei Quellen, ein Inventar

Keine Quelle allein reicht – erst zusammen beantworten sie: Was ist konfiguriert, was wird genutzt, was geht nach Entra?



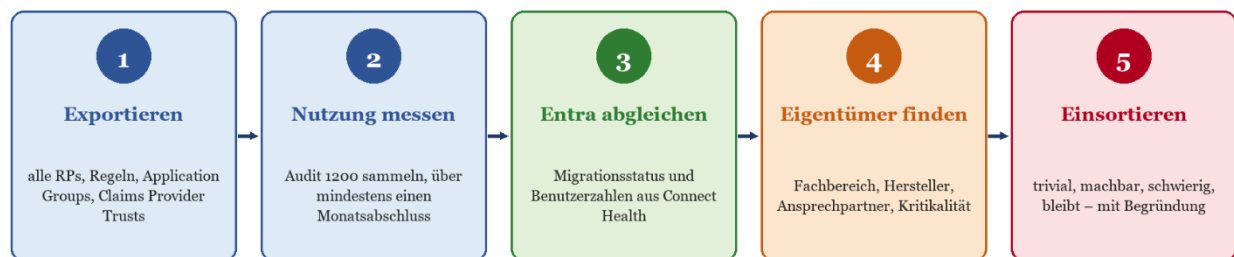
Skizze 1: Konfiguration, Audit und Entra-Bewertung ergeben zusammen das Inventar

Quelle	Liefert	Liefert nicht	Typischer Stolperstein
ADFS-Konfiguration (PowerShell)	alle RPs, Bezeichner, Endpunkte, Regeln, Richtlinien, Zertifikate, Notizen	Nutzung, Eigentümer, Kritikalität	tote RPs sehen genauso wichtig aus wie lebende
ADFS-Audit (Ereignis 1200)	Token pro RP, eindeutige Benutzer, letzter Zugriff – beliebig lange, wenn du sammelst	Bewertung der Migrierbarkeit	Sicherheitsprotokoll läuft auf großen Farmen in Tagen über
Entra ID: AD FS application migration	Anmeldungen und Benutzer für 1, 7 und 30 Tage, Migrationsstatus, Regelanalyse	RPs ohne Anmeldung in 30 Tagen, Microsoft-RPs	monatliche Anwendungen fallen durchs Raster
Fachbereiche und Dokumentation	Eigentümer, Geschäftsprozess, Hersteller, Vertragslage	irgendetwas Technisches	„Das hat damals die IT eingerichtet“

Tabelle 1: Was welche Quelle zum Inventar beiträgt

Skizze 2: Die Bestandsaufnahme in fünf Schritten

Erst sichern, dann zählen, dann fragen, dann bewerten – nie in umgekehrter Reihenfolge



Faustregel für den Zeitplan

Schritt 1 dauert eine Stunde. Schritt 2 dauert so lange wie dein längster Geschäftszyklus – Lohnlauf, Quartalsabschluss, Jahresinventur. Schritt 4 dauert erfahrungsgemäß länger als alle anderen zusammen, weil niemand zugeben will, dass ihm die Anwendung gehört.

Skizze 2: Die Reihenfolge der Bestandsaufnahme

WICHTIG – Erst sichern, dann aufräumen

Die Versuchung ist groß, beim Inventarisieren gleich auszumisten. Widersteh ihr, bis Schritt 1 erledigt ist: Ohne Export ist eine gelöschte Relying Party samt Regeln weg. Wie du die ganze Farm zusätzlich sicherst, beschreibt [ADFS sichern und wiederherstellen – Rapid Restore Tool in der Praxis](#).

Alle Relying Parties exportieren – mit Regeln

Der Export hat zwei Zwecke. Erstens ist er deine Versicherung: Mit einem vollständigen Abzug kannst du jede Relying Party wiederherstellen oder zumindest nachbauen. Zweitens ist er Rohmaterial für die Bewertung. Beides erledigst du auf dem primären Knoten der Farm – bei WID der Server, auf dem Get-AdfsSyncProperties die Rolle „PrimaryComputer“ meldet, bei SQL jeder beliebige Knoten – in einer administrativen PowerShell-Sitzung.

Der Komplettabzug für den Notfall

Export-Clixml schreibt die Objekte mit allen Eigenschaften in XML-Dateien. Das ist kein Format zum Lesen, aber das beste zum Wiederherstellen, weil Import-Clixml dir später exakt dieselben Eigenschaften zurückgibt. Exportiere dabei nicht nur die Relying Parties, sondern alles, was an ihnen hängt.

```
$ziel = 'C:\ADFS-Inventar\' + (Get-Date -Format 'yyyy-MM-dd')
New-Item -ItemType Directory -Path $ziel -Force | Out-Null

Get-AdfsRelyingPartyTrust      | Export-Clixml "$ziel\RelyingPartyTrusts.xml"
Get-AdfsClaimsProviderTrust   | Export-Clixml "$ziel\ClaimsProviderTrusts.xml"
Get-AdfsApplicationGroup      | Export-Clixml "$ziel\ApplicationGroups.xml"
Get-AdfsServerApplication     | Export-Clixml "$ziel\ServerApplications.xml"
Get-AdfsWebApiApplication     | Export-Clixml "$ziel\WebApiApplications.xml"
Get-AdfsNativeClientApplication | Export-Clixml "$ziel\NativeClientApplications.xml"
Get-AdfsAccessControlPolicy    | Export-Clixml "$ziel\AccessControlPolicies.xml"
Get-AdfsProperties            | Export-Clixml "$ziel\FarmProperties.xml"

# Kurzer Überblick, damit du weißt, womit du es zu tun hast
'{0} Relying Parties, davon {1} deaktiviert' -f `
    (Get-AdfsRelyingPartyTrust).Count,
    (Get-AdfsRelyingPartyTrust | Where-Object { -not $_.Enabled }).Count
```

Listing 1: Vollständiger Abzug aller Vertrauensstellungen, Anwendungen und Richtlinien

