Two parties (Alice and Bob) agree on a common secret key $SK$, in order to establish a secret channel. Basic security requirement:

- implicit authentication

- only the intended partners can compute the session key.

**Summary**

- Password-based Authenticenticated Key Exchange
- EKE, OKE and a generalization
- Trapdoor Hard-to-Invert Isomorphisms
- Examples

**Authenticated Key Exchange**

Two parties (Alice and Bob) agree on a common secret key $SK$, in order to establish a secret channel. Basic security requirement:

- implicit authentication

- only the intended partners can compute the session key.
Authentication

To prevent active attacks, some kind of authentication of the flows is required:
- **Asymmetric**: $(sk_A, pk_A)$ and possibly $(sk_B, pk_B)$
- **Symmetric**: common (high-entropy) secret
- **Password**: common (low-entropy) secret
  - *e.g.* a 20-bit password

Password-based Authentication

Password (low-entropy secret) *e.g. 20 bits*
- exhaustive search is possible
- basic attack: **on-line exhaustive search**
  - the adversary guesses a password
  - tries to play the protocol with this guess
  - failure ⇒ it erases the password from the list
  - and restarts...
  - after 1,000,000 attempts, the adversary wins
    - cannot be avoided
  
  We want it to be the best attack…

Dictionary Attack

- **Off-line exhaustive search**
  - a few passive or active attacks
  - failure/transcript ⇒ erasure of MANY passwords from the list: this is called **dictionary attack**
- To prevent them:
  - a passive eavesdropping
    - no *useful* information about the password
  - an active trial
    - cancels *at most one* password

Summary

- Password-based Authenticated Key Exchange
  - EKE, OKE and a generalization
    - Trapdoor Hard-to-Invert Isomorphisms
- Examples
**Encrypted Key Exchange**

**Bellovin-Merritt**

**Problems:**
- Encoding of $pk$ not often uniformly distributed in the ES plaintext space
- $pk$ and $c$ are rarely on the same space
- Nice exception: ElGamal (DH-EKE) on $<g>$
- Many security analyses in the ROM, ICM, ...

**Open Key Exchange**

**Lucks**

- The public key $pk$ is sent in **clear**:

**Surjection: Necessary**

- If not, given $c'$, one eliminates the $\pi$'s that lead to a $c$ which is not in the image set of $\text{EA}_{pk}$: *partition attack*
- If yes, given $c'$, any $\pi$ is possible: sending the correct $k$ means *guessing the good $\pi$*

**Efficient Implementation**

- Using the **one-time pad**, and bijections $\text{EA}_{pk} = f_{pk}$ and $\text{DA}_{sk} = g_{sk} = f_{pk}^{-1}$

- $f_{pk}$ must be a **bijection** onto a group $(G_{pk}, \otimes)$
- $f_{pk}$ must be *"hard-to-invert"
- $G$ must be a random function (RO) onto $G_{pk}$
Efficiently Sampleable

\( f_{pk} \) must be \textit{trapdoor} “hard-to-invert”, not necessarily “one-way”: but just \textit{samplable}

\((r, c) \leftarrow S(pk)\) such that \(r\) random in \(M_{pk}\) and \(c = f_{pk}(r)\)

\(pk\) must be easy to generate

\(f_{pk}\) must be a bijection ⇒ \textit{to be checked}

Hard-to-Invert: not Enough?

When \(pk\) is chosen by Alice

\(sk\) is unknown to the adversary

- the adversary can know only one pre-image \(r\) (for the guessed password \(\pi\))
- for other \(\pi\)’s, the “hard-to-invert” property prevents from extracting/checking other \(r\) values

This is the intuition... For the formal proof

- Hard-to-invert
- Bijection
- Morphism

Morphism: for the Proof

For checking a password, one uses \(k\) or \(SK\)

⇒ one must compute \(r\) (appears in \(H-H’\) queries)

- Either \(c’\) sent by Bob: from any correct \((\pi, r)\)
  such that \(c’ = f_{pk}(r) \otimes G(\pi)\), one can invert \(f_{pk}\)
  - by simulating \(c’ = f_{pk}(a)\)
    for a known \(a\)
  - by embedding the challenge \(y\) in \(G(\pi)\)
    \[ y = c’ \otimes f_{pk}(a) = f_{pk}(r) \otimes f_{pk}(a) = f_{pk}(r-a)\]
- Or by the adversary: from two correct pairs \((\pi, r)\)

Trapdoor Hard-to-Invert Isomorphisms Family

\(F = (f_{pk}, \_pk)\) trapdoor hard-to-invert isomorphisms

- \((pk, sk) \leftarrow G(1^k)\): generation
  - \(f_{pk}\) is an isomorphism from \(M_{pk}\) onto \(G_{pk}\)
- \((r, c) \leftarrow S(pk)\): sample
  - such that \(r\) random in \(M_{pk}\) and \(c = f_{pk}(r)\) (random in \(G_{pk}\))

- Given \(y\) and \(pk\), check whether \(y \in f_{pk}(M_{pk}) = G_{pk}\)
- Given \(y\) and \(sk\), easy to invert \(f_{pk}\) on \(y\)
- Without \(sk\), hard to invert \(f_{pk}\)
Password-based Authenticated Key Exchange

EKE, OKE and a generalization

Trapdoor Hard-to-Invert Isomorphisms

**Candidates**

**Diffie-Hellman**: $sk = x, pk = g^x$

$$f_{pk}(g^a) = g^{ax} = pk^a, g_{sk}(b) = b^{1/x}$$

$f_{pk}$ is not one-way, but hard-to-invert under the **CDH assumption**

$\Rightarrow$ classical DH-AKE variants (PAK or AuthA)

**RSA**: $sk = d, pk = (n,e)$

$f_{pk}$ is one-way under the **RSA assumption**, but $pk$ must contain a valid RSA key: NIZK proof

$\Rightarrow$ variant of “protected OKE”

**Candidates (Cont'd)**

**Square root**: $sk = (p,q), pk = n$

$f_{pk}$ is an automorphism onto $QR_n$, but for specific moduli only (Blum moduli)

$\Rightarrow$ to be checked: can be done (verified) efficiently

$f_{pk}$ is one-way under the **integer factoring problem**

$\Rightarrow$ the first **Password-Based Authenticated Key Exchange** based on **factoring**