Group Diffie-Hellman Key Exchange (Spillman Book)

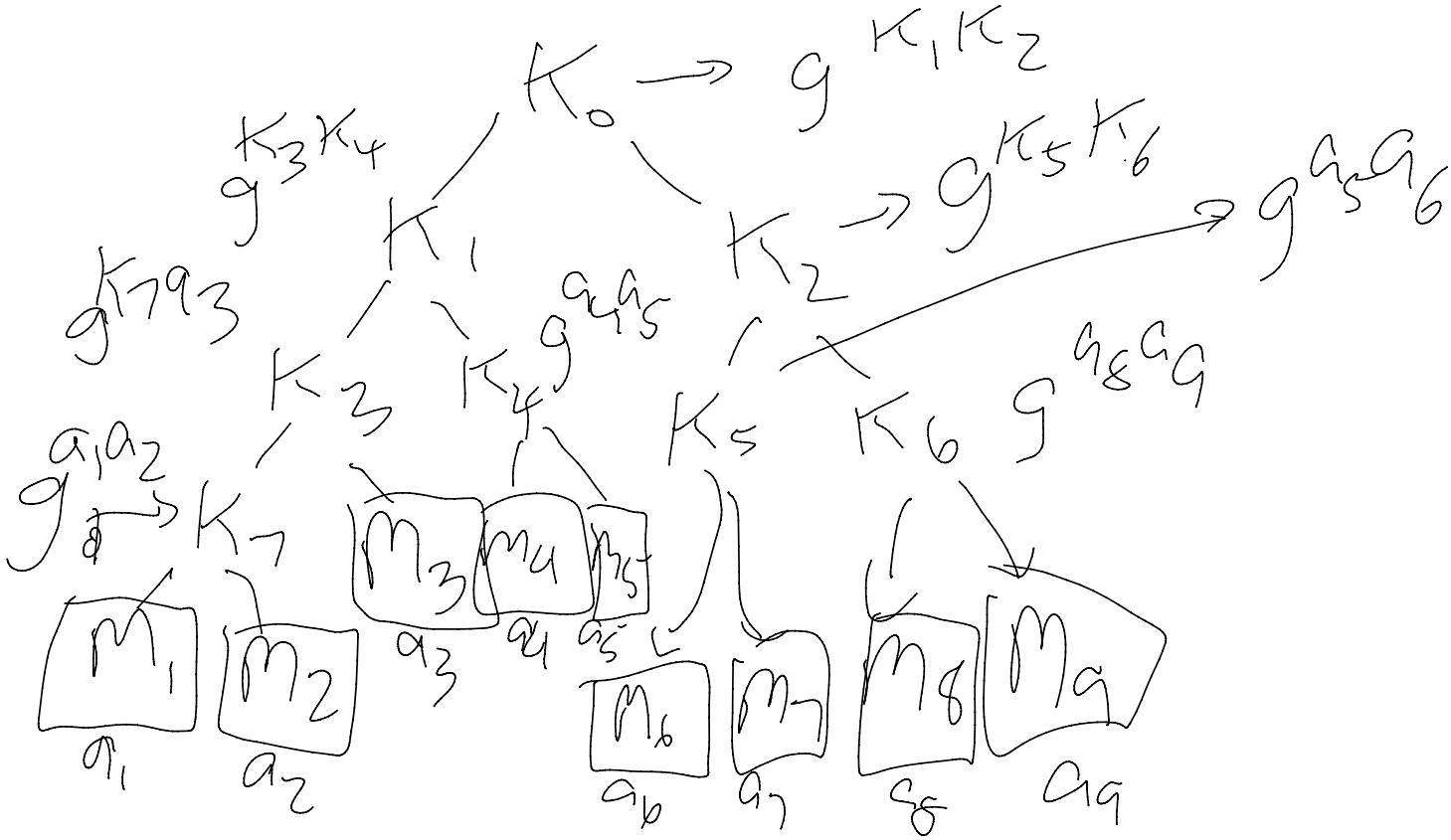
Wednesday, November 18, 2020 11:39 AM

Sometimes we just share a secret key with one other person, but in some instances, we want more flexibility.

We might have a group at work that is all privy to the same set of information, and it would be nice if everyone in that group had a shared secret key, so that group communication didn't have to be encrypted by tons of different shared keys between pairs of users.

