Ανάπτυξη Κινητών Εφαρμογών

Νικόλαος Κορφιάτης & Δημήτρης Ρίγγας

nkorf@ionio.gr | riggas@ionio.gr

Τμήμα Πληροφορικής

Ιόνιο Πανεπιστήμιο

Περιεχόμενα (Contents)

Η Σημασία του Authentication

Τι είναι το Authentication;

Authentication (Πιστοποίηση) είναι η διαδικασία επαλήθευσης της ταυτότητας ενός χρήστη, συσκευής ή συστήματος.

Βασικές Έννοιες:

- **Authentication**: "Ποιος είσαι;" (Who are you?)
- **Authorization**: "Τι επιτρέπεται να κάνεις;" (What can you do?)
- **Identity**: Η ψηφιακή ταυτότητα του χρήστη

Παραδείγματα:

- Username/Password
- Biometric authentication (δακτυλικά αποτυπώματα, FaceID)
- Hardware tokens (YubiKey)
- One-Time Passwords (OTP)

Γιατί είναι κρίσιμο για Mobile Apps;

Προστασία Δεδομένων

- Προσωπικές πληροφορίες χρηστών
- Οικονομικά δεδομένα (πληρωμές, τραπεζικές συναλλαγές)
- Ευαίσθητα δεδομένα υγείας, επικοινωνίας

Εμπιστοσύνη Χρηστών

- Οι χρήστες πρέπει να εμπιστεύονται την εφαρμογή με τα δεδομένα τους
- Ένα security breach μπορεί να καταστρέψει τη φήμη μιας εφαρμογής
- GDPR και άλλοι νομικοί κανονισμοί απαιτούν ισχυρή προστασία

Απειλές & Κίνδυνοι

Συνηθισμένες Επιθέσεις:

1. **Credential Stuffing**: Χρήση κλεμμένων credentials από άλλα breaches
2. **Man-in-the-Middle (MITM)**: Υποκλοπή επικοινωνίας
3. **Phishing**: Εξαπάτηση χρηστών για κλοπή credentials
4. **Session Hijacking**: Κλοπή active sessions
5. **Brute Force**: Δοκιμή πολλών συνδυασμών passwords

Επιπλέον Ρίσκο στις Κινητές Συσκευές:

- Κλοπή/απώλεια συσκευής
- Insecure local storage
- Reverse engineering της εφαρμογής
- Malicious apps στην ίδια συσκευή

Αρχές Ασφάλειας στην Αυθεντικοποίηση Χρηστών

1. Defense in Depth
Πολλαπλά επίπεδα ασφάλειας, όχι μόνο ένα

2. Least Privilege
Ελάχιστα απαραίτητα δικαιώματα για κάθε χρήστη/διαδικασία

3. Never Trust, Always Verify
Έλεγχος σε κάθε request, όχι μόνο στο login

4. Secure by Default
Ασφαλείς ρυθμίσεις από την αρχή, όχι opt-in

5. Fail Securely
Σε περίπτωση σφάλματος, το σύστημα να μην εκθέτει δεδομένα

Παράγοντες αυθεντικοποίησης

Something You Know (Γνώση)

- Password, PIN, Security Questions

Something You Have (Κατοχή)

- Mobile device, Security token, Smart card

Something You Are (Βιομετρικά)

- Fingerprint, Face recognition, Iris scan, Voice

Multi-Factor Authentication (MFA)

Συνδυασμός 2 ή περισσότερων factors για αυξημένη ασφάλεια

- **2FA**: Δύο factors (π.χ. password + SMS code)
- **3FA**: Τρεις factors (σπάνιο σε mobile apps)

Token-Based Authentication:

Σύγχρονη προσέγγιση χωρίς sessions

Session-Based vs Token-Based Authentication

Traditional Session-Based:

```
[Client] ---> Login ---> [Server]
         <--- SessionID <---
         ---> Request + SessionID --->
         [Server έλεγχος στη μνήμη/DB]
         <--- Response <---
```

Μειονεκτήματα:

- Stateful (server πρέπει να κρατάει sessions)
- Δύσκολη κλιμάκωση (scaling)
- Προβλήματα σε distributed systems

