MODULE-1-INTRO

סיכום המודולים שעברנו :

מודול 1 : הכרות -

עקרונות ZERO TRUST :

Zero Trust = “Never trust, always verify"

שלושה עקרונות:

1. **Verify Explicitly**
אימות על בסיס סיגנלים: מיקום, מכשיר, סיכון, זהות, Session.
2. **Least Privilege Access**
הרשאות מינימליות (RBAC, PIM).
3. **Assume Breach**
בדיקת גישה מתמדת, סגמנטציה, ניטור.

סוגי זהויות :
סוגי זהויות (Identities)

User Identities:

Cloud-only

Synced (Entra Connect / Cloud Sync)

External Identities:

B2B (Guest)

B2C (Customers)

Workload Identities:

Managed Identities

Service Principals

Applications

קישור להסבר על ארכיטקטורה וניהול זהויות :
https://learn.microsoft.com/en-us/azure/well-architected/security/identity-access

להכיר פרוטוקולי אימות :

נושא	הסבר קצר	דוגמאות / הערות
מה זה SAML	Security Assertion Markup Language – פרוטוקול אימות מבוסס XML	נפוץ במערכות ותיקות, אפליקציות ארגוניות, פורטלים
מטרה	לאפשר SSO בין זהויות (Identity Provider) לשירות (Service Provider)	מאפשר להתחבר פעם אחת ולגשת למספר מערכות
מבנה כללי	IdP מנפיק **Assertion** (טענת זהות) ל-SP	הכול מקודד ב-XML
IdP	Identity Provider – גורם שמבצע אימות	לדוגמה: Entra ID, ADFS, OKTA
SP	Service Provider – האפליקציה שאליה המשתמש ניגש	Salesforce, SAP, שירות פנימי
Assertion	מסמך XML המכיל את פרטי המשתמש לאחר אימות	כולל: NameID, Attributes, זמן, חתימה
Binding Methods	הדרך שבה המסר מועבר בין צדדים	HTTP Redirect, HTTP POST (הנפוץ ביותר)
Metadata	קבצי XML המתארים את התצורה	כל צד מייבא Metadata של הצד השני
NameID	מזהה המשתמש שנשלח ל-SP	לרוב: email / UPN
Attributes	תכונות נוספות שנשלחות	group, department, roles
חתימה (Signing)	IdP חותם על ה-Assertion	חובה לתוקף ויישור קו אבטחתי
הצפנה (Encryption)	ניתן להצפין את ה-Assertion עצמו	אופציונלי אך מומלץ בסביבה רגישה
Flow בסיסי	1. המשתמש ניגש ל-SP → 2. מופנה ל-IdP → 3. IdP מאמת → 4. שולח Assertion ל-SP → 5. SP מעניק גישה	התהליך מבוסס Redirect

נושא	הסבר קצר	דוגמאות / הערות
יתרונות	SSO, מבוסס סטנדרטים, עובד עם מערכות Legacy	נפוץ בארגונים גדולים
חסרונות	מבוסס XML (כבד), לא מתאים טוב למובייל, ללא Tokens מודרניים	OAuth/OIDC עדיפים לאפליקציות מודרניות
Endpoint חשובים	SSO URL, Logout URL, Metadata URL	מוגדרים בשני הצדדים
תמיכה ב-Entra ID	Entra ID יכול לעבוד כ-IdP עבור SAML	דרך Enterprise Applications
Provisioning	אפשר לשלוח Attributes למיפוי ב-SP	מוגדר ב-Attribute Mapping
Certificate Expiration	תעודת החתימה מתעדכנת כל 3 שנים (מיקרוסופט)	חובה לעדכן ב-SP לפני פקיעה

טבלת סיכום – Federation (פדרציה)