Die Application Groups gehören unbedingt dazu. Moderne OAuth- und OpenID-Connect-Anwendungen stehen in ADFS nicht als klassische Relying Party in der Liste, sondern als Web-API mit zugehöriger Server- oder Client-Anwendung in einer Application Group. Wer nur Get-AdfsRelyingPartyTrust exportiert, übersieht sie – und gerade diese Anwendungen sind oft die, die eine Entwicklungsabteilung ohne viel Aufhebens angebunden hat. Wie sie aufgebaut sind, erklärt [OAuth 2.0 und OpenID Connect mit ADFS Application Groups](#). Ebenso wichtig sind die Claims Provider Trusts: Jeder Eintrag außer „Active Directory“ ist eine Partnerorganisation oder ein weiterer Identitätsanbieter, und die tauchen in keiner Nutzungsstatistik als eigene Anwendung auf. Hintergründe stehen in [Claims Provider Trust – Partnerorganisationen anbinden](#).

WARNUNG — Der Export ist ein Konfigurationsabzug, keine Datensicherung

Export-Clixml enthält öffentliche Zertifikate und die Regeln, aber keine privaten Schlüssel und keine Client-Geheimnisse von Server-Anwendungen. Für eine Wiederherstellung der kompletten Farm samt Token-Signing-Zertifikat brauchst du eine echte Sicherung. Und: Die Dateien verraten einem Angreifer die komplette Anwendungslandschaft. Leg sie nicht auf die Abteilungsfreigabe.

Eine Inventarzeile pro Relying Party

Für die Bewertung brauchst du etwas, das in Excel passt. Das folgende Skript erzeugt pro Relying Party eine Zeile mit den Eigenschaften, die für die Migration den Ausschlag geben: Protokoll, Zugriffsrichtlinie, Anzahl der Transformationsregeln, verwendete Attributspeicher, MFA-Regeln, Metadaten-Überwachung und Verschlüsselung. Das Notizfeld nimmst du gleich mit – mit etwas Glück steht dort sogar, wem die Anwendung gehört.

```
$inventar = foreach ($rp in Get-AdfsRelyingPartyTrust) {
    $regeln = [string]$rp.IssuanceTransformRules
    $stores = [regex]::Matches($regeln, 'store\s*=?\s*"([^\"]+)"') |
        ForEach-Object { $_.Groups[1].Value } | Sort-Object -Unique

    [pscustomobject]@{
        Name           = $rp.Name
        Aktiv          = $rp.Enabled
        Bezeichner     = $rp.Identifizier -join ';'
        Protokoll      = if ($rp.SamlEndpoints) { 'SAML' }
                     elseif ($rp.WSFedEndpoint) { 'WS-Fed' }
                     else { 'kein passiver Endpunkt (WS-Trust o. Ä.)' }
        Zugriffsrichtlinie = $rp.AccessControlPolicyName
        TransformRegeln = ([regex]::Matches($regeln, '=>')).Count
        Attributspeicher = $stores -join ','
        RegexInRegeln  = $regeln -match 'RegexReplace|=?'
        AutorisierungAlt = [bool]$rp.IssuanceAuthorizationRules
        MfaRegeln      = [bool]$rp.AdditionalAuthenticationRules
    }
```

```

ClaimsProvider      = $rp.ClaimsProviderName -join ', '
Metadaten           = $rp.MetadataUrl
AutoUpdate          = $rp.AutoUpdateEnabled
SignierteRequests   = $rp.SignedSamlRequestsRequired
Verschlüsselt       = $rp.EncryptClaims -and [bool]$rp.EncryptionCertificate
TokenLebensdauer    = $rp.TokenLifetime
Notizen             = $rp.Notes
    }
}
$inventar | Export-Csv "$ziel\RP-Inventar.csv" -NoTypeInformation -Encoding UTF8 -Delimiter ';'
$inventar | Format-Table Name, Aktiv, Protokoll, TransformRegeln, Attributspeicher -AutoSize

```

Listing 2: Inventartabelle mit den migrationsrelevanten Eigenschaften

Die Zählung der Regeln über das Zeichen „=>“ ist bewusst grob – sie trennt die Anwendung mit drei Pass-Through-Regeln von der mit 27 Regeln und zwei SQL-Abfragen, und mehr soll sie an dieser Stelle nicht. Ein Attributspeicher außer „Active Directory“ ist dagegen ein hartes Signal: Entra ID kann Claims nicht aus eigenen SQL- oder LDAP-Quellen beziehen. Diese Anwendung landet nicht in der Schublade „trivial“, egal wie harmlos ihr Name klingt.

Die Regeln lesbar ablegen

Für die spätere Übersetzung nach Entra-Claims und Conditional Access willst du jede Regelsammlung im Klartext vor dir haben, nicht als Zelle in einer CSV-Datei. Eine Textdatei pro Relying Party ist unspektakulär und genau richtig – sie lässt sich vergleichen, kommentieren und dem Hersteller schicken.

```

foreach ($rp in Get-AdfsRelyingPartyTrust) {
    $datei = Join-Path $ziel ('Regeln_' + ($rp.Name -replace '[\\/:?*"<>|]', '_') + '.txt')
    @(
        "# $($rp.Name) ($($rp.Identifizier -join ', '))"
        "# Zugriffsrichtlinie: $($rp.AccessControlPolicyName)"
        '', '## IssuanceTransformRules', $rp.IssuanceTransformRules
        '', '## IssuanceAuthorizationRules', $rp.IssuanceAuthorizationRules
        '', '## AdditionalAuthenticationRules', $rp.AdditionalAuthenticationRules
    ) | Set-Content -Path $datei -Encoding UTF8
}

```

Listing 3: Claim Rules pro Relying Party als Textdatei

Wer beim Lesen der Regeln feststellt, dass die Claim Rule Language doch nicht mehr ganz so präsent ist wie beim Einrichten, findet die Syntax im [ADFS Claim Rules Leitfaden](#). Für die eigentliche Übersetzung in Entra-Bordmittel gibt es einen eigenen Beitrag: [Claim Rules nach Conditional Access übersetzen](#).

