

WEP → BROKEN

integrity $\xrightarrow{?}$ CRC lineare
 authentication $\xrightarrow{?}$ authenticator non sceglie IV
 confidentiality $\xrightarrow{?}$ IV piccolo, se si ripete trovo M

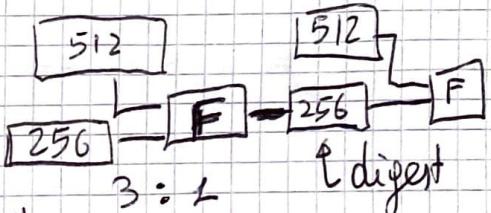
INTEGRITY \doteq msg authentication \doteq msg non alterato

Vernam Non lo garantisce.
 simm. Key: genera TAG = F(K, msg), irreversibile
 OK MITM, SPOOFING, NO REPLY \rightsquigarrow uso nonces
 MAC vs DS: DS più forte, ha "non ripudio"
 HASH?
 NON INVERTIBILE
 data X, trovare X': H(X) = H(X')
 è difficile (weak)
 trovare X₁, X₂: H(X₁) = H(X₂)
 (tra tutti i msg) (strong)

BIRTHDAY
 23 persone: stesso compleanno 6%. (weak) rispetto a me.
 23 persone: stessi compleanni tra due persone: 50% (strong)

\hookrightarrow n-bit digest, security level è $2^{n/2}$: dig: 24 bit, 50% coll $\rightsquigarrow 2^{24/2}$
 \hookrightarrow 40 persone > 23, stessi compleanni? > 90%
 \hookrightarrow Key₂ ha len(Key) + 22, quanto è più sicura? $2^{10} \cdot 2^{10} \cdot 2^2 = 2^{22}$

HASH: msg diviso in chunks, 512 bit
 (Merkle-Damgaard) complemento: N blocchi



dove metto Secret?
 INIZIO:
 NO, msg extension attack

FINE: faccio BRUTE force su blocco finale,
 complemento: 1 (precomputation attack)

HMAC : una $\text{HMAC}_K(M) = H \left[\begin{array}{c} K^+ \oplus \text{opad} \\ \uparrow \\ \text{estendo a} \\ 512 \text{ bit} \end{array} \parallel H(K^+ \oplus \text{ipad} \parallel M) \right]$

- esso sempre IV da mettere in F con $K^+ \oplus \text{ipad}$ e $K^+ \oplus \text{opad}$

- complementa' : $N+2$, $K \oplus \text{opad}$ inner hash + pad

↳ non espando

- basta mix pseudorandom, collision non necessaria !!
- Secret UNICO, pad diversi.

USER
AUTH

- pw : low-entropy, ho overload (riuso), charset limitato
 ↓ misurabile!

$$H(X) = -\sum p_i \log_2 p_i \equiv \underline{\log_2 (\text{n° possibilità})} !!!$$

Esempio : moneta, due risultati : prob: $\frac{1}{2}$

$$H(X) = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} = 1 \text{ bit}$$

Esempio : dado, 6 risultati, prob: $\frac{1}{6}$

$$H(X) = -\frac{1}{6} \log_2 \frac{1}{6} - \frac{1}{6} \log_2 \frac{1}{6} - \dots = -6 \log_2 \frac{1}{6} = 2,58 \text{ bit}, \text{ tra } 2 \text{ e } 3$$

$2^2 \text{ altern. } 2^3 \text{ altern.}$

Esempio : dato formato 1DD1MM $\rightarrow H(X) = \log_2 (366) \approx 8 \text{ bit}$

- pin : 3 elementi, 10 valori $\approx 10^3 \cdot \left(-\frac{1}{10^3} \log_2 \left(\frac{1}{10^3} \right) \right) \approx 13 \text{ bit}$

il secondo ha 5 bit $\rightarrow 2^5 = 32$ volte più sicuro.