נושא	הסבר קצר	הערות / דוגמאות
מה זה Federation	מנגנון שמאפשר למשתמש מאזור זהויות אחד (Identity Provider) לבצע אימות מול שירות/אפליקציה שנמצאים באזור זהויות אחר	"Trust Relationship" בין שני גופים
מטרה	לאפשר SSO בין ארגונים / מערכות שונות, ללא צורך בניהול משתמשים בכל מערכת בנפרד	לדוגמה: ארגון A ניגש למערכת של ארגון B
IdP (Identity Provider)	הגורם שמבצע אימות ומנפיק Assertion / Token	Entra ID, ADFS, OKTA, Ping
SP / Relying Party	האפליקציה/שירות שסומכים על האימות של ה-IdP	Salesforce, SAP, אפליקציה פנימית
Trust Relationship	"יחסי אמון" מבוססי תעודות ומטא-נתונים בין IdP ל-SP	כל צד מייבא Metadata של הצד השני
פרוטוקולים נפוצים	SAML 2.0, WS-Federation, OpenID Connect	OAuth הוא Authorization, לא Authentication)

נושא	הסבר קצר	הערות / דוגמאות
Metadata	קובצי XML שמכילים הגדרות של IdP/SP – כתובות, תעודות, אלגוריתמים	חיוני להגדרת הפדרציה
Tokens / Assertions	המידע שמועבר מה-IdP ל-SP אחרי אימות	SAML Assertion / ID Token
Single Sign-On (SSO)	משתמש מתחבר פעם אחת ל-IdP ויכול לגשת לשירותים רבים	חוויית משתמש חלקה
Single Logout (SLO)	סגירת Session רוחבית. תלוי ככל הנראה ביישום	לא כל אפליקציה תומכת
Attribute Mapping	שליחת ערכים מה IdP ל-SP כמו email, group, role	מוגדרים ב-Claims
Certificate Requirements	נדרש חתימה על Assertion/Token. תעודה חייבת להתעדכן לפני פקיעה	Entra ID מחליף Signing Key אחת ל־3 שנים
Frequent Flows	חיבור אפליקציות צד שלישי, חיבור ארגונים, Single Sign-On חוצה תחומים	נפוץ במערכות Legacy
Federation vs Synchronization	Federation = אימות דרך IdP חיצוני. Synchronization = העתקת חשבונות לאותו Directory	
Federation vs SAML	פדרציה היא המנגנון הרחב; SAML הוא רק אחד הפרוטוקולים שמיישמים אותו	כמו "רכב" מול "טויוטה"
Federation ב-Entra ID	Entra ID יכול לעבוד כ-IdP או כ-SP	תלוי בתרחיש

טבלת השוואה – SAML vs OAuth 2.0 vs OIDC

נושא	SAML 2.0	OAuth 2.0	OpenID Connect (OIDC)
סוג פרוטוקול	Authentication	Authorization	Authentication + Authorization
מטרת הפרוטוקול	אימות משתמשים (SSO)	מתן הרשאות לאפליקציות לגשת ל-	אימות משתמש + פרופיל משתמש מודרני

נושא	SAML 2.0	OAuth 2.0	OpenID Connect (OIDC)
		API	
שימוש עיקרי	אפליקציות ארגוניות, מערכות Legacy, פורטלים	אפליקציות שרת–ל–שרת, גישה ל-APIs	אפליקציות מודרניות, מובייל, Web
פורמט הודעות	XML	JSON	JSON
Token Type	SAML Assertion	Access Token / Refresh Token	ID Token (JWT) + Access Token
מוציא הטוקן	IdP (Identity Provider)	Authorization Server	Authorization Server + OIDC Provider
איך מזהים משתמש?	Assertion עם NameID + Attributes	לא מבצע Authentication!	ID Token מכיל פרטי משתמש
Single Sign-On (SSO)	כן	לא	כן (מובנה בפרוטוקול)
Single Logout (SLO)	נתמך (לא תמיד זמין)	לא חלק מהפרוטוקול	מוגבל, לרוב דרך SAML/OIDC Hybrid
Bindings	HTTP Redirect / POST	HTTP / API Calls	HTTP Redirect / POST
סוג אפליקציות	Web מבוססות דפדפן	Web, Mobile, API, Daemons	Web, Mobile, SPA
Scenarios אופייניים	SAP, Salesforce Legacy, אפליקציות ארגוניות ישנות	Bots, Automation, Server-to-Server	SPA, Web, Mobile, מודרני
האם יש פרופיל משתמש מובנה?	כן (Attributes)	לא	כן (UserInfo Endpoint)
מה נדרש בצד האפליקציה?	תמיכה ב־SAML	טיפול ב־Tokens	ספריות OIDC סטנדרטיות
שימוש ב-Entra ID	Enterprise Apps	App Registrations / Enterprise Apps	App Registrations
האם יש Consent?	לא	כן	כן

