Secret key agreement

Recommended reading
KL, Chapter 10, Sections 8.3.2 and 8.3.3

So far we have studied cryptography in settings where Alice and Bob share a secret key unknown to the adversary.

This begs the question: How can Alice and Bob agree on a key without the adversary learning about it?

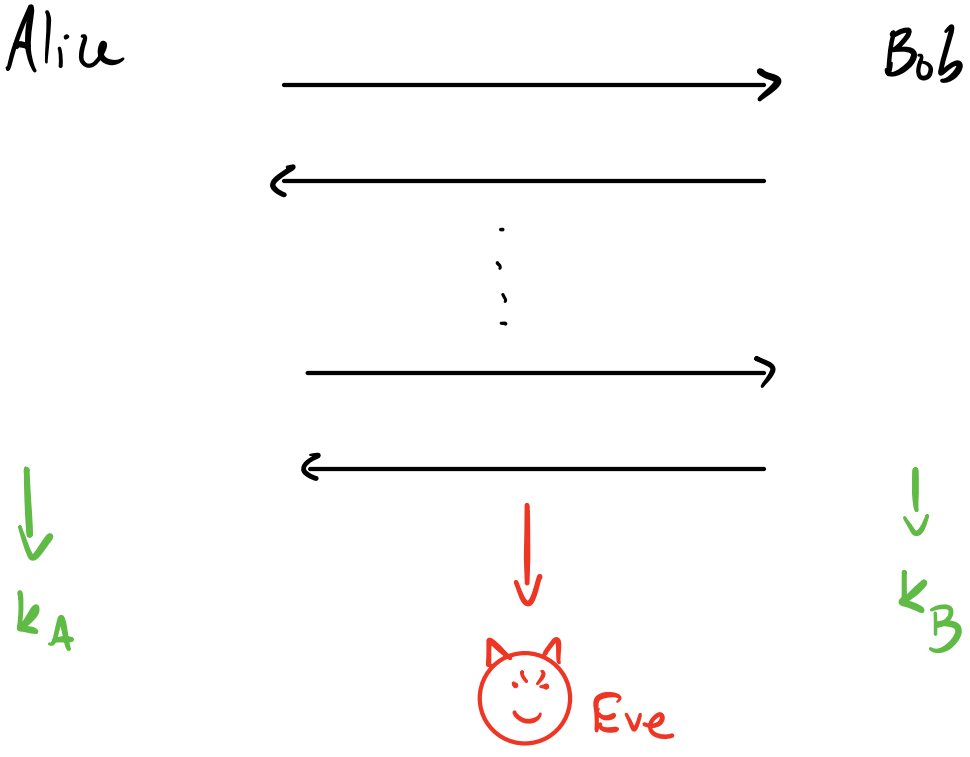
1) They could use secure couriers to transport bags full of keys. This happens in practice, but is slow and expensive.

2) They could rely on some trusted key distribution center. But trusting third parties is dangerous.

3) They could create the key from thin air, using just public communication.... ???
We'll explore option 3!

First, let's try to define a secret-key agreement protocol. We will assume that Alice and Bob communicate through a public but authenticated channel, meaning that the eavesdropper Eve can listen in on the conversation but can't tamper with it.

→ Alice and Bob communicate back and forth, then output their respective keys k_A, k_B.



Eve learns the whole transcript

Correctness: We say that the SKA protocol is correct if $k_A = k_B$ with probability 1

Security (against eavesdroppers): Intuitively, the SKA protocol is secure if based on the public transcript an efficient Eve can't distinguish between the key and an independent unif. random bitstring of the same length.

More precisely, consider the following game between an adversary $\mathcal{A}$ and a challenger, parameterized by n:

→ The challenger runs the SKA between Alice and Bob with initial input 1^n (Alice, Bob are PPT algos)

→ Let T be the communication transcript generated by the SKA protocol. Challenger samples $b \leftarrow \{0,1\}$.
If $b=0$: set $\hat{k} = k$, the real shared key produced by the SKA protocol ($k \in \{0,1\}^n$)
If $b=1$: set $\hat{k} \leftarrow \{0,1\}^n$.

→ Give $(T, \hat{k})$ to $\mathcal{A}(1^n)$.

→ $A(1^n, T, \hat{k})$ outputs b' and wins if $b' = b$.

We say that the SKA is secure if for any PPT adv A there is a negl. function $\varepsilon(n)$ s.t.

$$\Pr[A(1^n) \text{ wins game above}] \leq \frac{1}{2} + \varepsilon(n).$$

Diffie-Hellman SKA

In 1976, before the advent of modern crypto, Diffie and Hellman published a revolutionary paper.

https://ee.stanford.edu/~hellman/publications/24.pdf

Out of various amazing contributions, they proposed a beautiful SKA protocol!