Token-Based Authentication Flow

```
[Client] ---> Login (credentials) ---> [Server]
         <--- Access Token (user data + signature)<---
         ---> Request + Token ---> [Server]
         [Επαλήθευση token χωρίς state]
         <--- Response <---
```

Πλεονεκτήματα:

- **Stateless**: Ο server δεν κρατάει session state
- **Scalable**: Εύκολη κλιμάκωση σε πολλά servers
- **Cross-Domain**: Λειτουργεί σε διαφορετικά domains
- **Mobile-Friendly**: Ιδανικό για mobile apps

JWT - JSON Web Tokens

Δομή ενός JWT:

```
header.payload.signature
```

Παραδείγματα:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.
SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

JWT - JSON Web Tokens

Header (Base64 encoded):

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Payload (Base64 encoded):

```
{
  "sub": "1234567890",          // Subject (user ID)
  "name": "John Doe",
  "email": "john@example.com",
  "iat": 1516239022,            // Issued At
  "exp": 1516242622             // Expiration
}
```

Signature:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret
)
```

Σημαντικό: Το JWT είναι **signed** αλλά **όχι encrypted**!
Μην βάζετε ευαίσθητα δεδομένα στο payload.

JWT Standard Claims

Registered Claims (RFC 7519):

- **iss** (Issuer): Ποιος εξέδωσε το token
- **sub** (Subject): Για ποιον είναι το token (user ID)
- **aud** (Audience): Για ποιον προορίζεται
- **exp** (Expiration): Πότε λήγει (Unix timestamp)
- **nbf** (Not Before): Πότε αρχίζει να ισχύει
- **iat** (Issued At): Πότε εκδόθηκε
- **jti** (JWT ID): Μοναδικό identifier

Custom Claims:

Μπορείτε να προσθέσετε custom fields:

```
{
  "role": "admin",
  "permissions": ["read", "write"],
  "tenant_id": "company123"
}
```

Access Tokens vs Refresh Tokens

Access Token:

- **Σύντομη διάρκεια** (15 λεπτά - 1 ώρα)
- Χρησιμοποιείται για API requests
- Αν κλαπεί, περιορισμένη ζημιά λόγω expiration

Refresh Token:

- **Μακρά διάρκεια** (ημέρες - μήνες)
- Χρησιμοποιείται μόνο για να πάρει νέο access token
- Αποθηκεύεται secure (encrypted storage)
- Μπορεί να ανακληθεί (revoked) από τον server

Token Rotation:

Κάθε φορά που χρησιμοποιείται το refresh token, εκδίδεται νέο refresh token (για extra security).

Token Storage & Security

Που να αποθηκεύσουμε tokens στο mobile;

Όχι:

- Plain text files
- SharedPreferences/UserDefaults χωρίς encryption
- Application memory (μόνο)

Ναι:

- **iOS**: Keychain Services
- **Android**: EncryptedSharedPreferences, Keystore
- **React Native**: react-native-keychain, expo-secure-store

Βέλτιστες πρακτικές:

1. Encrypt refresh tokens
2. Access tokens μπορούν να είναι σε memory (short-lived)
3. Clear tokens on logout
4. Implement token expiration checks

Επικύρωση & Επαλήθευση Token

Server-Side Validation:

1. **Verify Signature**: Έλεγχος ότι το token δεν έχει τροποποιηθεί
2. **Check Expiration**: Έλεγχος `exp` claim
3. **Validate Issuer**: Έλεγχος `iss` claim
4. **Check Audience**: Έλεγχος `aud` claim
5. **Check Revocation**: Έλεγχος σε revocation list (optional)

Client-Side:

```
// Decode JWT (χωρίς verification)
const decoded = JSON.parse(atob(token.split('.')[1]));