נושא	SAML 2.0	OAuth 2.0	OpenID Connect (OIDC)
Security Model	חתימת XML	JWT Tokens + Scopes	JWT + Claims + Scopes
Delegated / Application Permissions	לא רלוונטי	כן	כן
קלות אינטגרציה	מורכב / XML	פשוט / JSON	הכי פשוט
תמיכה במובייל	חלשה	טובה מאוד	מצוינת
שימוש מומלץ	מערכות ותיקות	API Access	אפליקציות מודרניות + Login

תהליך אימות משתמש מול ENTRA :

```
User → App → Entra (Authorize) → Enter UPN
                                ↓
IdP Routing (Tenant, Domain, Federation)
                                ↓
Authentication (Pwd/MFA/FIDO2)
                                ↓
Conditional Access Evaluation
                                ↓
Token Issued (ID/Access)
                                ↓
App Validates Token & Grants Access
```

משתמש מקבל טוקן : ראינו את הטוקן בקישור : JSON Web Tokens - jwt.io

להבין את הרעיון שעומד מאחורי DID - Decentralized identity

מה זה DID – Decentralized Identifier?

DID (מזהה מבוזר) הוא מזהה דיגיטלי שהמשתמש *שולט בו באופן מלא*, ללא תלות בארגון מרכזי כמו Microsoft, Google או מדינה.

במקום שמאגר מרכזי ינהל את הזהות — השליטה עוברת **למשתמש עצמו**.

הרעיון המרכזי

במודל זהות רגיל:

- Entra ID / Google / Gov ID = הבעלים של הזהות
- המשתמש רק משתמש בה

ב-DID:

- המשתמש הוא **הבעלים של הזהות**
- המידע נשמר במכשיר/Wallet
- אין צורך בשרת מרכזי כדי לאמת את הזהות

זו זהות שמבוססת על עקרונות **Self-Sovereign Identity (SSI)** — Web3.

🧩 מרכיבי DID

רכיב	הסבר
DID	מחרוזת מזהה ייחודית (did:ion:123456...)
DID Document	מסמך המתאר איך לאמת את המזהה (מפתחות ציבוריים, Endpoints)
Verifiable Credential	תעודה דיגיטלית חתומה שהמשתמש נושא בארנק
Wallet	האפליקציה/מכשיר שמחזיק את הזהות והתעודות

🏛️ איך זה עובד בפועל?

1️⃣ משתמש יוצר **DID** מקומי בארנק (למשל Microsoft Authenticator)
2️⃣ המכשיר מייצר **זוג מפתחות** — פרטי (אצל המשתמש) וציבורי (מפורסם ב־DID Document)
3️⃣ גורם כלשהו (למשל אוניברסיטה/ארגון) מנפיק למשתמש **Verifiable Credential**
4️⃣ המשתמש מציג את התעודה לגוף אחר (Verifier)
5️⃣ הגוף בודק חתימה מול ה-DID Document בלי צורך במאגר זהויות מרכזי

🛠️ DID מול זהות רגילה (Entra ID / Federation)

נושא	זהות רגילה	DID
בעלות	הארגון שולט בזהות	המשתמש שולט בזהות

נושא	זהות רגילה	DID
אימות	מול מאגר אחד מרכזי	מבוזר, מבוסס מפתחות
תלות בשרת	חובה	לא
פרטיות	נתונים אצל הענן	נתונים אצל המשתמש
שימוש טיפוסי	SSO לארגון, SaaS	בדיקות זהות, תעודות קורס, הוכחת גיל/הסמכה