- Se $H(X) = b < 1 \Rightarrow$ trasmetto b bit di info A ~~info~~, bit, non posso comprimere più di $(1-b) \%$

DICTONARY
ATTACK

: dato BD di partenza + pw generator, genera nuove pw

PAP

VS

CHAP

- pw in chiaro quando trasmessa,
 - posso memorizzare nel DB

- autenticazione solo all'inizio
(decide peer)
 - inizia peer, è "one time password"
 - sicuro ne attacco DB,
insicuro ne attacco comunicazione

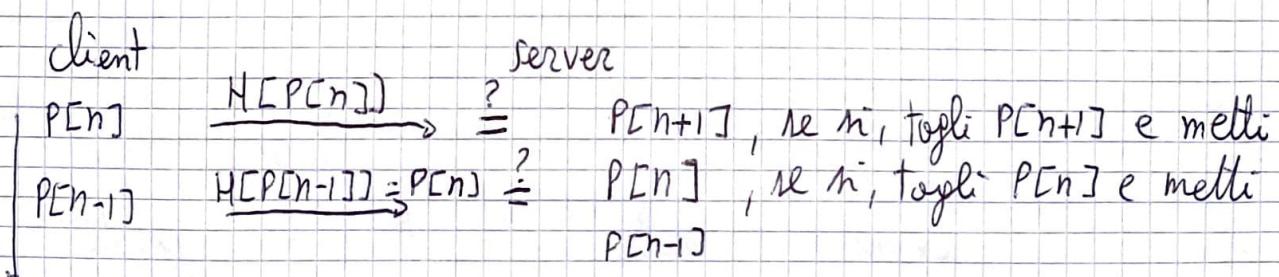
Se tentativi illuminati non uso hash "SHA 256," e' veloce.

- non voglio OTP → HASH CHAIN:

dato "N", e un valore iniziale, client genera le memorizzate

$N+1$ valori hash : $P[0]$, $P[1] = P[P[0]]$, $P[2] = P[P[P[0]]]$...

Never memorize SOLO $P[N+1]$. Funziona così:



- Server ha sempre $P(i+1)$ rispetto client!!
 - Server memorizza solo pw utente, client vulnerabile,
 - Compatto come PAP
 - P[D] lo manda server a client "offline"

OTP 2 factor

- HOTP : no chain, si counter
- TOTP : mo time, non counter (lo converto)

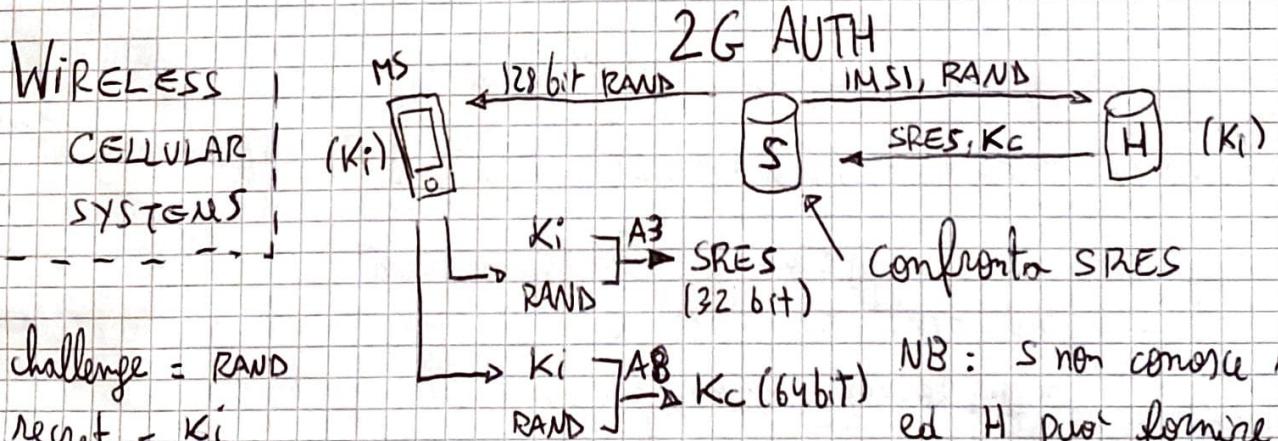
CHAP MUTUAL AUTHENTICATION : una stessa shared secret per le 2 autenticazioni
MALE! REFLECTION ATTACK :

Se server manda challenge A, posso reinviare a lui challenge A, (RIVUSO) mi risponde, e quindi uso tale risposta nella vecchia autenticazione
PERCONE' AUTENTICAZIONE UNILATERALE. DEVO LEGARLE. Come?