// Έλεγχος expiration
if (decoded.exp * 1000 < Date.now()) {
  // Token expired, χρειάζεται refresh
}
```

Σημείωση: Επικύρωση (verification) κάνει μόνο ο server!

Authorization Framework: OAuth 2.0 & PKCE

Τι είναι το OAuth 2.0;

OAuth 2.0 είναι ένα **authorization protocol** (όχι authentication!).

Βασικό Πρόβλημα που Λύνει:
Πώς να δώσει ένας χρήστης σε μια εφαρμογή πρόσβαση στα δεδομένα του, χωρίς να μοιραστεί το password του;

Παράδειγμα:

- Θέλω η mobile app μου να ανεβάσει photos στο Google Drive
- Δεν θέλω να δώσω το Google password μου στην app
- Το OAuth επιτρέπει στην app να πάρει περιορισμένη πρόσβαση

Χαρακτηριστικά:

- Delegated authorization
- Scoped permissions
- Token-based

OAuth 2.0 Roles

1. Resource Owner (Χρήστης)
Ο ιδιοκτήτης των δεδομένων

2. Client (Mobile App)
Η εφαρμογή που ζητάει πρόσβαση

3. Authorization Server
Εκδίδει tokens μετά από επιτυχή πιστοποίηση
(π.χ. Google, Facebook, Auth0)

4. Resource Server
Φιλοξενεί τα προστατευμένα δεδομένα
(π.χ. Google Drive API, Facebook Graph API)

Σημείωση: Authorization Server και Resource Server μπορεί να είναι το ίδιο σύστημα.

Authorization Code Flow (Βασικό)

Πρόβλημα χωρίς PKCE (Proof Key for Code Exchange):
Malicious apps μπορούν να υποκλέψουν το authorization code.

Λύση - PKCE (RFC 7636):

1. App δημιουργεί **code_verifier** (random string)
2. Υπολογίζει **code_challenge** = SHA256(code_verifier)
3. Στέλνει code_challenge στον authorization request
4. Όταν κάνει exchange του code, στέλνει τον code_verifier
5. Server επαληθεύει: SHA256(code_verifier) == code_challenge

→ Μόνο το app που ξεκίνησε το flow μπορεί να ολοκληρώσει το exchange!

OAuth Scopes

Τι είναι τα Scopes;

Καθορίζουν τι δικαιώματα ζητάει η εφαρμογή.

Παραδείγματα:

```
scope=openid profile email          // OIDC basic
scope=read:calendar write:calendar  // Google Calendar
scope=user:email repo               // GitHub
scope=photos.read photos.write      // Custom API
```

Best Practices:

- Ζητήστε **μόνο** τα scopes που χρειάζεστε
- Εξηγήστε στον χρήστη γιατί τα χρειάζεστε
- Μπορείτε να ζητήσετε επιπλέον scopes αργότερα (incremental authorization)

Identity Layer πάνω στο OAuth: OpenID Connect (OIDC)

OAuth vs OpenID Connect

OAuth 2.0:

- **Authorization** protocol
- "Μπορεί το app να διαβάσει τα emails μου;"
- Δίνει access tokens για API access

OpenID Connect (OIDC):

- **Authentication** protocol
- "Ποιος είναι αυτός ο χρήστης;"
- Επέκταση του OAuth 2.0
- Δίνει **ID tokens** + access tokens

OIDC = OAuth 2.0 + Identity Layer

ID Token (JWT)

Τι είναι το ID Token;

Ένα JWT που περιέχει πληροφορίες ταυτότητας χρήστη.

```
{
  "iss": "https://accounts.google.com",
  "sub": "110169484474386276334",
  "aud": "your-client-id",
  "exp": 1516239022,
  "iat": 1516235422,
  "name": "John Doe",
  "email": "john@example.com",
  "email_verified": true,
  "picture": "https://lh3.googleusercontent.com/..."
}
```

Διαφορά από Access Token:

- ID token για την **ταυτότητα** του χρήστη
- Access token για **πρόσβαση σε APIs**

Social Login Providers: Google, Facebook, Apple, GitHub κ.ά.

Γιατί Social Login;

Πλεονεκτήματα για Users:

- Δεν χρειάζεται νέο password
- Γρήγορη εγγραφή (one-click)
- Trusted providers (Google, Apple, etc.)

Πλεονεκτήματα για Developers:

- Μειωμένη ανάπτυξη (δεν φτιάχνεις auth system)
- Καλύτερη ασφάλεια (providers έχουν security teams)
- Built-in 2FA από providers
- Profile data (name, email, photo)

Μειονεκτήματα:

- Εξάρτηση από third-party services
- Privacy concerns
- Terms of Service limitations

Cloud Authentication Services: Firebase, AWS Cognito, Supabase, Auth0

Τι είναι τα Cloud Auth Services;

Managed Authentication Solutions:
Πλήρεις authentication/authorization υπηρεσίες ως cloud services.

Τι προσφέρουν:

- User registration & login
- Social providers integration
- Multi-factor authentication (MFA)
- Password reset flows
- Session management
- User management dashboard
- SDKs για mobile/web

Πλεονεκτήματα:

- Γρήγορη ανάπτυξη
- Proven security
- Scalability
- Compliance (GDPR, SOC2)

Firebase Authentication

Χαρακτηριστικά:

- Google product (free tier available)
- Easy integration με άλλα Firebase services
- Realtime Database, Firestore, Cloud Functions

Supported Providers:

- Email/Password
- Phone (SMS)
- Google, Facebook, Apple, Twitter, GitHub, Microsoft
- Anonymous authentication
- Custom authentication (με tokens)

Pricing:

- Free: Unlimited users
- Επιπλέον χρεώσεις για Phone Auth (SMS costs)

Firebase Auth - Implementation