This is the motivation behind group keys…

In some work place, we have some users, and pairs of users may share a secret key, but ALSO, there will be some groups (subsets of users) who all want to share the same secret key…



K7 is shared between M1, M2
K3 is shared between M1, M2, M3
K4 is shared between M4, M5
K1 is shared between M1, M2, M3, M4, M5
K5 is shared between M6, M7
K6 is shared between M8, M9
K2 is shared between M6, M7, M8, M9
K0 is shared by all.

Strengths

We can have different shared keys between different groups.
It turns out we can add or remove users fairly efficiently, in terms of run time, we end up having to change O(log n) keys, where n is the total number of users.

Weaknesses

There is a pretty forced structure, not easy to form arbitrary groups.
When we add a new user, there are limited sets of groups we may add her to.

How do we assign keys?
i
We are using Diffie-Hellman.
There is a publicly global prime number p.
There is a publicly global generator g.
All people share these two pieces of information.

Each end user picks their own private key a_i, for user i.

1. Each pair of people who are leaf nodes connected by the same capital K (virtual person), will do a regular Diffie-Hellman Key Exchange and the shared key becomes the value of that capital K.
2. If you are a user and your sibling node isn't a real person but is a virtual person (a capital K in the drawing), then you'll do a Diffie-Hellman Key Exchange with that node (a computer) such that the computer's secret key is what was previously calculated. Iterate this process from the bottom to the top of the tree until all shared keys are filled in.

$K_7 = g^{a1a2}$
$BK_7 = g^{K7}$ (this is what the fake person sends to user M3, which M3 will then raise to the a3 power to get their shared key).

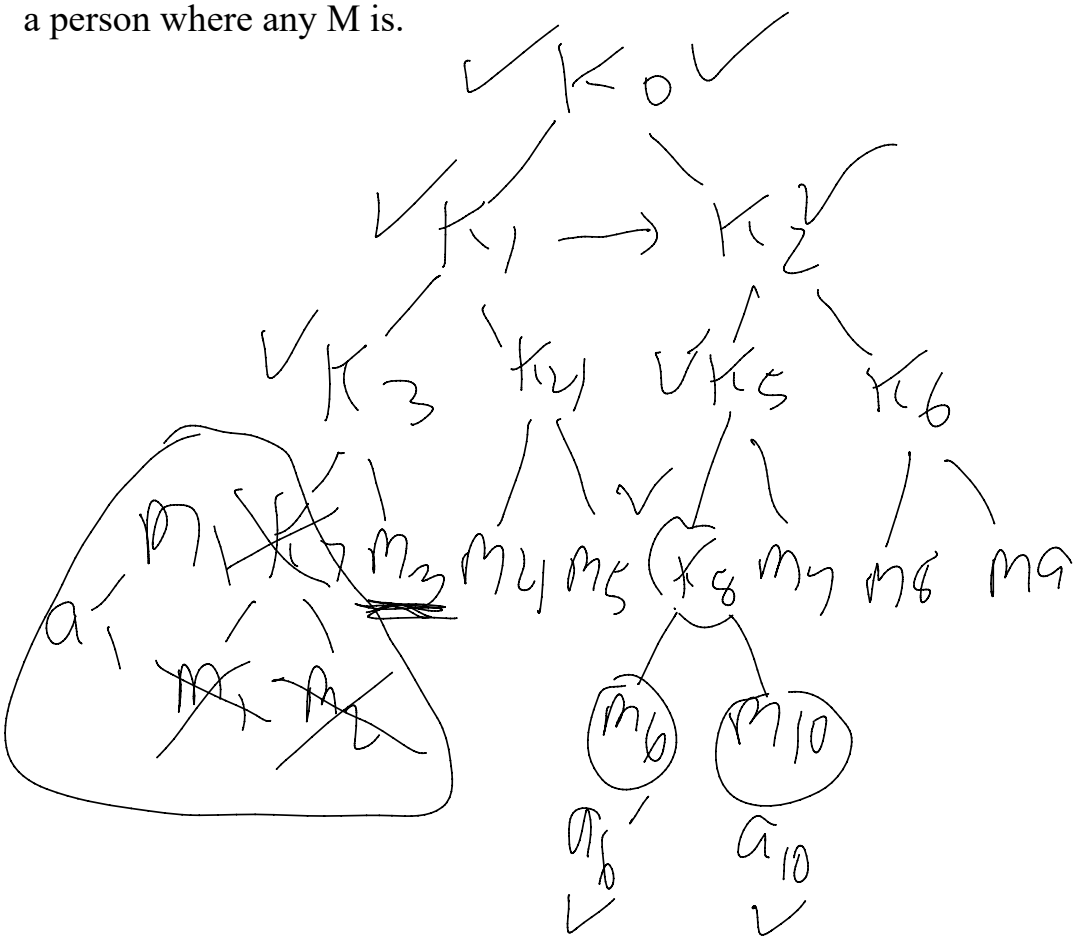
When M3 and K7 do the Diffie-Hellman key exchange, their shared key will be:

$K_3 = g^{K7a3}$ (Shared key between M1, M2, and M3)

What about adding a user?

