✍️ חתימה דיגיטלית — Digital Signature

CCNA CCNP Hash Functions · Asymmetric Encryption · RSA / DSA / ECDSA · Non-Repudiation · HMAC

📖 מושגי יסוד

חתימה דיגיטלית: ערך מוצפן שמאמת את זהות השולח ואת שלמות ההודעה.

Hash (גיבוב): פונקציה חד-כיוונית שיוצרת סיכום קצר ייחודי לכל קובץ/הודעה.

מפתח פרטי (Private Key): מוחזק בסוד ע"י השולח — משמש ל**חתימה**.

מפתח ציבורי (Public Key): ידוע לכולם — משמש ל**אימות** החתימה.

Non-Repudiation (אי-הכחשה): השולח לא יכול לטעון שלא שלח.

Integrity (שלמות): ההודעה לא שונתה בדרך.

Authentication (אימות): אישור זהות השולח.

HMAC: Hash + מפתח סודי משותף — לאימות ללא PKI.

🎯 מה חתימה דיגיטלית מספקת?

מאפיין	הסבר
✅ **אימות זהות**	רק מי שמחזיק ב-Private Key יכול ליצור את החתימה
✅ **שלמות**	שינוי כלשהו בנתונים → Hash שונה → חתימה לא תואמת
✅ **אי-הכחשה**	השולח לא יכול להכחיש — רק הוא מחזיק ב-Private Key
❌ **סודיות**	חתימה דיגיטלית לא מצפינה את התוכן! נדרשת הצפנה נפרדת

🚨 **שאלת בחינה נפוצה:** חתימה דיגיטלית **לא** מספקת סודיות (Confidentiality)!

🖊️ תהליך יצירת החתימה (צד השולח)

1 חישוב Hash
הרצת ההודעה דרך SHA-256
↑ שולח

←

2 הצפנת Hash
עם **Private Key** של השולח
↑ שולח

←

3 שליחה
הודעה מקורית + חתימה מוצפנת
↑ שולח

←

4 פענוח Hash
עם **Public Key** של השולח
↑ מקבל

←

5 השוואה
Hash מחושב = Hash מפוענח?
↑ מקבל

ℹ️ **כלל אצבע:** חותמים עם **Private Key** (של השולח) ← מאמתים עם **Public Key** (של השולח). זה הפוך מהצפנה רגילה!

⚖️ חתימה דיגיטלית VS. HMAC

	חתימה דיגיטלית	HMAC
מפתחות	Asymmetric (PKI)	Symmetric (מפתח משותף)
Non-Repudiation	✓ כן	✗ לא
נדרש CA?	✓ כן	✗ לא
שימוש נפוץ	TLS, S/MIME, Code Signing	IPsec, OSPF Auth, RADIUS
מהירות	איטי יותר	מהיר יותר

⚙️ אלגוריתמי חתימה דיגיטלית

אלגוריתם	אורך מפתח	הערה
RSA	512–4096 bit	הנפוץ ביותר. חתימה + הצפנה
DSA (DSS)	512–1024 bit	מהיר ביצירה, 10–40x איטי באימות vs RSA
ECDSA	160–521 bit	מפתח קטן, אבטחה גבוהה. נפוץ ב-TLS 1.3
ElGamal	512–1024 bit	בסיס DH. הודעה מוצפנת = כפול גודל מקור

🔢 פונקציות HASH נפוצות

פונקציה	גודל Output	סטטוס
MD5	128 bit	מיושן — לא מאובטח
SHA-1	160 bit	Legacy — Cisco לא ממליץ
SHA-256	256 bit	✓ מומלץ
SHA-384/512	384/512 bit	✓ CCNP / Next-Gen

🔑 הצפנה VS. חתימה — מה ההבדל?

הצפנה (Encryption)
מצפינים עם **Public Key** של הנמען
מפענחים עם **Private Key** של הנמען
מטרה: סודיות

חתימה (Signature)
חותמים עם **Private Key** של השולח
מאמתים עם **Public Key** של השולח
מטרה: אימות + שלמות

⚠️ **שאלה נפוצה:** הצפנה ← Public Key של נמען.
חתימה ← Private Key של שולח!

🌐 שימושים נפוצים

פרוטוקול/שימוש	אלגוריתם	רמה
TLS/HTTPS	RSA / ECDSA	CCNA
IPsec VPN (RSA-Sig)	RSA + SHA	CCNA
SSH Host Keys	RSA / ECDSA	CCNA
S/MIME (Email)	RSA / DSA	CCNA
Code Signing	RSA / ECDSA	CCNA
OSPF Authentication	HMAC-MD5/SHA	CCNA
DNSSEC	RSA / ECDSA	CCNP
BGP Authentication	MD5 / SHA HMAC	CCNP
IKEv2 (ECDSA)	ECDSA + SHA-256	CCNP

🔒 RSA AUTHENTICATION ב-IPSEC (IOS)

תהליך RSA ב-**Hash**: IPsec → **הצפנה עם Private Key** → **שליחה עם Digital Certificate** → **אימות עם Public Key**

```
! יצירת RSA Key Pair R1(config)# crypto key generate rsa modulus 2048 ! הגדרת Trustpoint (CA) R1(config)# crypto pki trustpoint MY-CA R1(ca-trust)# enrollment url http://192.168.1.100 R1(ca-trust)# rsakeypair MY-RSA-KEY ! IPsec עם RSA R1(config)# crypto isakmp policy 10 R1(isakmp)# authentication rsa-sig R1(isakmp)# hash sha256 R1(isakmp)# encryption aes 256 ! בדיקה R1# show crypto key mypubkey rsa R1# show crypto pki certificates R1# show crypto isakmp sa
```

שאלות CCNA / CCNP — חתימה דיגיטלית

CCNA רמת CCNA | CCNP רמת CCNP | שאלה | תשובה | הסבר

1 CCNA באיזה מפתח משתמשים ליצירת חתימה דיגיטלית?

ב-Private Key של השולח.