Eigenschaft	Was sie für die Migration bedeutet	Tendenz
SamlEndpoints / WSFedEndpoint	SAML ist der einfachste Weg; WS-Fed geht in Entra ID ebenfalls, aber nur mit einem Endpunkt	trivial bis machbar
kein passiver Endpunkt	aktive Clients über WS-Trust oder Sonderfälle – genauer hinsehen	schwierig
AccessControlPolicyName	Standardrichtlinien lassen sich meist als Zuweisung plus Conditional Access abbilden	machbar
IssuanceAuthorizationRules	alte Autorisierungsregeln ohne Richtlinie: Logik erst verstehen, dann übersetzen	machbar
AdditionalAuthenticationRules	MFA-Logik wird zur Conditional-Access-Richtlinie	machbar
Attributspeicher ≠ AD	SQL, LDAP oder eigene Attributspeicher gibt es in Entra ID so nicht	schwierig
Regex in Regeln	Entra kennt Transformationen wie Extract(), Contains() oder ToLower(), aber keine freien Ausdrücke	machbar bis schwierig
ClaimsProviderName mit Partnern	Partner-IdPs werden zu B2B-Gästen oder externer Föderation – ein Projekt für sich	schwierig

SignedSamlRequestsRequired	Entra ID nimmt signierte Requests an, prüft die Signatur aber nicht	klären
EncryptClaims / Verschlüsselungszertifikat	Token-Verschlüsselung kann Entra ID; Verschlüsselung nur der NameID nicht	meist unkritisch
AutoUpdate / MetadataUrl	automatische Metadatenpflege gibt es in Entra ID nicht – kein Blocker, aber Pflegeaufwand	unkritisch

Tabelle 2: Eigenschaften einer Relying Party und ihre Bedeutung für den Umzug

Nutzungsdaten auswerten – was lebt, was ist tot?

Ein Inventar ohne Nutzungsdaten ist eine Inventurliste ohne Mengen: Du weißt, dass es Schrauben gibt, aber nicht, ob es drei oder dreitausend sind. Bei Relying Parties ist das noch heikler, weil die Nutzung extrem ungleich verteilt ist. Ein paar Anwendungen machen den Großteil der Anmeldungen aus, ein langer Schwanz wird einmal im Monat oder einmal im Jahr benutzt – und genau in diesem Schwanz verstecken sich die Lohnbuchhaltung, das Portal für die Jahresinventur und die Plattform, auf der der Wirtschaftsprüfer einmal im Frühjahr Dokumente abholt.

Ereignis 1200 auf den ADFS-Servern auswerten

Mit eingeschaltetem Auditing schreibt ADFS ab Windows Server 2016 für jedes ausgestellte Token das Ereignis 1200 ins Sicherheitsprotokoll, Quelle „AD FS Auditing“. Im Nachrichtentext steckt ein XML-Block mit Benutzer, Relying Party, Client-IP und Endpunkt. Damit das Ereignis überhaupt entsteht, muss das Dienstkonto Sicherheitsüberwachungen erzeugen dürfen, die Unterkategorie „Anwendung generiert“ in auditpol muss Erfolg protokollieren, und die Farm muss Erfolgsüberwachungen schreiben. Wie du das einrichtest und prüfst, beschreibt [ADFS-Ereignisprotokolle lesen – Admin-Log, Debug-Tracing und Auditing](#). Hier geht es um die Auswertung.

```
# Voraussetzungen prüfen
auditpol.exe /get /subcategory:"{0cce9222-69ae-11d9-bed3-505054503030}"
Get-AdfsProperties | Select-Object LogLevel, AuditLevel

# Token-Ausstellungen der letzten 30 Tage von allen Farmknoten einsammeln
$server = 'adfs01.contoso.local', 'adfs02.contoso.local'
$seit = (Get-Date).AddDays(-30)

$ereignisse = foreach ($s in $server) {
    Get-WinEvent -ComputerName $s -ErrorAction SilentlyContinue -FilterHashtable @{
        LogName = 'Security'; ProviderName = 'AD FS Auditing'; Id = 1200; StartTime = $seit
    }
}

$nutzung = foreach ($e in $ereignisse) {
    [pscustomobject]@{
        Zeit = $e.TimeCreated
        RP = [regex]::Match($e.Message,
        '<RelyingParty>(.*?)</RelyingParty>').Groups[1].Value
        Benutzer = [regex]::Match($e.Message, '<UserId>(.*?)</UserId>').Groups[1].Value
    }
}

$nutzung | Group-Object RP | ForEach-Object {
    [pscustomobject]@{
        RP = $_.Name
        Token = $_.Count
        Benutzer = ($_.Group.Benutzer | Sort-Object -Unique).Count
        Zuletzt = ($_.Group.Zeit | Measure-Object -Maximum).Maximum
    }
} | Sort-Object Benutzer -Descending |
Export-Csv "$ziel\RP-Nutzung.csv" -NoTypeInfo -Encoding UTF8 -Delimiter ';'

```

Listing 4: Token-Ausstellungen pro Relying Party zählen

Die GUID im auditpol-Aufruf steht für die Unterkategorie „Anwendung generiert“ und funktioniert unabhängig von der Sprache des Betriebssystems – auf einem deutschen Server ist das angenehmer, als den lokalisierten Namen zu raten. In der Spalte RP steht der Bezeichner der Relying Party, nicht ihr Anzeigename. Den Abgleich mit dem Inventar aus Listing 2 machst du über die Spalte Bezeichner.

TIPP — Erst ein Ereignis ansehen, dann dem Skript vertrauen

Öffne vor dem ersten Lauf ein einzelnes 1200-Ereignis in der Ereignisanzeige und schau dir den XML-Teil an. Je nach Konfiguration und Version stehen Werte leicht anders da, und ein Regex, der leer zurückkommt, liefert eine wunderschön formatierte Tabelle voller Nullen. Das sieht nach Aufräumpotenzial aus und ist in Wahrheit ein Auswertungsfehler.

