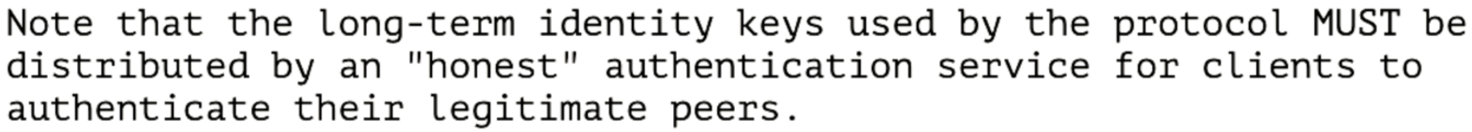
User Authentication within Groups

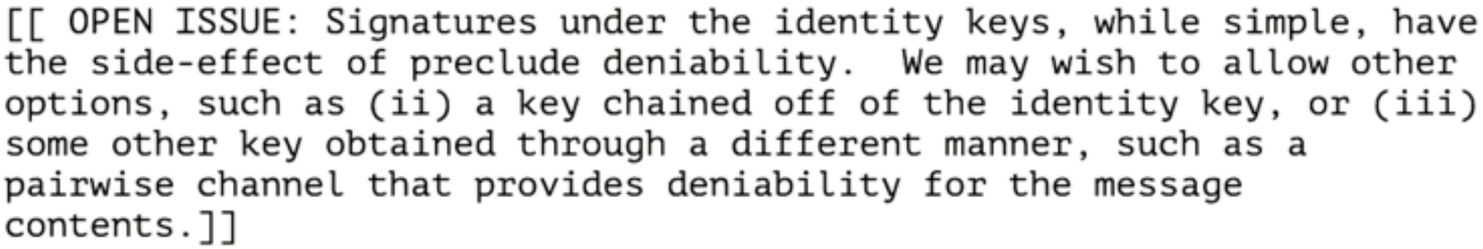
Britta Hale