We will choose a leaf node to add, and then, we will bubble up the tree,

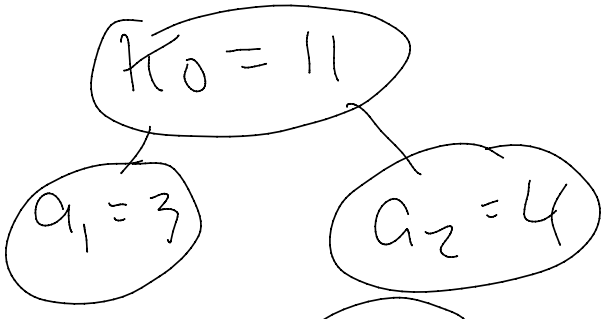
changing only the shared keys that are necessary to change. We can add a person where any M is.



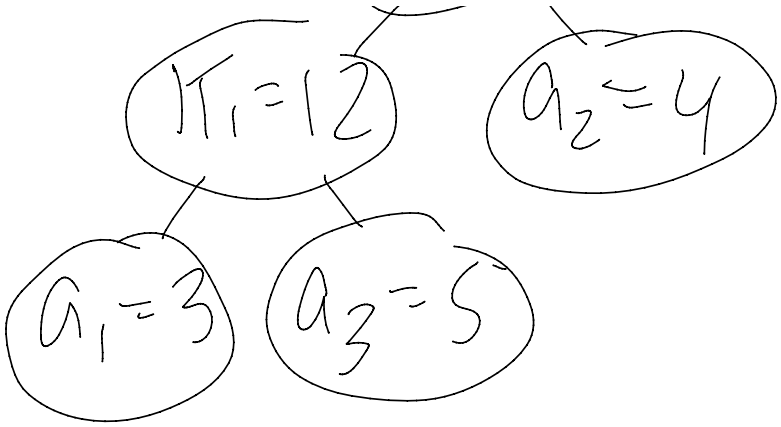
Now, let's consider removing a user, let's say M2…

Small example by hand

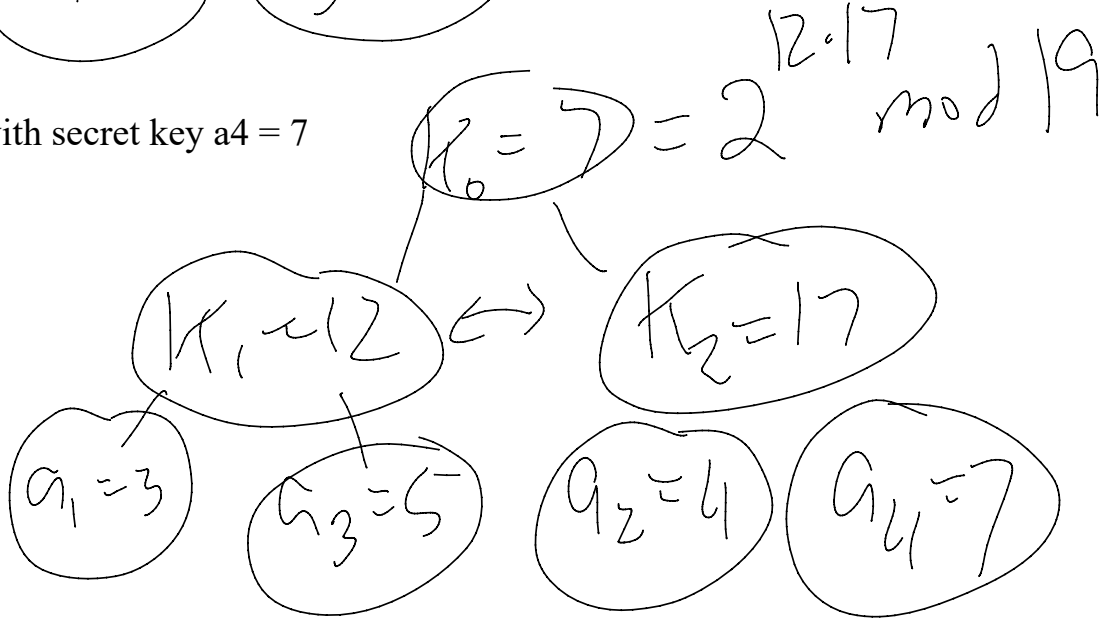
p = 19, a = 2
Start with m1, m2.
a1 = 3, a2 = 4 --> shared key is $2^{12} \bmod 19 = 11$