Das Messfenster: länger als dein längster Geschäftszyklus

Hier liegt die eigentliche Falle. Auf einer gut besuchten Farm ist das Sicherheitsprotokoll nach wenigen Tagen voll, und die ältesten Einträge sind weg. Ein „letzte 30 Tage“ im Skript nützt nichts, wenn im Protokoll nur fünf Tage stehen. Und selbst 30 saubere Tage reichen nicht für Anwendungen, die im Monats-, Quartals- oder Jahresrhythmus benutzt werden.

- Vergrößere das Sicherheitsprotokoll auf den ADFS-Servern oder leite die Ereignisse per Windows Event Forwarding oder an dein SIEM weiter.
- Alternativ lässt du Listing 4 als geplante Aufgabe täglich laufen und hängst das Ergebnis an eine Sammeldatei an. Das ist hemdsärmelig, aber nach drei Monaten hast du eine Nutzungshistorie, die kein Portal liefert.
- Wenn Connect Health ohnehin installiert ist, kannst du die ADFS-Anmeldungen aus Entra ID per Diagnoseeinstellung in einen Log-Analytics-Arbeitsbereich schicken und dort mit selbst gewählter Aufbewahrung auswerten.
- Plane das Messfenster so, dass mindestens ein Monatsabschluss und möglichst ein Quartalsende darin liegen. Was dann noch null Token hat, ist ein Kandidat zum Abschalten – nicht zum Löschen.

AD-FS-Anwendungsmigration in Entra ID

Microsoft hat dir einen Teil der Arbeit bereits abgenommen – vorausgesetzt, die Connect-Health-Agents für AD FS laufen auf deinen Farmknoten. Die Agents lesen die Konfiguration der Relying Parties und die Anmelde-Audits und laden beides nach Entra ID hoch. Im Entra Admin Center findest du die Auswertung unter „Enterprise apps“ > „Usage & insights“ > „AD FS application migration“. Lesend reichen die Rollen Globaler Leser oder Berichtsleser, für die Ein-Klick-Migration brauchst du mindestens Cloudanwendungsadministrator. Lizenzseitig setzt Microsoft Entra ID P1 oder P2 voraus. Wie du die Agents installierst und überwachst, steht in [ADFS überwachen – Health Checks, Diagnose-Toolbox und Entra Connect Health](#).

Das Dashboard hat drei Reiter. „All apps“ listet alle Relying Parties mit Anmeldungen im gewählten Zeitraum – 1, 7 oder 30 Tage – samt Anzahl der Anmeldungen und eindeutigen Benutzer. „Ready to migrate“ zeigt die Anwendungen mit Status „Ready“ oder „Needs review“, „Ready to configure“ die bereits über den Assistenten angelegten Entra-Anwendungen. Spannend für das Inventar ist der Migrationsstatus pro Relying Party und die Detailansicht dahinter: Entra ID prüft jede Relying Party gegen eine Reihe von Tests und zerlegt die Claim Rules einzeln in „geht“ und „geht so nicht“.

Entra-Status	Bedeutung laut Microsoft	Übersetzt für dein Inventar
Ready to migrate	Konfiguration wird in Entra ID	Kandidat für „trivial“ – nach

	vollständig unterstützt	Gegenprüfung
Needs review	Teile lassen sich nicht 1:1 übernehmen, sind aber kein Blocker	meist „machbar“: Regeln übersetzen, CA bauen
Additional steps required	Einstellungen werden in Entra ID nicht unterstützt, Migration im aktuellen Zustand nicht möglich	„schwierig“ oder „bleibt“ – Ursache klären
nicht in der Liste	keine Anmeldungen in 30 Tagen oder Microsoft-eigene RP	selbst bewerten, Nutzung aus eigenem Audit

Tabelle 3: Migrationsstatus in Entra ID und was du daraus machst

Die Tests selbst sind aufschlussreich, weil sie zeigen, wo Entra ID anders tickt als ADFS. Einige typische Befunde aus der Detailansicht:

Test bzw. Befund	Ergebnis	Was dahintersteckt
Test-ADFSRPAdditionalAuthenticationRules	Pass/Warning	MFA-Regeln vorhanden – werden zu Conditional Access
Test-ADFSRPIssuanceTransformRules	Pass/Warning	nicht übertragbare Transformationsregeln, Einzelanalyse pro Regel
Test-ADFSRPAdditionalWSFedEndpoint	Pass/Fail	mehrere WS-Fed-Endpunkte; Entra ID unterstützt einen
Test-ADFSRPClaimsProviderName	Pass/Fail	mehrere Claims Provider aktiv – Partner-Szenario, B2B prüfen
Test-ADFSRPSignedSamlRequestsRequired	Pass/Fail	Signaturprüfung des Requests verlangt; Entra prüft sie nicht
Test-ADFSRPTokenLifetime	Pass/Warning	eigene Token-Lebensdauer; in Entra über Sitzungssteuerung in CA
EXTERNAL_ATTRIBUTE_STORE	Regelbefund	Claims aus einem anderen Speicher als AD – in Entra ID nicht möglich
UNSUPPORTED_CONDITION_PARAMETER	Regelbefund	Bedingung mit regulärem Ausdruck – mit Transformationen nachbauen

Tabelle 4: Auswahl der Prüfungen der AD-FS-Anwendungsmigration

Die Zahlen lassen sich auch per Microsoft Graph abholen und neben dein eigenes Inventar legen. Das Cmdlet stammt aus dem Modul Microsoft.Graph.Reports und braucht die Berechtigung Reports.Read.All. Früher stand an dieser Stelle in jeder Anleitung irgendein Aufruf aus den Modulen MSONline oder AzureAD – die sind ausgemustert und haben in neuen Skripten nichts mehr verloren.

```
Connect-MgGraph -Scopes 'Reports.Read.All'

Get-MgReportRelyingPartyDetailedSummary -Period 'D30' |
    Select-Object RelyingPartyName, RelyingPartyId, UniqueUserCount,
        TotalSignInCount, FailedSignInCount, MigrationStatus |
    Sort-Object UniqueUserCount -Descending |
    Export-Csv "$ziel\Entra-RP-Bewertung.csv" -NoTypeInfo -Encoding UTF8 -Delimiter ';'