Based on joint work with Benjamin Dowling

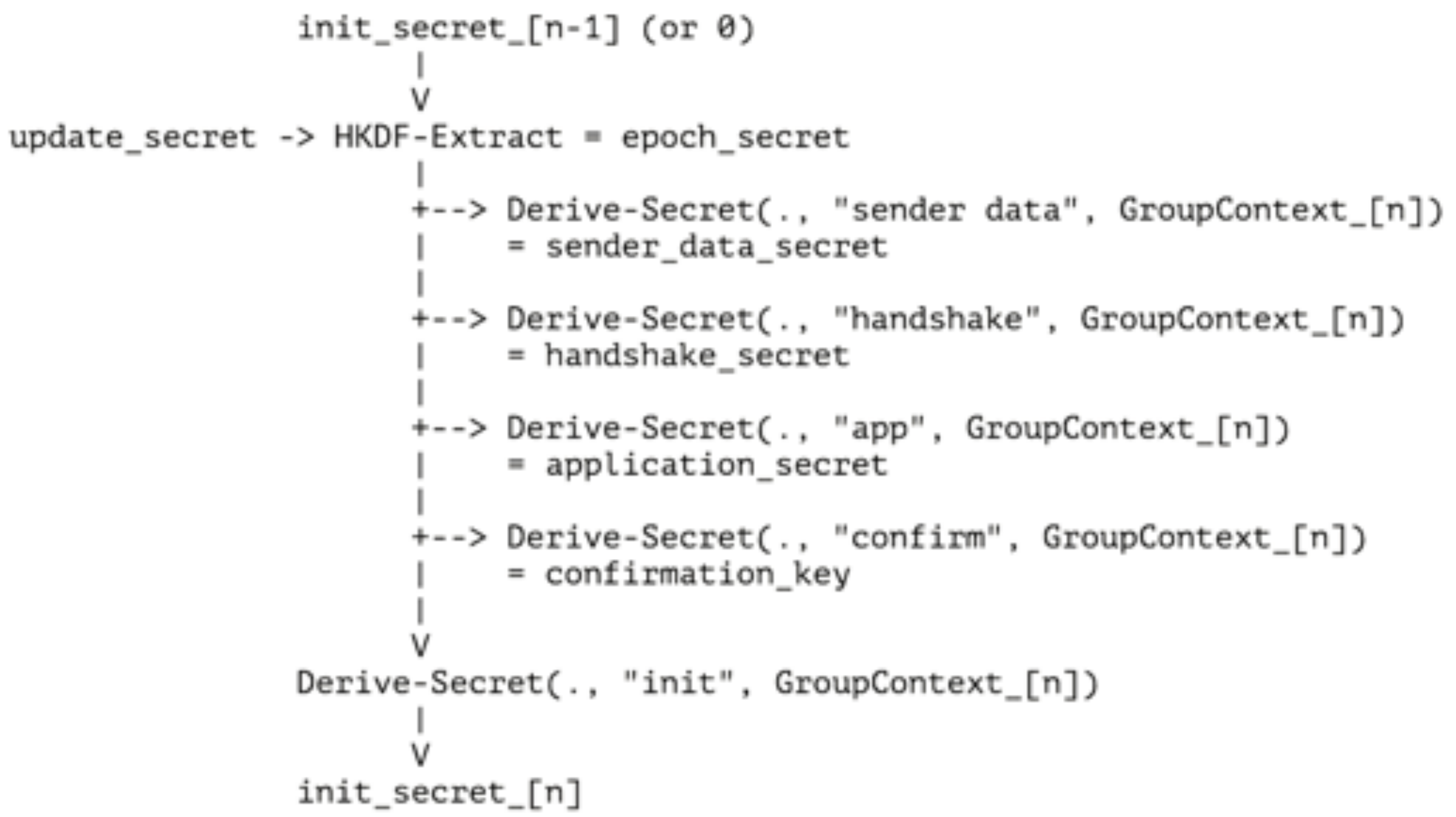
*The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.



