

Securitatea și confidențialitatea datelor în contextul aplicațiilor mobile

Ghimpu Lucian Eduard

April 7, 2019

CONDUCĂTOR ȘTIINȚIFIC
DR. CZIBULA ISTVAN, PROFESOR UNIVERSITAR

Contents

1	Introducere	3
1.1	Introducere	3
1.2	Ecosistemul aplicațiilor mobile	3
1.3	GDPR	3
2	Securitatea și confidențialitatea datelor în contextul aplicațiilor mobile	3
2.1	Autentificarea si înregistrarea	3
2.1.1	Ceva introducere *	4
2.1.2	JWT si OAuth2	5
2.1.3	Autentificare prin senzori biometrici	7
2.1.4	Autentificare prin factori multipli	7
2.2	Comunicarea cu serverul si servicii (networking) *	7
2.2.1	HTTP și HTTPS	7
2.2.2	Alte canale de comunicare (SMS)	7
2.2.3	Web sockets	7
2.3	Persistența datelor	7
2.3.1	Metode de persistare a datelor	7
2.3.2	Criptografie	7
2.3.3	Gestionarea datelor sensibile	7
2.4	Alți factori	7
2.4.1	Permiuni	7
2.4.2	Webviews	7
2.4.3	Distribuirea aplicației	7
2.4.4	Probleme specifice pe anumite platforme	7
3	Medicarium	7
3.1	Analiza aplicației	7
3.1.1	Problematica	7
3.1.2	Cazuri de utilizare	7
3.2	Proiectarea aplicației	7
3.2.1	Arhitectura	7
3.2.2	UML ceva???	7
3.3	Implementarea aplicației - Serverul și serviciile	7
3.3.1	Server REST	7
3.3.2	Node.js	7
3.3.3	MongoDB	7
3.3.4	Rute disponibile	7
3.3.5	Autentificare in doi pași	7
3.4	Implementarea aplicației - Clientul mobil	7
3.4.1	Android Jetpack	7
3.4.2	Kotlin	7
3.4.3	Autentificarea	7
3.4.4	Securitatea aplicației	7
3.4.5	Gestionarea permiunilor	7
3.4.6	Gestionarea fisierelor	7
3.5	Testarea	7

4	Manual de utilizare	7
5	Concluzii	7
	List of Figures	8
	References	8
1	Introducere	
1.1	Introducere	
1.2	Ecosistemul aplicațiilor mobile	
1.3	GDPR	
2	Securitatea și confidențialitatea datelor în contextul aplicațiilor mobile	
2.1	Autentificarea si înregistrarea	

2.1.1 Ceva introducere *

Indiferent ca vorbim de aplicatii web, desktop sau mobile, majoritatea folosesc o metoda de autentificare. Autentificarea și înregistrarea stau la baza problematicei securitatii datelor. In clasamentul OWASP Top 10 din 2017 [1], problemele legate de autentificare si gestiunea sesiunii, sunt clasate pe locul 2. Iar in clasamentul OWASP top 10 Mobile din 2016 [2], autentificare nesigura si autorizarea necorespunzatoare sunt clasate pe locul 4, respectiv 6.

Unicitatea aplicatiilor mobile este data de faptul ca un dispozitiv mobil poate devenii accesibil oricarui persoane datorita portabilitatii lor. Un dispozitiv mobil poate fi furat, pierdut sau accesat temporal de o persoana necunoscuta fara permisiunea posesorului. Prin urmare nevoia de un sistem de autentificare robust este mandatorie atunci cand vorbim de aplicatii care gestioneaza date sensibile (aplicatii financiare, sociale, medicale, etc...).

In cadrul aplicatiilor mobile, autentificarea se poate face prin mai multe metode. De la simpla autentificare prin utilizator si parola, pana la utilizarea de senzori biometrici. Mitigari clasice precum impunerea unei parole sigure raman valabile si in contextul aplicatiilor mobile.

Pentru alegerea metodei de autentificare trebuie sa ne punem in prima instanta urmatoarele doua intrebari:

- Care este scopul aplicatiei? O aplicatie care notifica utilizatorul despre starea meteo poate ca nu ar avea nevoie de autentificare prin senzori biometrici.
- Aplicatia gestioneaza date confidentiale? Un exemplu potrivit ar fi o aplicatie precum BT pay, care gestioneaza contul curent al unui utilizatorului se foloseste de mai multe metode de autentificare, o data prin datele de logare, iar apoi prin senzori biometrici.

Dupa ce ne este clar care este scopul aplicatiei si cu ce fel de date lucreaza, putem include una sau mai multe metode de autentificare bazate dupa urmatoarii factori:

1. Ceva ce utilizatorul stie (parola, pin etc...)
2. Ceva ce il defineste pe utilizator (amprenta, retina)
3. Ceva ce utilizatorul detine (parole generate temporal)



Figure 1: Metode de autentificare [3]

2.1.2 JWT si OAuth2

Majoritatea aplicatiilor de azi se folosesc de cea mai simpla forma de autentificare, prin folosirea de credentiale (ceva ce utilizatorul stie) si unui token de acces (ceva ce utilizatorul detine). Utilizatorul isi creeza cont pentru o anumita aplicatie, foloseste credentialele pentru a se loga, cererea de autentificare ajunge la server unde se verifica credentialele iar apoi se genereaza un token care va fi folosit de utilizator pentru a accesa diferite resurse in aplicatie.

Scenariu descris anterior se refera la folosirea de JWT (JSON Web Token), un standard (RFC 7519) [4] adoptat de multe aplicatii mobile in zilele noastre. JSON Web Token este o metoda sigura de autorizare a transferului de informatii intre doua parti [5], deobicei clientul mobil si serverul la care se face cererea. Clientul revendica de la server o dovada, un token, care apoi este folosit de client pentru a accesa diferite resurse.