```

Listing 5: Nutzung und Migrationsstatus aus Entra ID per Graph PowerShell

WICHTIG – Das 30-Tage-Fenster ist eine Momentaufnahme

Die Entra-Auswertung zeigt ausschließlich Relying Parties mit Anmeldungen in den letzten 30 Tagen. Läuft deine Lohnabrechnung am 25. und schaust du am 27. des Folgemonats ins Portal, fehlt sie in der Liste. Nimm die Entra-Daten deshalb als Ergänzung zur eigenen Messung, nie als deren Ersatz. Und weil der Status nur einmal täglich neu berechnet wird, siehst du Änderungen an einer Relying Party erst am nächsten Tag.

FAKTEN – Was die Ein-Klick-Migration kann – und was nicht

Der Assistent unterstützt nur SAML-Anwendungen. Er übernimmt Bezeichner und Antwort-URL,

kompatible Claims und die Gruppen aus der Zugriffsrichtlinie, sofern sie nach Entra ID synchronisiert sind.

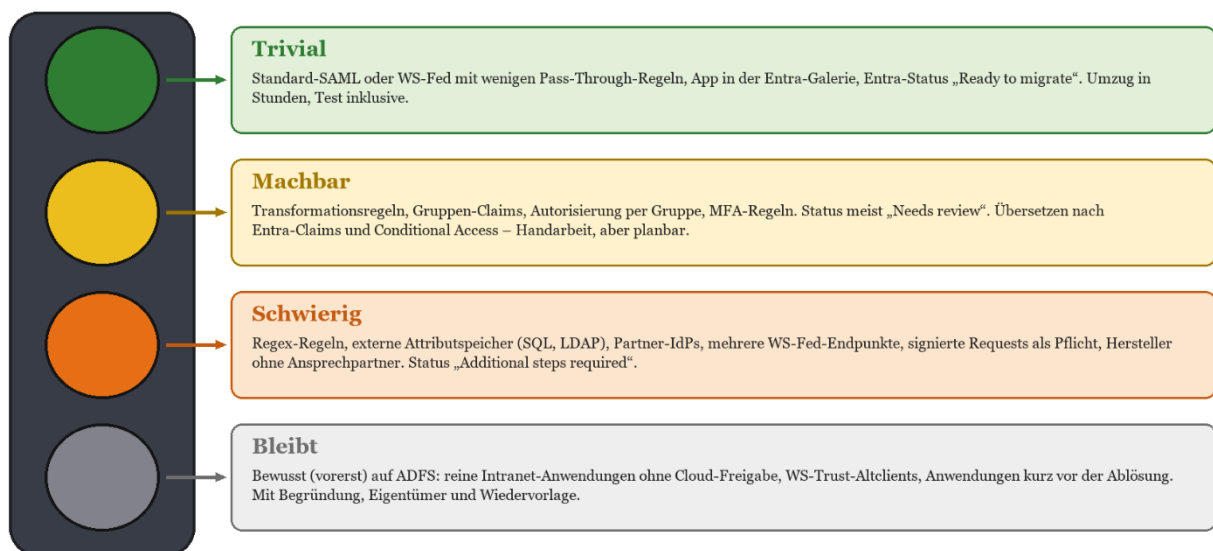
WS-Fed, OAuth und OpenID Connect richtest du von Hand ein. Conditional-Access-Richtlinien legt der Assistent nicht an, das Signaturzertifikat wird nicht übernommen. Die neue Anwendung bleibt inaktiv, bis du den Anmeldeverkehr umstellst – aufräumen musst du im Zweifel selbst.

Kategorisieren: trivial, machbar, schwierig, bleibt

Jetzt liegen drei Tabellen auf dem Tisch: Konfiguration, eigene Nutzungsdaten, Entra-Bewertung. Daraus wird eine Entscheidung pro Anwendung. Bewährt hat sich eine Ampel mit vier statt drei Farben. Die vierte Kategorie „bleibt“ ist kein Eingeständnis des Scheiterns, sondern ehrliche Projektplanung: Es gibt Anwendungen, die aus guten Gründen vorerst auf ADFS bleiben, und es ist besser, das offen hinzuschreiben, als sie in „schwierig“ verrotten zu lassen.

Skizze 3: Die Migrations-Ampel

Vier Kategorien statt drei Farben – denn manche Anwendung bleibt einfach, wo sie ist



Skizze 3: Die Migrations-Ampel mit vier Kategorien

Kategorie	Kriterien	Typische Beispiele	Nächster Schritt
Trivial	SAML oder WS-Fed, wenige Pass-Through-Regeln, nur AD als Quelle, Entra-Status „Ready“, Hersteller dokumentiert Entra ID	SaaS aus der Entra-Galerie: CRM, Reisekosten, Lernplattform, Ticketsystem	in Wellen migrieren, Pilotgruppe, Test, Umstellung beim Hersteller
Machbar	Transformationsregeln, Gruppen-Claims, Autorisierung per Gruppe, MFA-Regeln, eigene Token-Lebensdauer; Status „Needs review“	Intranet-Wiki mit Gruppenrollen, Personalportal mit Abteilungs-Claim	Regeln einzeln übersetzen, Conditional Access entwerfen, mit Testkonten prüfen
Schwierig	externe Attributspeicher, Regex-Logik, Partner-IdPs, mehrere WS-Fed-Endpunkte, signierte Requests als Pflicht, Hersteller ohne Entra-Erfahrung; Status „Additional steps required“	Partnerportal mit Claims Provider Trust, Anwendung mit SQL-Rollenabfrage, Eigenentwicklung von 2014	Ursache je Blocker klären, Alternativen bewerten, eigenes Teilprojekt mit Budget
Bleibt	kein sinnvoller Zielweg oder	Fertigungssystem im	Begründung, Eigentümer

	bewusste Entscheidung: reine Intranet-Nutzung ohne Cloud-Freigabe, Ablösung ohnehin geplant, WS-Trust-Altclients	abgeschotteten Netz, Anwendung mit Restlaufzeit von einem Jahr	und Wiedervorlage dokumentieren, Farm schlank weiterbetreiben
--	--	--	---