Note that the long-term identity keys used by the protocol MUST be distributed by an "honest" authentication service for clients to authenticate their legitimate peers.

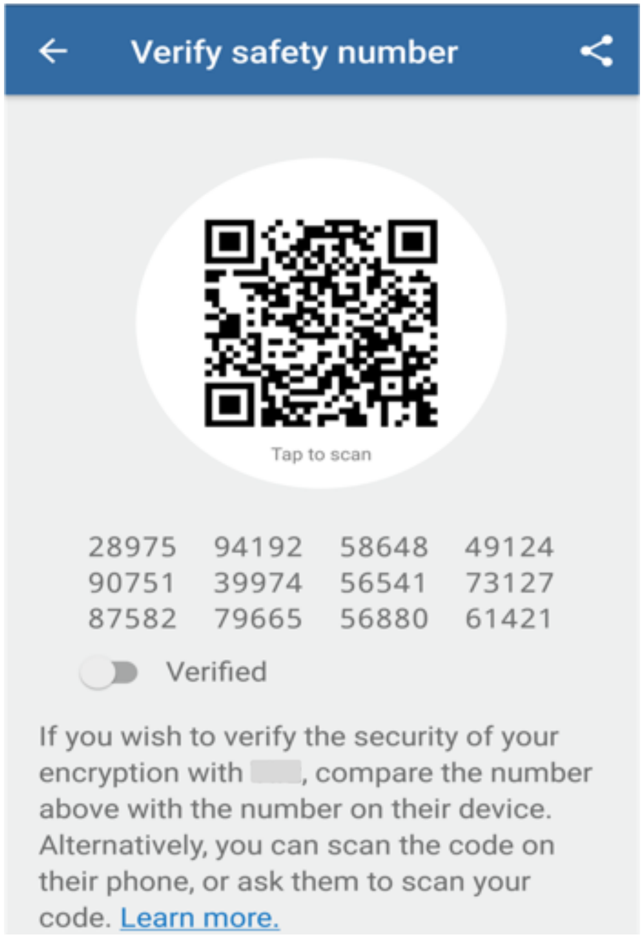


[[OPEN ISSUE: Signatures under the identity keys, while simple, have the side-effect of preclude deniability. We may wish to allow other options, such as (ii) a key chained off of the identity key, or (iii) some other key obtained through a different manner, such as a pairwise channel that provides deniability for the message contents.]]



```
               init_secret_[n-1] (or 0)
                     |
                     V
update_secret -> HKDF-Extract = epoch_secret
                     |
                     +--> Derive-Secret(., "sender data", GroupContext_[n])
                     |    = sender_data_secret
                     |
                     +--> Derive-Secret(., "handshake", GroupContext_[n])
                     |    = handshake_secret
                     |
                     +--> Derive-Secret(., "app", GroupContext_[n])
                     |    = application_secret
                     |
                     +--> Derive-Secret(., "confirm", GroupContext_[n])
                     |    = confirmation_key
                     |
                     V
               Derive-Secret(., "init", GroupContext_[n])
                     |
                     V
               init_secret_[n]
```

Lessons from Signal Authentication



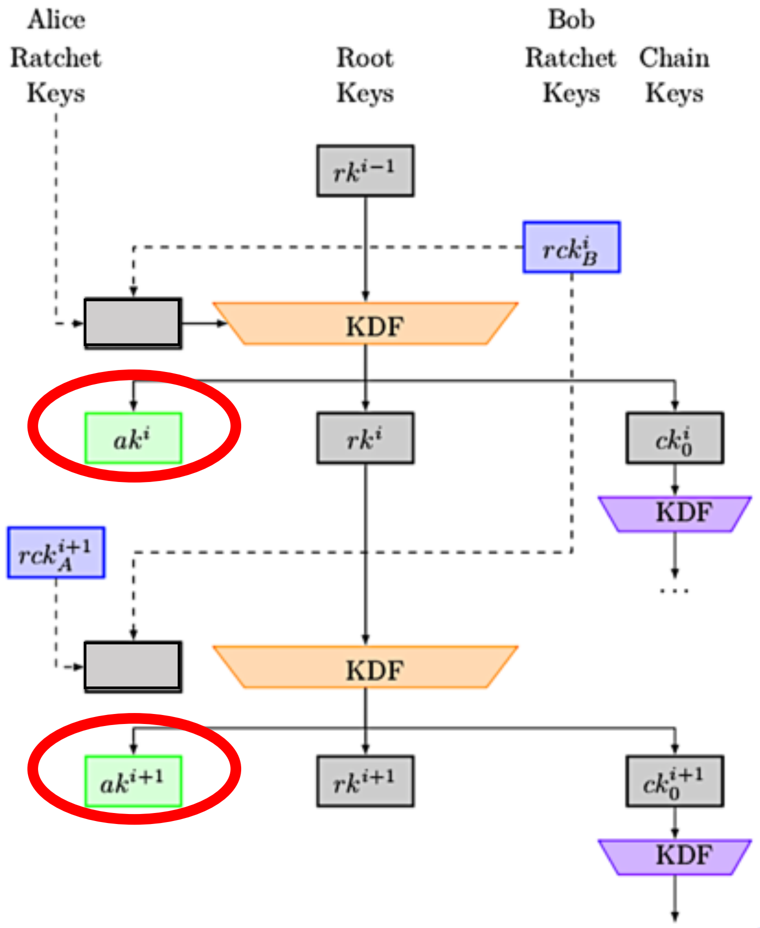
28975 94192 58648 49124
90751 39974 56541 73127
87582 79665 56880 61421

Verified

If you wish to verify the security of your encryption with [redacted], compare the number above with the number on their device. Alternatively, you can scan the code on their phone, or ask them to scan your code. Learn more.