Din punct de vedere tehnic, un JWT are urmatoarea forma 11111.22222.33333 si este alcatuit din 3 parti:

1. Antetul (Header)
2. Datele utile (Payload)
3. Semnatura (Signature)

HEADER: ALGORITHM & TOKEN TYPE

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

Figure 2: Antetul unui JWT, alcatuit din tipul de algoritm de inregistrare (HS256) si tipul de token (JWT).

PAYLOAD: DATA

```
{  
  "numar": "1234567890",  
  "nume": "John Doe",  
  "admin": false  
}
```

Figure 3: Partea utila al unui JWT contine date sau permisiuni pe care clientul le are.

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
)
```

Figure 4: Semnatura unui JWT este alcătuită din antetul encodat, datele encodate, algoritmul folosit în antet și un secret. Semnatura are rolul de a oferi o metoda de verificare pentru a asigura că conținutul nu a fost modificat

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJudW1hcnI6IjEyMzQ1Njc4OTAiLCJudW1lIjoieSm9obiBEb2UiLCJkYXRhIjoxNTE2MjM5MDIyfQ.uEE41yzcc12Z1Tp0J20NwLbg_js4sMq6ikJRue87QUo
```

Figure 5: JWT va fi format în final de 3 siruri de caractere de tip Base64-URL separate prin punct.

O dată ce clientul detine un token, acesta trebuie tratat cu multă grijă în cadrul unei aplicații. Acesta poate fi folosit mai apoi în antetul tuturor cererilor de tip HTTP sub forma `Authorization: Bearer {token}`. Din acest motiv cel mai probabil se dorește salvarea token-ului în memoria locală a dispozitivului mobil pentru a putea fi apoi folosit în viitor. Acesta poate fi criptat iar la rândul lui la nivelul clientului, deși în mod nativ atât pe android cât și pe ios există metode sigure de stocare a date de tip token (SharedPreferences în mod private pe Android și keychain pe iOS).

Pentru un nivel și mai mare de siguranță, se poate limita durata de timp pe care este valabil un token. Spre exemplu aplicația BT Pay folosește un token care este valid timp de 10-15 minute. După ce token-ul expiră, utilizatorul este nevoit să se autentifice din nou.

Avantajul principal pe care îl oferă JWT este facilitatea prin care se demarează tot procesul prin care un client își revendică datele sau drepturile de la un server. Un alt aspect important îl reprezintă faptul că în spate, totul se produce folosit obiecte de tipul JSON, obiecte care sunt extrem de răspândite în orice limbaj de programare și în tot internetul.

- 2.1.3 Autentificare prin senzori biometrici
- 2.1.4 Autentificare prin factori multipli
- 2.2 Comunicarea cu serverul si servicii (networking) *
- 2.2.1 HTTP și HTTPS
- 2.2.2 Alte canale de comunicare (SMS)
- 2.2.3 Web sockets
- 2.3 Persistența datelor
- 2.3.1 Metode de persistare a datelor
- 2.3.2 Criptografie
- 2.3.3 Gestionarea datelor sensibile
- 2.4 Alți factori
- 2.4.1 Permisuni
- 2.4.2 Webviews
- 2.4.3 Distribuirea aplicației
- 2.4.4 Probleme specifice pe anumite platforme

3 Medicarium

- 3.1 Analiza aplicației
- 3.1.1 Problematika
- 3.1.2 Cazuri de utilizare
- 3.2 Proiectarea aplicației
- 3.2.1 Arhitectura
- 3.2.2 UML ceva???
- 3.3 Implementarea aplicației - Serverul și serviciile
- 3.3.1 Server REST
- 3.3.2 Node.js
- 3.3.3 MongoDB
- 3.3.4 Rute disponibile
- 3.3.5 Autentificare in doi pași
- 3.4 Implementarea aplicației - Clientul mobil
- 3.4.1 Android Jetpack
- 3.4.2 Kotlin
- 3.4.3 Autentificarea
- 3.4.4 Securitatea aplicației
- 3.4.5 Gestionarea permisiunilor
- 3.4.6 Gestionarea fisierelor
- 3.5 Testarea

4 Manual de utilizare

List of Figures

1	Metode de autentificare [3]	4
2	Antetul unui JWT, alcatuit din tipul de algoritm de inregistrare (HS256) si tipul de token (JWT).	5
3	Partea utila al unui JWT contine date sau permisiuni pe care clientul le are.	5
4	Semnatura unui JWT este alcatuita din antetul encodat, datele encodeate, algoritmul folosit in antet si un secret. Semnatura are rolul de a oferi o metoda de verificare pentru a asigura ca continutul nu a fost modificat	6
5	JWT va fi format in final de 3 siruri de caractere de tip Base64-URL separate prin punct.	6

References

- [1] OWASP Top 10 2017
https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
- [2] OWASP Top 10 Mobile 2016
https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10
- [3] 3 types of Authentication
<https://www.slideshare.net/awesomeadmin/secure-your-salesforce-org-with-twofactor-authentication>
- [4] RFC 7519 JWT
<https://tools.ietf.org/html/rfc7519>
- [5] JSON Web Token (JWT)
<https://tools.ietf.org/html/rfc7519>
- [6] ENISA, Privacy and data protection in mobile applications, 29.01.2019
<https://www.enisa.europa.eu/publications/privacy-and-data-protection-in-mobile-applications>
- [7] ENISA, Smartphone Secure Development Guidelines, 27.02.2017,
<https://www.enisa.europa.eu/publications/smartphone-secure-development-guidelines-2016>
- [8] OWASP Mobile Application Security Verification Standard,
<https://github.com/OWASP/owasp-masvs>