Tabelle 5: Die vier Kategorien der Migrations-Ampel

Die Sonderfälle, die in keine Schublade passen

Die Microsoft-365-Vertrauensstellung „Microsoft Office 365 Identity Platform“ gehört in kein normales Anwendungsinventar. Sie ist keine Anwendung, die du migrierst, sondern die Föderation deiner Domänen. Ihr Umzug ist die Umstellung von Federated auf Managed, typischerweise mit Staged Rollout – nachzulesen in [Microsoft 365 von Federated auf Managed umstellen – mit Staged Rollout](#). Schreib sie trotzdem in die Liste, mit eigener Kategorie „Domänenumstellung“, damit niemand auf die Idee kommt, sie beim Aufräumen zu deaktivieren.

SharePoint Server ist ein eigener Fall. Technisch lässt er sich per SAML auch an Entra ID anbinden, die Umstellung betrifft aber Benutzerprofile, Berechtigungen und gegebenenfalls Suchdienste. Ein „trivial“ ist das nie. Wie die Anbindung an ADFS aussieht, beschreibt [SharePoint Server mit ADFS – SAML-Authentifizierung einrichten](#). Anwendungen hinter dem Web Application Proxy mit Vorauthentifizierung tauchen ebenfalls als Relying Party auf und hängen in der Migration an der Frage, was den Proxy ablöst – dazu mehr in [Web Application Proxy – Architektur und Ablöse](#).

TIPP — Status aus Entra nie ungeprüft übernehmen

„Ready to migrate“ heißt, dass Entra ID die Konfiguration abbilden kann. Es heißt nicht, dass der Hersteller mitspielt, dass das Antwort-URL-Format stimmt oder dass die Anwendung den neuen Aussteller ohne Neustart akzeptiert. Ein kurzer Blick in die Herstelldokumentation und ein Test mit einem Pilotkonto gehören zu jeder grünen Ampel dazu.

Ein Beispielinventar

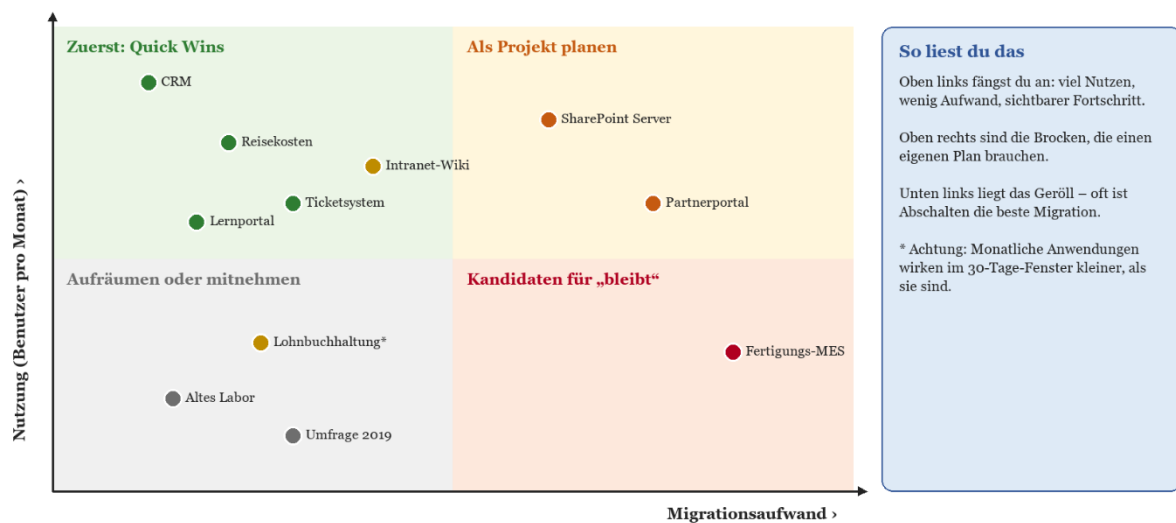
So sieht ein Ausschnitt aus einem echten – natürlich anonymisierten – Inventar einer mittelgroßen Farm aus. Bemerkenswert ist weniger die Verteilung als die letzte Spalte: Ohne Eigentümer gibt es keine Migration, sondern nur eine Vermutung.

Relying Party	Protokoll	Benutzer/30 T.	Entra-Status	Ampel	Eigentümer
CRM (SaaS)	SAML	1.240	Ready to migrate	trivial	Vertrieb
Reisekosten (SaaS)	SAML	860	Ready to migrate	trivial	Finanzen
Intranet-Wiki	WS-Fed	310	Needs review	machbar	Kommunikation
SharePoint Server	WS-Fed	1.900	–	schwierig	IT, Fachbereiche
Partnerportal	SAML	85	Additional steps required	schwierig	Einkauf
Fertigungs-MES	WS-Fed	140	Needs review	bleibt	Produktion
Lohnbuchhaltung	SAML	0 bzw. 38	– bzw. Ready	trivial*	Personal
Umfrage 2019	SAML	0	–	abschalten	unbekannt
Microsoft Office 365 Identity Platform	WS-Fed/WS-Trust	–	nicht gelistet	eigener Pfad	IT

Tabelle 6: Ausschnitt aus einem Beispielinventar (* je nach Stichtag mal sichtbar, mal nicht)

Skizze 4: Nutzung gegen Aufwand – die Reihenfolge ergibt sich fast von selbst

Jeder Punkt eine Relying Party aus einem anonymisierten Beispielinventar



Skizze 4: Nutzung und Aufwand ergeben die Migrationsreihenfolge

Eigentümer finden – die eigentliche Detektivarbeit

Die technische Analyse ist an einem Nachmittag erledigt. Die Frage, wem eine Anwendung gehört, beschäftigt dich Wochen. Ein paar Spuren führen erfahrungsgemäß schneller ans Ziel als die Rundmail an alle Abteilungen:

- Das Notizfeld der Relying Party und die Beschreibung in alten Change-Tickets – manchmal steht dort tatsächlich ein Name.
- Die Benutzer aus Listing 4: Wer sich häufig an einer Anwendung anmeldet, kennt meist auch den Ansprechpartner.
- Bezeichner und Antwort-URLs: Die Domäne im Bezeichner verrät oft den Hersteller, und der wiederum den Vertrag und damit die Kostenstelle.
- Das Zertifikat: Wer hat das Verschlüsselungs- oder Signaturzertifikat der Anwendung beantragt?