לאימות החתימה משתמשים ב-Public Key של השולח. זה הפוך מהצפנה!

2 CCNA מה חתימה דיגיטלית לא מספקת?

סודיות (Confidentiality) — החתימה לא מצפינה את ההודעה.

לסודיות נדרש להצפין את ההודעה בנפרד עם Public Key של הנמען.

3 CCNA מה הצעד הראשון בתהליך חתימה דיגיטלית?

חישוב Hash של ההודעה (לדוגמה עם SHA-256).

סדר: Hash → הצפנת Hash עם Private Key → שליחת הודעה + חתימה.

4 CCNA מה ההבדל בין HMAC לחתימה דיגיטלית?

HMAC משתמש במפתח סודי משותף (Symmetric). חתימה — ב-PKI (Asymmetric).

HMAC לא מספק Non-Repudiation כי שני הצדדים מחזיקים באותו מפתח.

5 CCNA מה הפקודה ליצירת RSA Key ב-IOS?

`crypto key generate rsa modulus 2048`

מפתח של 2048 bit מומלץ. בדיקה: `show crypto key mypubkey rsa`

6 CCNA מה Non-Repudiation ולמה חתימה דיגיטלית מספקת אותו?

אי-הכחשה — השולח לא יכול לטעון שלא שלח, כי רק הוא מחזיק ב-Private Key.

HMAC לא מספק זאת כי גם הנמען מחזיק במפתח המשותף.

7 CCNA מה DSA ואיך הוא שונה מ-RSA לחתימות?

DSA (Digital Signature Algorithm) — מהירות חתימה דומה ל-RSA, אך אימות 10–40x איטי יותר.

DSA מוגדר ב-DSS (Digital Signature Standard). RSA עדיף לרוב השימושים.

8 CCNA למה SHA-1 נחשב Legacy ב-Cisco?

Cisco מגדיר SHA-1 כ-Legacy וממליץ לפחות על SHA-256 לצורך integrity.

MD5 אינו מאובטח לחלוטין — נפגע ל-collision attacks. SHA-256 ומעלה מומלץ.

9 CCNP מה היתרון של ECDSA על פני RSA?

מפתח קצר יותר (160 bit לעומת 2048 bit) עם אבטחה מקבילה — יעיל יותר.

ECDSA (Elliptic Curve DSA) נפוץ ב-TLS 1.3, IKEv2. ECC מבוסס על עקומות אליפטיות.

10 CCNP מה ההבדל בין IKEv1 ל-IKEv2 בהקשר של חתימות?

IKEv2 תומך ב-ECDSA ו-IKEv1. SHA-256/384 הסתמך בעיקר על RSA ו-SHA-1.

IKEv2 גם יעיל יותר — 2 exchanges במקום 6 ב-IKEv1. מומלץ לכל deployment חדש.

11 CCNP כיצד RSA Authentication עובד ב-IPsec?

שני הצדדים מחשבים Hash, מצפינים עם Private Key שלהם, ושולחים עם Digital Certificate.

הנמען מפענח עם Public Key של השולח (מהאישור), ומשווה ל-Hash שחישב בעצמו.

12 CCNP מה הפקודה לאימות RSA Auth ב-IPsec IOS?

`crypto isakmp policy` תחת `authentication rsa-sig`

בדיקה: `debug crypto isakmp` | `show crypto isakmp sa` לבעיות אימות

🔑 טבלת סיכום מהיר לבחינה

מושג	עיקר הדבר	נקודה לבחינה
חתימה דיגיטלית	Hash מוצפן עם Private Key	מספקת: אימות, שלמות, אי-הכחשה. **לא סודיות!**
Private Key	שמור אצל השולח בלבד	משמש **לחתימה** — לא לאימות!
Public Key	ידוע לכולם	משמש **לאימות** החתימה — לא ליצירתה!
SHA-256	פונקציית Hash מומלצת	SHA-256 = מומלץ. MD5 = שבור. SHA-1 = Legacy
RSA vs ECDSA	שני אלגוריתמי חתימה מרכזיים	ECDSA = מפתח קצר + יעיל. RSA = נפוץ ותואם לאחור
HMAC	Hash + מפתח סודי משותף	אין Non-Repudiation. אין צורך ב-CA
rsa-sig ב-IOS	authentication rsa-sig	דורש Trustpoint + crypto pki enroll
IKEv2	תומך ECDSA + SHA-256	עדיף על IKEv1 — פחות exchanges, אלגוריתמים חדישים