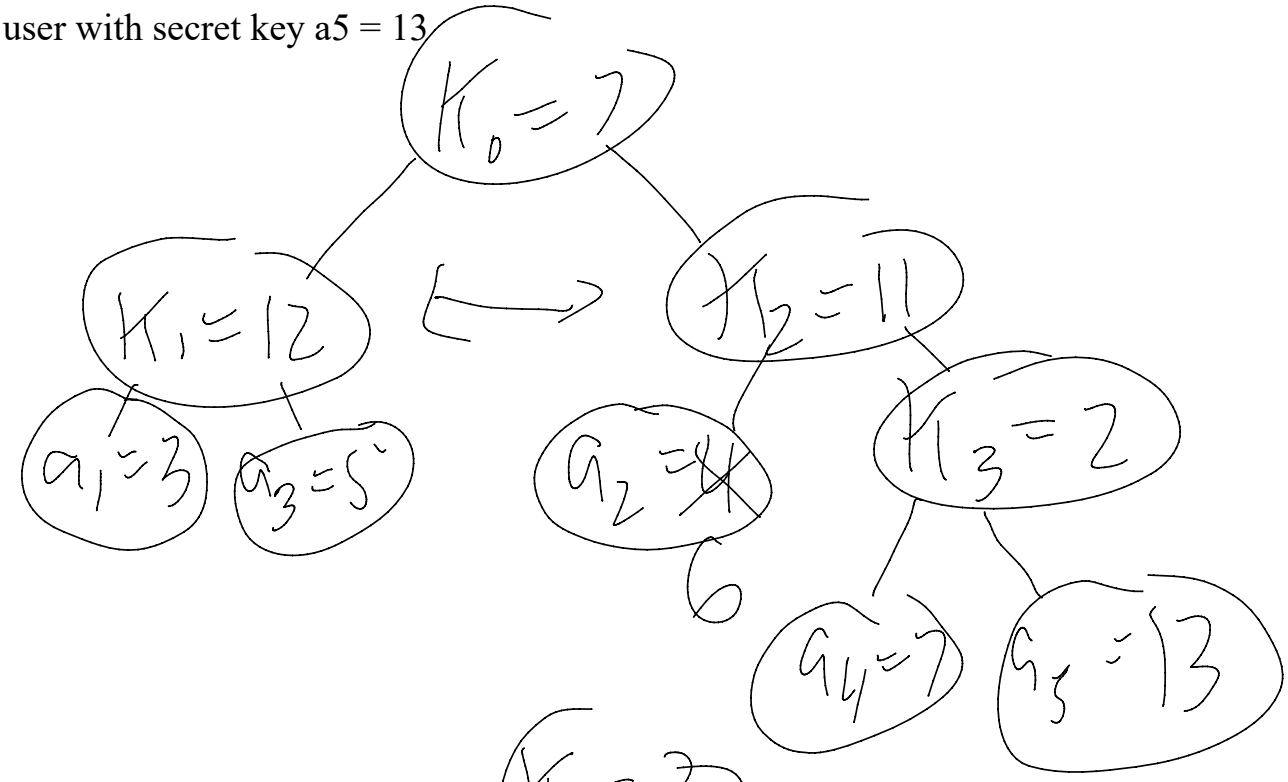
Add a user with secret key a3 = 5



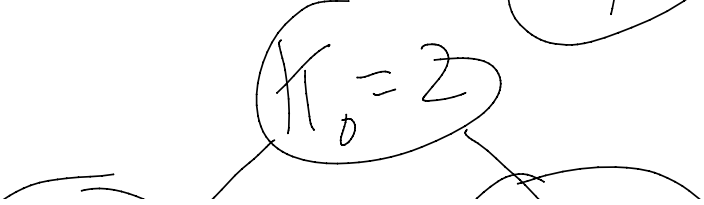
Add user with secret key a4 = 7



Add user with secret key a5 = 13



Delete user M1…



Delete user M1…