Trag die Ergebnisse gleich in das Notizfeld der Relying Party ein, etwa mit Set-AdfsRelyingPartyTrust -TargetName 'Intranet-Wiki' -Notes 'Eigentümer: Kommunikation; Inventur 10/2026; Ampel machbar'. Dann muss die nächste Person nicht wieder von vorn anfangen.

Die Relying Party, an der die Lohnbuchhaltung hing

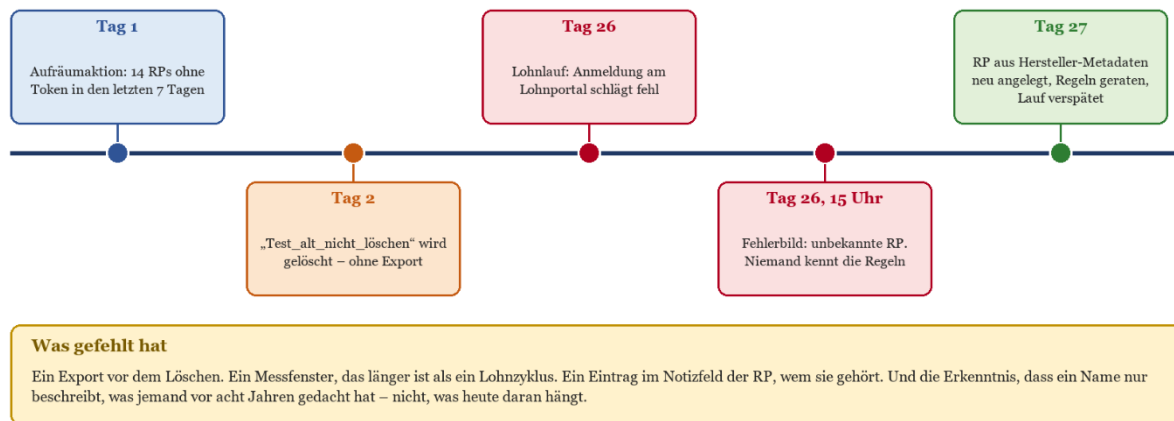
Und damit zur Geschichte, die in keinem Beitrag zu diesem Thema fehlen darf. Eine Farm, rund 60 Relying Parties, ein engagierter Admin mit einem Aufräumauftrag. Er wertet die Token-Ausstellungen der letzten sieben Tage aus – mehr gab das Sicherheitsprotokoll nicht her – und findet 14 Relying Parties ohne einen einzigen Zugriff. Eine davon heißt „Test_alt_nicht_löschen“. Der Name klingt nach einem Testsystem aus der Zeit vor dem letzten Server-Upgrade, die Antwort-URL zeigt auf einen Dienstleister, den niemand im Team kennt. Weg damit. Ohne Export, denn „ist ja nur ein Test“.

Drei Wochen später ist Lohnlauf. Die Personalabteilung kann sich am Portal des Abrechnungsdienstleisters nicht mehr anmelden. Der Dienstleister sieht nur „Fehler beim Identitätsanbieter“, ADFS meldet eine unbekannte Relying Party, und niemand weiß, welche

Claims das Portal erwartet hatte. Die Relying Party war vor acht Jahren als Test angelegt worden, dann ging der Test einfach in Produktion, und der Name blieb – mitsamt dem trotzigem „nicht löschen“, das vermutlich genau für diesen Moment gedacht war. Die Löhne gingen einen Tag später raus. Die Stimmung in der Personalabteilung brauchte etwas länger.

Skizze 5: Die kurze Geschichte von „Test_alt_nicht_löschen“

Anonymisiert, aber leider nicht erfunden – und ein Grund, warum Deaktivieren vor Löschen kommt



Skizze 5: Sieben Tage Messfenster, ein Monat Lohnzyklus – die Rechnung ging nicht auf

Die Lehre daraus ist nicht „lösche nie etwas“. Die Lehre ist: Erst exportieren, dann über einen vollen Geschäftszyklus messen, dann deaktivieren statt löschen. Eine deaktivierte Relying Party stellt keine Token mehr aus, lässt sich aber mit einem Befehl zurückholen. Meldet sich innerhalb eines Quartals niemand, kannst du sie mit gutem Gewissen entfernen – den Export hast du ja.

```
# Schreitest: deaktivieren, Datum notieren, beobachten
$rp = Get-AdfsRelyingPartyTrust -Name 'Test_alt_nicht_löschen'
Set-AdfsRelyingPartyTrust -TargetRelyingParty $rp `
    -Notes ($rp.Notes + " | Deaktiviert $(Get-Date -Format 'yyyy-MM-dd') im Rahmen der
Inventur")
Disable-AdfsRelyingPartyTrust -TargetName 'Test_alt_nicht_löschen'

# Wenn das Telefon klingelt
Enable-AdfsRelyingPartyTrust -TargetName 'Test_alt_nicht_löschen'

# Erst nach einem vollen Zyklus ohne Beschwerde
Remove-AdfsRelyingPartyTrust -TargetName 'Test_alt_nicht_löschen'
```

Listing 6: Deaktivieren statt Löschen – der Schreitest mit Rückweg

WARNUNG – Ein Name ist keine Dokumentation

„Test“, „alt“ und „temp“ im Namen einer Relying Party sagen nichts darüber aus, was heute daran hängt. Entscheidend sind Nutzungsdaten über einen ausreichend langen Zeitraum und ein bestätigter Eigentümer. Fehlt beides, ist Deaktivieren mit Beobachtungsphase der einzige verantwortbare Weg.