Epoch-Level Authentication

Modified Device-to-User Signal Authentication

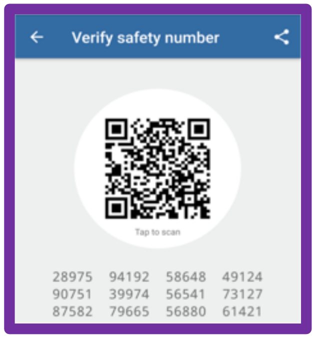


New QR-code computation:

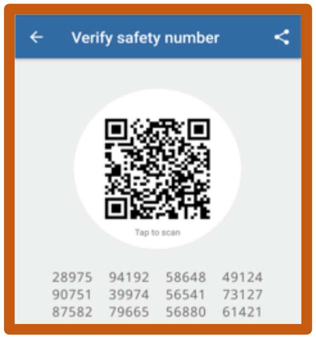
$$\mathtt{fprint}^{i-1} = \mathsf{HMAC}(ak^{i-1}, H^{i-1}\|\mathtt{fvers}\|role)$$
$$\mathtt{fprint}^{i} = \mathsf{HMAC}(ak^{i}, H^{i}\|\mathtt{fvers}\|role)$$

*Session specific

*Asynchronicity in computation

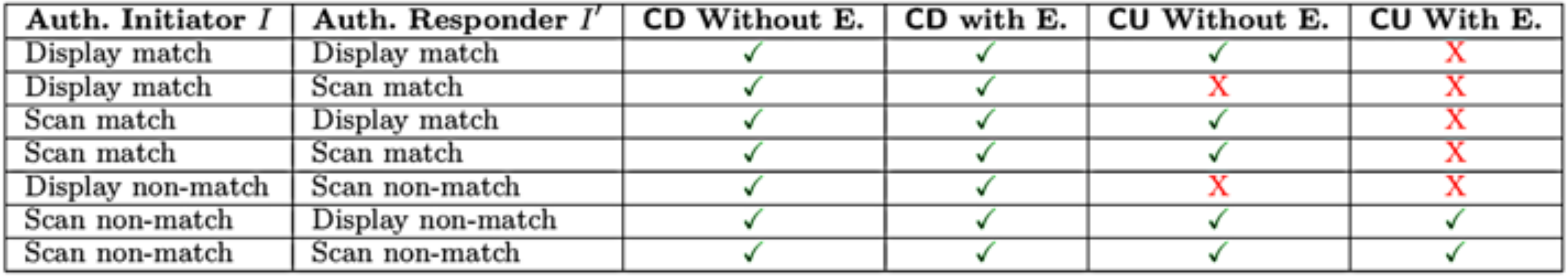


28975 94192 58648 49124
90751 39974 56541 73127
87582 79665 56880 61421



28975 94192 58648 49124
90751 39974 56541 73127
87582 79665 56880 61421

Achievable Guarantees

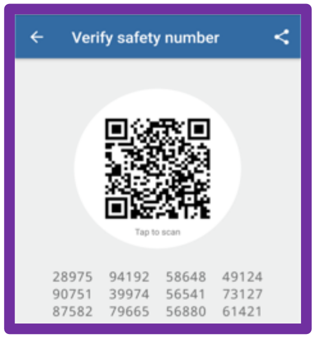


Auth. Initiator I	Auth. Responder I'	CD Without E.	CD with E.	CU Without E.	CU With E.
Display match	Display match	✓	✓	✓	X
Display match	Scan match	✓	✓	X	X
Scan match	Display match	✓	✓	✓	X
Scan match	Scan match	✓	✓	✓	X
Display non-match	Scan non-match	✓	✓	X	X
Scan non-match	Display non-match	✓	✓	✓	✓
Scan non-match	Scan non-match	✓	✓	✓	✓