Flavia, NCA, H(secret, attuale Challenge B, Nuovo Challenge A) ($A \rightarrow B$)

Boss, H(secret, nuovo Challenge A, attuale Challenge B) ($B \rightarrow A$)

il problema era che con troppi C_i , attacker li faceva
 e negavano ad A o B. Con invece rimane bloccato



- challenge = RAND
 - secret = K_i
 - hash = A3
 - A3 proprietario, com 128 mo, mantenuto segreto, rotto appena rivelato.
 - No mutual auth
- NB: S non conosce A3, A8 ed H può fornire anche RAND (sicurezza in più)

3G → no mutual authentication!

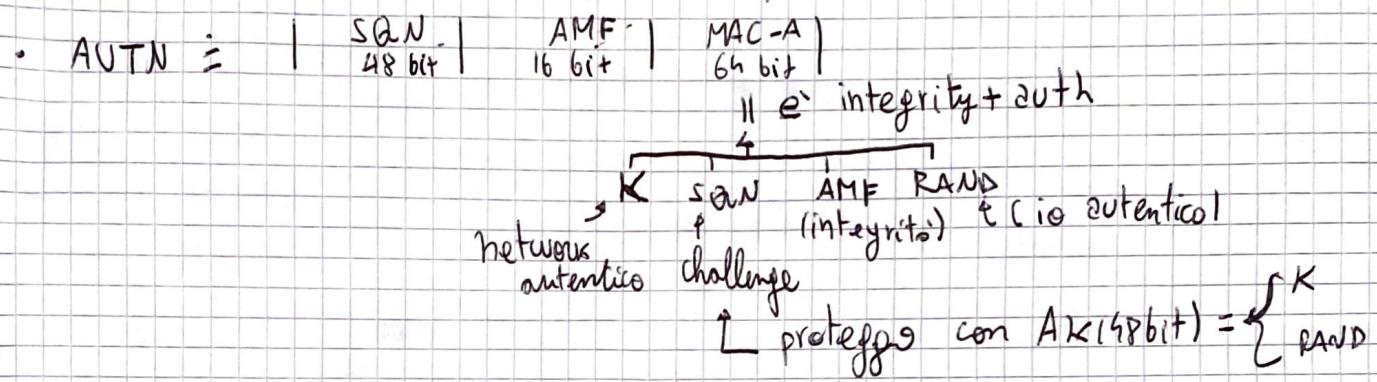
→ H manda a S (= VLR) AUTH vector [XRES, RAND, AUTN, CK, IK]

↗ token

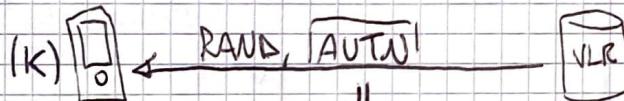
- MS riceve RAND, AUTN, ottiene RES, $\doteq XRES$?
- MS, dato K e 'RAND', genera $\begin{cases} \text{RES} \\ \text{CK} \\ \text{IK} \end{cases}$ (112 bit tutti) (alg. pubblici diversi)
- Network authentication: MS NONCE, devo "sincronizzare", entro ora la mia challenge.