🔒 היתרונות של DID

- שליטה מלאה של המשתמש (No central authority)
- פרטיות טובה יותר
- מתאים לעולם רגולטורי (GDPR/Privacy)
- העברת תעודות בצורה מאובטחת בין גופים
- לא תלוי באינטרנט בזמן הצגה (Offline verification)

⚠️ חסרונות / אתגרים

- טכנולוגיה עדיין חדשה
- דרוש אימוץ מצד ארגונים (Issuers/Verifiers)
- ניהול מפתחות יכול להיות מורכב למשתמשים
- לא מחליף SSO ארגוני (עדיין צריך Entra לצורך SaaS/Enterprise access)

🟦 DID ב-Microsoft Entra (Verified ID)

Verified ID Microsoft Entra הוא שירות שמאפשר:

- יצירה וניהול של Verifiable Credentials
- הנפקת תעודות לעובדים/לקוחות (Employment / Certification / Education)
- אימות של תעודות מצד גופים אחרים
- שימוש ב-DID מבוסס ION (מעל רשת Bitcoin)

מפתחות נמצאים ב־Authenticator, לא בענן.

קישור : https://learn.microsoft.com/en-us/entra/verified-id/decentralized-identifier-overview

טבלת השוואה – Microsoft Entra B2B vs B2C

נושא	Entra B2B (External Identities)	Entra B2C (Customer Identity)
מטרה	שיתוף פעולה בין ארגונים (Partners, Vendors, Guests)	ניהול זהות של לקוחות/אזרחים (Public Users)
קהל יעד	עובדים מאירגונים אחרים (Partners)	לקוחות, משתמשי אפליקציות, משתמשי Web
ניהול משתמשים	הארגון המארח **לא הבעלים** של המשתמש – זהות מנוהלת בענן/IdP של הארגון המקורי	הארגון **הוא הבעלים** של זהות הלקוח (Directory ייעודי)
דרך התחברות	משתמשים נכנסים עם החשבון הקיים שלהם (Microsoft, Google, פנימי, Federation)	משתמשים מתחברים עם כל ספק זהות: Email, Google, Facebook, Phone, Local Account
Provisioning	Guest Object נוצר ב-Entra של הארגון	משתמשים נשמרים בתיקיית B2C Directory
Authentication Flow	מבוסס Entra ID הקיים של המשתמש	הגדרת Identity Providers באפליקציה
תשלום / עלויות	חיוב לפי MAU של Guests מעבר לכמות חינם	חיוב לפי MAU של לקוחות (אחר)
ניהול הרשאות	משתמשים נכנסים כ־Guests → ניתן למפות אותם לקבוצות / תפקידים	מודל הרשאות שמוגדר ע"י האפליקציה
שימושים נפוצים	שיתוף Teams, SharePoint, PowerApps, SaaS	פורטלים, אתרי לקוחות, אפליקציות מובייל
Custom Branding	בסיסי בלבד	מתקדם מאוד — ממשק Login יכול להיות מותאם אישית
Policy Enforcement	Conditional Access + MFA + Identity Protection	Policies נפרדים ב-B2C (Custom Policies / User Flows)
פדרציה	נתמך (פדרציה בין ארגונים)	נתמך מול IdP צד ג' (Google/FB/ADFS)
ניהול מחזור חיים (Lifecycle)	מנוהל כמו User רגיל, אפשר לבצע Access Reviews	מנוהל ע"י האפליקציה / policies
Experience למשתמש	"Sign in with your organizational account"	"Sign up / Sign in" — כמו אתר לקוחות

נושא	Entra B2B (External Identities)	Entra B2C (Customer Identity)
מי שולט בסיסמא?	הארגון של המשתמש (IdP החיצוני)	האפליקציה/הארגון שלך (אם Local Accounts)
מתי בוחרים B2B?	כשאתה משתף תוכן או שירותים עם **ארגונים אחרים**	כשאתה מייצר **אפליקציה לציבור / לקוחות**
מתי בוחרים B2C?	לאפליקציות פנימיות/עסקיות → לא מתאים	לפורטלים, אתרי רישום, אפליקציות לקוח