Public Setup: G a cyclic group of order q.

g a generator for G.

Generated as $(G, q, g) \leftarrow Gen(1^n)$

To be discussed later

Alice		Bob
$a \leftarrow \mathbb{Z}_q$		$b \leftarrow \mathbb{Z}_q$
	$h_A = g^a$ →	
	← $h_B = g^b$	
$k_A = h_B^a$	Eve	$k_B = h_A^b$
$= (g^b)^a$		$= (g^a)^b$
$= g^{ba}$		$= g^{ab}$
$= g^{ab}$		

Please take a few minutes to absorb the beautiful simplicity of this protocol.

(for now, ignore the fact that this protocol outputs an element of G and not a bitstring)

Let's try to understand the assumptions required for the DH SKA protocol to be secure.

Eve learns (G, q, g, g^a, g^b) for $a, b \leftarrow \mathbb{Z}_q$.

The shared key is g^{ab}.

Therefore, it should at the very least be hard to compute g^{ab} given (G, q, g, g^a, g^b).

This is the Computational Diffie-Hellman (CDH) problem

CDH assumption for Gen:

For every PPT adv. A there is a negl function $\varepsilon(n)$ s.t. if $(G, q, g) \leftarrow Gen(1^n)$ then

$$\Pr_{a,b \leftarrow \mathbb{Z}_q}\left(A(G, q, g, g^a, g^b) = g^{ab}\right) \leq \varepsilon(n).$$

But our security notion for SKA requires that the key be indistinguishable from a unif. random string. So, actually, CDH isn't sufficient.

Instead, we require the

Decisional Diffie-Hellman assumption (DDH) for Gen:

For any PPT adv A there is a negl. function $\varepsilon(n)$ s.t. if $(G, q, g) \leftarrow Gen(1^n)$ then

$$\left| \Pr_{a,b \leftarrow \mathbb{Z}_q}\left(A(G,q,g,g^a,g^b,g^{ab})=1\right) - \Pr_{a,b,c \leftarrow \mathbb{Z}_q}\left(A(G,q,g,g^a,g^b,g^c)=1\right)\right| \leq \varepsilon(n)$$

The following theorem follows essentially by definition of DDH, if we modify the notion of security to require that it's hard to distinguish the key from a unif. random element of G.

Thm: Suppose that DDH holds for Gen. Then, the Diffie-Hellman SKA protocol is secure.

Note: We modified our security definition to require that the key be indist. from a random element of G. In practice, this isn't a problem. For example, we can use k as an input to a hash function that is assumed to behave like a random oracle. More generally, Alice and Bob can apply a key-derivation function to their shared key k.

Another note: We assumed an authenticated channel between Alice and Bob. The DH protocol is not secure if we remove this assumption. SKA over unauthenticated channels is a very interesting problem that we won't discuss.

Two natural questions arise:

1) Can we design secure SKA protocols generically based on things like OWFs / OWPs / CRHFs?

It turns out that the answer is, at least, not with "standard" techniques. More precisely, Impagliazzo & Rudich (STOC 1989) showed that the existence of a proof for this statement implies that $P \neq NP$

https://dl.acm.org/doi/10.1145/73007.73012

This shows that we're dealing with a fundamentally different type of crypto!

2) How hard are the CDH and DDH problems?

This depends on our choice of cyclic groups.

Obs 1: CDH is easier than the discrete logarithm problem. If we can efficiently compute discrete logarithms of random group elements then we can recover (a, b) from (g^a, g^b), compute ab, and compute g^{ab}

Obs 2: DDH is easier than CDH.

Obs 3: There exist groups where we believe that Dlog and CDH are hard (for classical computers) but where DDH is easy.

For example, consider $G = \mathbb{Z}_p^*$ with p prime of size $p \approx 2^n$.

As we discussed before, the Dlog problem wrt $\mathbb{Z}_p^*$ is believed to be hard for classical computers.

However, we can easily solve DDH in $\mathbb{Z}_p^*$.

How?