WEITER – Passende Beiträge für die nächsten Schritte

- › [SAML-Anwendungen von ADFS nach Entra ID migrieren](#)
- › [Claim Rules nach Conditional Access übersetzen](#)
- › [Microsoft 365 von Federated auf Managed umstellen – mit Staged Rollout](#)
- › [Relying Party Trust anlegen – per Metadaten und von Hand](#)
- › [ADFS abschalten – die Farm sauber außer Betrieb nehmen](#)
- › [Microsoft Entra ID](#)

Fazit: Erst zählen, dann ziehen

Eine ADFS-Farm, die seit zehn Jahren zuverlässig läuft, ist kein Problem, das man hektisch lösen müsste. Sie ist ein Bestand, den man kennen sollte, bevor man ihn umzieht. Die Bestandsaufnahme der Relying Parties ist dafür das Fundament – und sie ist weniger Arbeit, als die meisten fürchten. Der Export ist eine Sache von Minuten, die Auswertung der Token-Ausstellungen ein überschaubares Skript, und Entra ID liefert mit der AD-FS-Anwendungsmigration eine Bewertung frei Haus, sofern Connect Health läuft.

Die eigentliche Arbeit steckt woanders: im ausreichend langen Messfenster, in der ehrlichen Einordnung jeder Anwendung und in der Suche nach Eigentümern. Wer diese Arbeit macht, hat am Ende eine Liste, aus der sich ein Migrationsplan fast von selbst ergibt – mit schnellen Erfolgen am Anfang, eigenen Teilprojekten für die Brocken und einer offen dokumentierten Restmenge, die vorerst auf ADFS bleibt. Wer sie nicht macht, migriert nach Gefühl und lernt die Lohnbuchhaltung auf die harte Tour kennen. Wenn du für die Bewertung einen zweiten Blick von außen möchtest, findest du ihn im [Consulting zu ADFS \(Active Directory Federation Services\)](#). Inventur, Regelanalyse und Migrationsplanung vertieft außerdem [ADFS in der Praxis – das Buch im Detail](#), und wer das Vorgehen an einer Beispielfarm üben will, ist in den [ADFS-Schulungen](#) richtig.

FAQ: Relying Parties inventarisieren

Wie exportiere ich alle Relying Party Trusts aus ADFS?

Mit `Get-AdfsRelyingPartyTrust` | Export-Clixml auf einem Farmknoten. Exportiere zusätzlich Claims Provider Trusts, Application Groups, Access Control Policies und die Farmeigenschaften, damit dir keine OAuth-Anwendung und keine Partnerorganisation durchrutscht.

Wie sichere ich die Claim Rules einer Relying Party?

Die Regeln stehen in den Eigenschaften `IssuanceTransformRules`, `IssuanceAuthorizationRules` und `AdditionalAuthenticationRules`. Schreib sie pro Relying Party in eine Textdatei – das ist lesbar, vergleichbar und lässt sich bei Bedarf mit `Set-AdfsRelyingPartyTrust` wieder einspielen.

Wie finde ich heraus, welche Relying Parties noch genutzt werden?

Werte das Audit-Ereignis 1200 im Sicherheitsprotokoll aller ADFS-Server aus. Es enthält pro Token-Ausstellung Benutzer und Relying Party. Voraussetzung ist eingeschaltetes Auditing für die Unterkategorie „Anwendung generiert“.

Warum fehlen Anwendungen in der AD-FS-Anwendungsmigration in Entra ID?

Die Auswertung zeigt nur Relying Parties mit Anmeldungen in den letzten 30 Tagen. Microsoft-eigene Relying Parties wie die Microsoft-365-Vertrauensstellung blendet sie aus. Außerdem müssen die Connect-Health-Agents für AD FS auf den Farmknoten laufen.

Was bedeuten „Ready to migrate“, „Needs review“ und „Additional steps required“?

„Ready to migrate“ heißt, die Konfiguration ist in Entra ID vollständig abbildbar. „Needs review“ heißt, einzelne Einstellungen müssen angepasst werden, blockieren aber nicht. „Additional steps

required“ heißt, Entra ID unterstützt bestimmte Einstellungen nicht, die Anwendung lässt sich so nicht umziehen.

Kann ich die Entra-Bewertung per PowerShell abrufen?

Ja, mit `Get-MgReportRelyingPartyDetailedSummary` aus dem Modul `Microsoft.Graph.Reports` und der Berechtigung `Reports.Read.All`. Als Zeitraum gibst du zum Beispiel `D30` an.

Wie lange sollte ich die Nutzung messen?

Mindestens über deinen längsten regelmäßigen Geschäftszyklus, also über einen Monatsabschluss und möglichst ein Quartalsende. Sieben oder 30 Tage reichen für Anwendungen mit Monats- oder Jahresrhythmus nicht.

Soll ich ungenutzte Relying Parties sofort löschen?

Nein. Exportiere sie, deaktiviere sie mit `Disable-AdfsRelyingPartyTrust` und warte einen vollen Geschäftszyklus ab. Meldet sich niemand, kannst du sie entfernen.

Gehört die Microsoft-365-Vertrauensstellung ins Anwendungsinventar?

Ja, aber als Sonderfall. Sie wird nicht wie eine Anwendung migriert, sondern durch die Umstellung der Domänen von Federated auf Managed abgelöst.

Welche Relying Parties sind besonders schwer zu migrieren?

Anwendungen mit Claims aus SQL- oder LDAP-Attributspeichern, mit regulären Ausdrücken in den Regeln, mit Partner-Identitätsanbietern über Claims Provider Trusts, mit mehreren WS-Fed-Endpunkten oder mit zwingend signierten SAML-Requests.