→ setup

→ verifica autenticazione network



CONCLUSIONE



||

(seq \oplus AK) AMF, MAC_K (seq, AMF, RAND)

→ (RAND, K) → AK → (seq \oplus AK) \oplus AK = seq

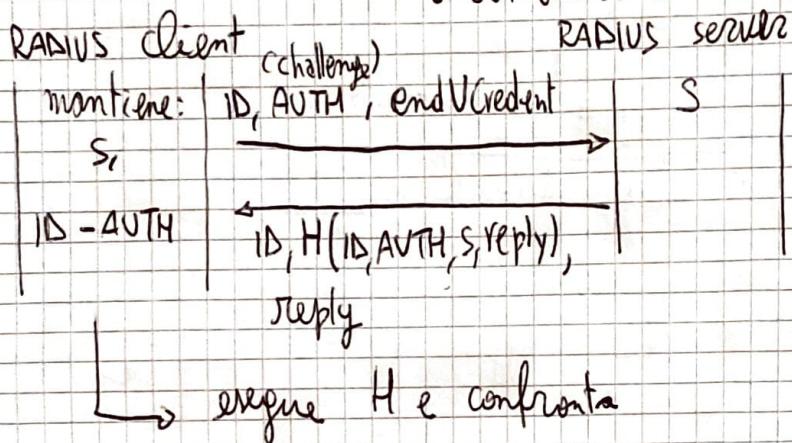
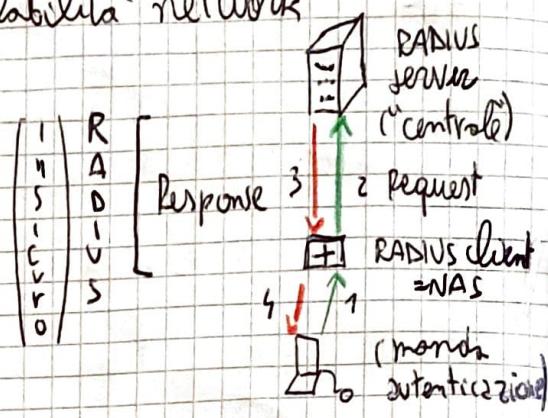
→ devo MAC_K (seq, AMF, RAND), se OK → creo SRES \doteq my identity

b ↴

RADIUS → serve per gestire scalabilità networks

- RADIUS server puo' comunicare con altro R. Server (proxy OP)
- PROBLEMI: solo reply autenticato

- hash, non HMAC, una MD5
- pw low entropy
- una stessa shared key per autenticazione
- authenticator hardcoded nel PTK!!



Nel Response pkt ho MD5(RequestID, RequestAuth, S, Pkt Response Info)

Nella request 16 byte random. ← campo 'authent'

PW encryption: $(pw^+) \oplus MD5(secret | RequestAuth)$ ← random in request.

Se pw > 16 byte? pluso stessa secret, errore.

CHAP + RADIUS • chp challenge generata da NAS, e inviata al server insieme a 'response' (fatta da user normale)

• parte NAS-server insicuro, poiché request-response non legati, potrei catturare [challenge, response] e spaccarmi per altro utente.

• Ho vulnerabilità, un MITM potrebbe combinare il msg → estendo "attributes"

Authenticator = HMAC-MD5(Request PKT)
s (Request PKT)

poiché ha il suo stesso authenticator dentro, lo pongo a 0.

"Dictionary Attack" to Shared Secret

- low entropy, vorrei avere Master Secret:
- client con Serial Number genera secret:
- $S_{\text{client}} = \text{HMAC}_{MS}(\text{Serial Number})$
- idea pessima: stesso segreto per PW encryption

- RADIUS non controlla riuso dell'authenticator, posso RIPRENDERE key stream
- non ho limiti di tentativi
- nei reply attack, posso creare dizionario request-response

oggi STANDARD per Authentication, Authorization, Accounting, ma è limitato in SCALABILITÀ ed ESTENSIBILITÀ \rightsquigarrow congestione

da RADIUS nasce 'Diameter'

- + completo
- + attuale
- diffuso

• DIAMETER HEADER

Flag: R (request), P (passa per 3°), E (error), T (ritrasmetto?)

• AVPs

V: vendor ID, M: se pkt è fondamentale, e non lo capisco, rigetto

P: privacy bit

• AGENTI DIAMETER