Recall that $\mathbb{Z}_p^*$ is a cyclic group of order $p-1$. There exists a generator $g \in \mathbb{Z}_p^*$ such that

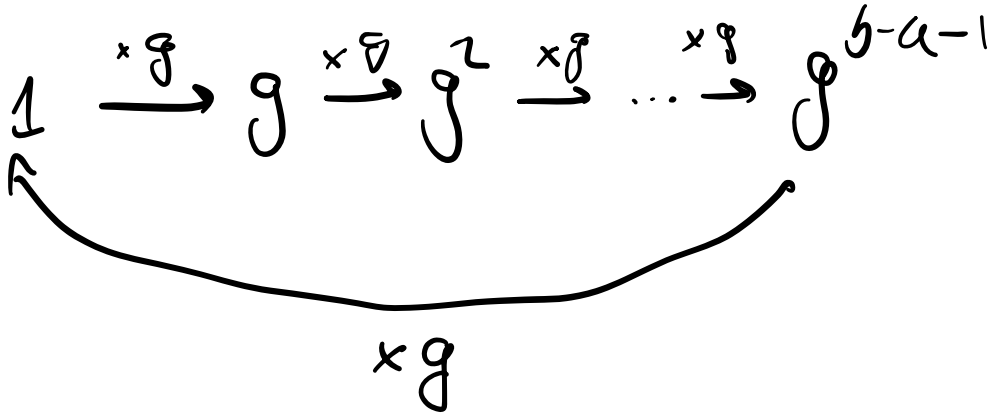
$$\mathbb{Z}_p^* = \{g^0 = 1, g^1, g^2, \ldots, g^{p-2}\}$$

these are all distinct.

if $g^a = g^b \bmod p$ for $b > a$

then $g^{b-a} = 1 \bmod p$

and so the group generated by g would have order $\leq b-a$.



Since g generates a group of order $p-1$, the above implies that $g^k = 1 \mod p$ if and only if $k = 0 \mod p-1$.

Therefore, for any $h \in \mathbb{Z}_p^*$ we have

$$h^{p-1} = g^{k(p-1)} = (g^{p-1})^k = 1^k = 1 \mod p$$

We define the set of quadratic residues mod p as

$$Q_1 = \{ g^0 = 1, g^2, g^4, \ldots, g^{p-3} \}$$

Then are all $h \in \mathbb{Z}_p^*$ such that $h = g^{2k} \mod p$ for some k. Note that $|Q_p| = \frac{p-1}{2}$.

Key property:

$$h \in Q_p \Rightarrow h^{\frac{p-1}{2}} = 1 \bmod p$$

$$h \notin Q_p \Rightarrow h^{\frac{p-1}{2}} = -1 \bmod p$$

Proof:

- $\left(h^{\frac{p-1}{2}}\right)^2 = 1 \bmod p \Rightarrow h^{\frac{p-1}{2}} = \pm 1 \bmod p$

- If $h \in Q_p$ then $h = g^{2k} \bmod p$ for some k.

  So $h^{\frac{p-1}{2}} = g^{2k \cdot \frac{p-1}{2}} = g^{k(p-1)} = 1 \bmod p$

  ↓ discussion above

- The polynomial $h^{\frac{p-1}{2}} - 1$ has at most $\frac{p-1}{2}$ roots. Since every $h \in Q_p$ is a root and $|Q_p| = \frac{p-1}{2}$, it follows that $h^{\frac{p-1}{2}} - 1 \neq 0 \bmod p$ for all $h \notin Q_p \Rightarrow$ for $h \notin Q_p$ we have $h^{\frac{p-1}{2}} = -1 \bmod p$.

Attack on DDH:

$\mathcal{A}$ receives $(G = \mathbb{Z}_p^*, g, p-1, g^a, g^b, g^c)$

Needs to guess whether $c = ab$ or if c is independent of a, b and unif. random over $\mathbb{Z}_{p-1}$.

1) Compute $(g^a)^{\frac{p-1}{2}}, (g^b)^{\frac{p-1}{2}}, (g^c)^{\frac{p-1}{2}}$

to determine the parities of a, b, c.

2) Check if $a \cdot b$ and c have the same parity

If $c = a \cdot b$, this always holds

If $c \leftarrow \mathbb{Z}_{p-1}$ independent of a, b, then c is even with probability $1/2$, while $a \cdot b$ for $a, b \leftarrow \mathbb{Z}_{p-1}$ is even with probability $3/4$

$\Rightarrow$ advantage $1/4$ in DDH over $\mathbb{Z}_p^*$.

How to chose groups for DH SKA?

Usually people work with prime order groups.
Note that $\mathbb{Z}_p^*$ doesn't have prime order.

→ Choose $p = 2q+1$ with p, q prime and work with the subgroup of quadratic residues mod p, which has order $\frac{p-1}{2} = q$

→ Points in elliptic curves