Thanks, in advance, for whoever volunteers to take notes!

MLS@IETF103 2018-10-05 @ 16:10 UTC+07:00

- Two implementations at hackathon -- Wire and mlspp
- Crypto interop issue

TreeKEM: double-join

DKG: How do you get into the state where the root node knowns but one of
the children are not?

RLB: Adds, updates, and removes. Adds leave leaf-to-root path empty, removes remove the path from leaf to root, and updates heal the path.

EKR: Agree that Init state is terrible, but all protocols have a linear
setup phase.

Ben: Don't you also expect many group members to send update right away.

RLB: Yes, in practice, and we're designing for asynchronous behavior. If
half the group never comes online, then half the tree is never populated.

RLB: We need to have the Book information available and serializeable to
inform people of group changes.

EKR: Is this is all a terrible idea?

RLB: Yes, outside of this one use case.

DKG: What is the x-axis here?

Raphael: X-axis is the time of people coming online and doing an update.

RLB: No adds or removes here, and there are more time steps here. (Previous chart was first 50 or so, this is first 1000.)

EKR: How much simpler does life get if Creator is special and cannot be
evicted.

Raphael: Gets cheaper if you don't get clean the top of the tree. Cost seems reasonable while still being able to evict the Creator.

RLB: If you want double join at Init time and evict the creator, then you need one bit per node in the tree to indicate if the creator is double joined.

EKR: Recommendation is creating tree and then giving everyone bookkeeping information to evict the Creator.

Ben: If you allow creator to be double joined but not evicted, there's no extra state.

RLB: If you prepopulate the whole tree, you're never *not* log(N) (no warm up period). You don't have to do any additional DH operations because you already give them the secrets.

DKG: Discussion is taking place based on assumption that large groups of
people will be added (?)

Raphael: Can probably do some simulations to determine if prepopulating
is better than Init.

RLB: Sending messages is constant time. Adds are easy. Updates and removes are expensive since information goes to subsets of the tree.

EKR: Prefer optimizing the joining operation in favor of expensive preopulating costs.

DKG: (missed)

EKR: Two sorts of models: (1) bulks adds and (2) spontaneous adds and slow growth. Let's keep things simple.

DKG: Predictablility of management might be easier or better in some cases.

RLB: Do Init with full tree warmup.

Raphael: Agree.

Ben: How much more work would it be to do a new Init for a new tree?

Raphael: Depends on how many people are in the tree. And you destroy everyone's local state by doing so.

Mark: Trees can get quite badly unbalanced?

Raphael: If empty nodes are skewed (?), yes.

Authentication

<nada>

Hierarchical key derivation

DKG: Considered usability implications of having the phone online in order to start new chats. Could the phone vend not-yet-used HDK roots to
start new chats?

Nadim: Trying to find middle ground between Signal and WhatsApp.

DKG: Recovery from compromise is the goal here.

Nadim: Could new HDK keys be derived each epoch, or would that completely destroy usability?

DKG: Requires a lot of focus on usability.

Emad: Disagree with Signal's usability decision over WhatsApp. Current draft assumes that multiple devices use multiple identity keys.

Nadim: Only comparing properties of different systems, not trying to argue for superiority.

RLB: Right now the spec only talks about identity keys within the scope
of a conversation. You might have a separate identity across conversations. Manual key verification regimes might just use one identity key across conversations. Might consider comparing against continually re-signing identity keys.

Emad: Mixing concept of identity and signing keys. Signing keys are always changing. Identity keys are long-term keys.

Jon: How does this compare against a situation where every device has its own key, but every QR code and sign key has the "master" device sign
an identity key of a new device. One device has the master identity key,
other devices have generated identity keys.

Handshake encryption

RLB: Clarifying that the cost is data transfer cost, not DH operations.

DKG: We are assuming separate identities per device, and cost is operations between devices related to the same identity (?)

Ben: Chart is showing that Welcome HS linear in number of group messages. We already accepted linear state in terms of roster and identity keys. How worried do we need to be about this?

Raphael: Trying to make it clear that this exists.

DKG: Current mechanism in getting state to new user is through distribution server. Nothing in spec that says it cannot pass through distribution server. So "should we allow OOB transfer of state" seems odd. Are we saying out-of-order?

Raphael: (missed)

EKR: Sending message needs the root, reading messages needs the whole path (?) Would be useful to lay out what parts of the tree are needed for each operation, and which parts of the tree belong to whom.

Jonathan: In practice, no one will verify conversations. They will simply try to read messages (?).

RLB: Can we come up with a hashing scheme that can be selectively updated?

DKG: Throwing in towel in metadata protection now is premature. No handshake encryption seems to be that. Should there be some cutoff between the modes so that some groups can continue to have better protections?

MT: Should endpoints reject messages (under some condition)?

RLB: Yes, with a NACK scheme.

MT: Could server also use epochs to reject?

RLB: Need some consistency.

Raphael: How much identifiable information does the server really see with handshake encryption?

MLS@IETF103 2018-10-08 @ 11:20 UTC+07:00

Agenda Bashing

Formal Analysis presentation may be short

MLS Message protection (Richard Barnes)

Russ – Is the goal per message PFS or per Key Schedule? Might not update the keyschedule for every message?

RB – New key and nonce derived for every message

EKR – Throw away keys after every message. No problem with using an application key with more than one message, but the key schedule would generate the same nonce so the key schedule would have to change,

RB – loose state lead to nonce re-use problem (?)

EKR – replace the empty string in HKDF expanded with random value to avoid accidental key reuse.

DKG – you could also introduce 4 – 8 octets when process starts to avoid state problem

EKR – Also failure of auth with GCM not as bad because of additional signing,

EKR – Generate per message chains for every-one in system. Hold epoch 0 for a while, do not know when you can destroy a key. Syncrhonization is required to tighten up PFS.

EKR – If you generate keys on demand in a naieve way then you open up for worse PFS properties.

RB – Per sender ratcheting and trees can be better

Emad OMara – Group chain is more complicated for sybncrhonization

RB – ordering does not need to be enforce, In order to get PF you need sync

EKR – In a large group the amount of bogus keys you have to generate is high. Not everyone sends during every update step, but you need to generate keys and store them. I'm still in faveo of persender chains, need tree.

DKG – TO get forward secrecy every copy of the key needs to be destroyed. Offline participants may not be able to delete. Are we overengineering? With large groups the cost is high and PFS is difficult to achieve.

EO – Something to put into architecture draft

RB – seems like consensus is on per sender chaining

DKG – there is no place for the nonce in the application struct might need to add one

EKR – Why are cipher suites are tied to curves

RB – idea was to bundle all the parameters in one code point

Ben Kaduk – Have one joint and keep it oiled.

EKR – Signature construction – added for cut and paste attack, not a bad design, but we have no analysis on if the is the right thing include in the signature. Do we need more statue. Is it possible to get 2 groups with same group id, could allow for ut and paste ID). Just needs more analysis

DKG – ins addition to formal analysis we have failed to include the right context in messages. Can you confuse someone if you reply to Richard or Eric. Could add dag, but we don't know how to render. DAG is better.

RB – send PR

Several people have read document, but we read more.

Who will review protocol doc: EKR and Chris offer to review.

Who will review arch: DKG ekr an RB

Interim: Before of after RWC 2019 (January) – San Jose