```
// React Native + Firebase
import auth from '@react-native-firebase/auth';

// Email/Password Sign Up
const signUp = async (email, password) => {
  try {
    const userCredential = await auth()
      .createUserWithEmailAndPassword(email, password);

    console.log('User created:', userCredential.user.uid);
  } catch (error) {
    console.error(error);
  }
};

// Google Sign In
import { GoogleSignin } from '@react-native-google-signin/google-signin';

const signInWithGoogle = async () => {
  const { idToken } = await GoogleSignin.signIn();
  const googleCredential = auth.GoogleAuthProvider.credential(idToken);
  return auth().signInWithCredential(googleCredential);
};
```

Firebase Auth Features

Email Verification:

```
const user = auth().currentUser;
await user.sendEmailVerification();
```

Password Reset:

```
await auth().sendPasswordResetEmail(email);
```

Phone Authentication:

```
const confirmation = await auth().signInWithPhoneNumber(phoneNumber);
const credential = await confirmation.confirm(code);
```

Custom Claims (για authorization):

```
// Backend - Admin SDK
admin.auth().setCustomUserClaims(uid, {
  role: 'admin',
  permissions: ['read', 'write']
});
```

Secure Storage σε Mobile: Αποθήκευση Credentials & Tokens

Storage Security Challenges

Απειλές στο Mobile:

1. **Device Theft/Loss**: Φυσική πρόσβαση στη συσκευή
2. **Malware**: Κακόβουλες εφαρμογές στην ίδια συσκευή
3. **Rooted/Jailbroken Devices**: Κατάργηση security restrictions
4. **Backup Exposure**: Unencrypted backups (iTunes, iCloud, Google)
5. **Memory Dumps**: Forensic analysis
6. **Reverse Engineering**: Εξαγωγή κλειδιών από το app binary

Τι πρέπει να προστατεύσουμε:

- Access tokens (short-lived, αλλά ευαίσθητα)
- Refresh tokens (long-lived, πολύ κρίσιμα!)
- User credentials (ποτέ δεν πρέπει να αποθηκεύονται!)
- API keys, secrets

iOS - Keychain Services

Τι είναι το Keychain;

Encrypted storage από το iOS για ευαίσθητα δεδομένα.

Χαρακτηριστικά:

- Hardware-backed encryption (Secure Enclave)
- Protected με device passcode
- Isolated per-app (by default)
- Μπορεί να sync μεταξύ devices (iCloud Keychain)
- Survives app uninstall (optional)

Android - Keystore System

Android Keystore:

Hardware-backed key storage (σε devices με Trusted Execution Environment).

Χαρακτηριστικά:

- Keys δεν μπορούν να extracted από το device
- Crypto operations στο hardware
- User authentication required (biometric/PIN)
- Key invalidation on security events

Πηγές & Resources

Documentation:

- OAuth 2.0 RFC 6749
- PKCE RFC 7636
- OpenID Connect Spec

Tools & Libraries:

- jwt.io - JWT debugger
- Firebase Docs

Learning:

- OWASP Mobile Security
- OWASP Authentication Cheat Sheet