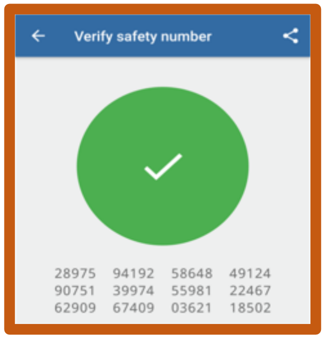
CD: Compromised Device

CU: Compromised User

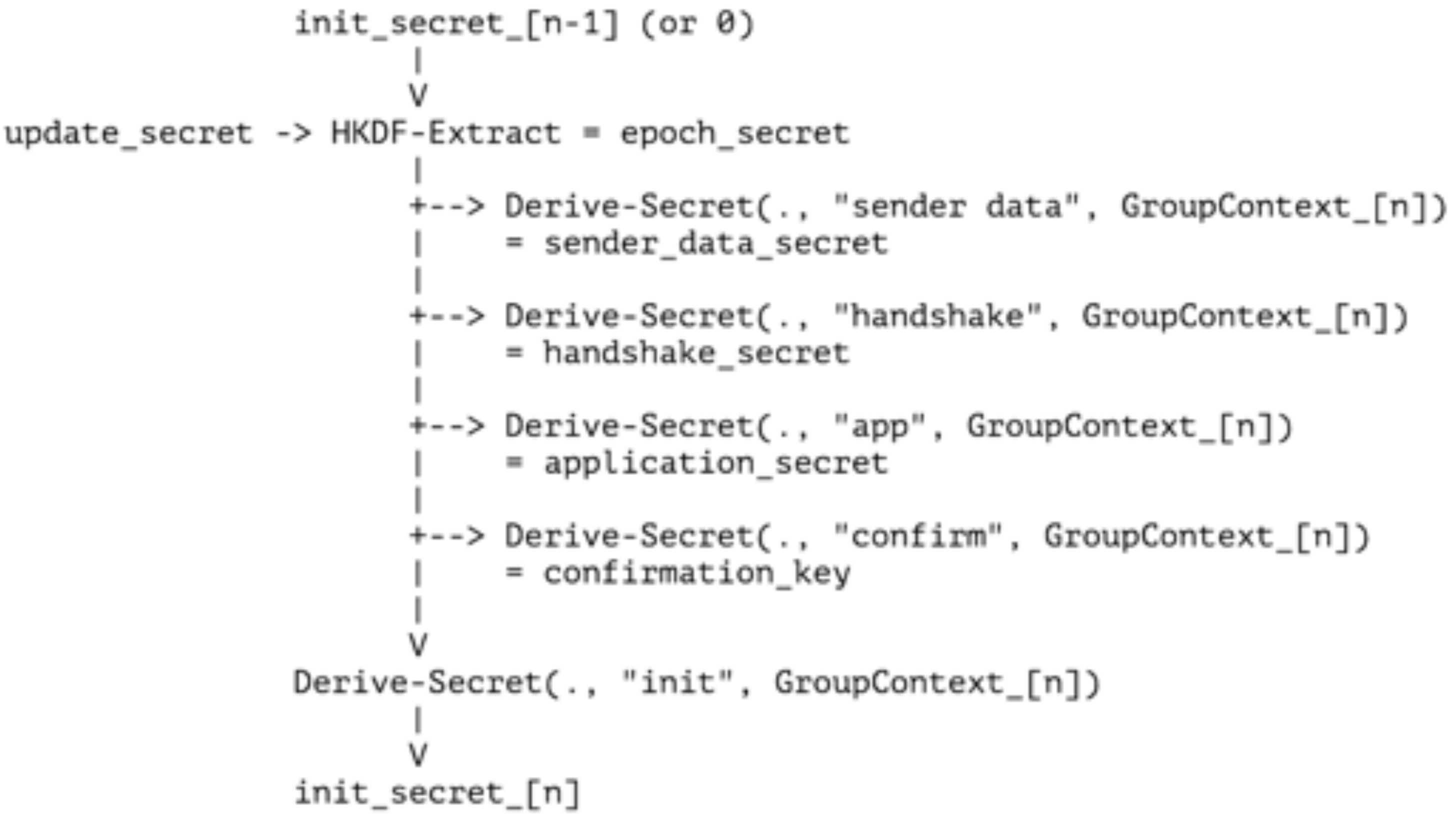
E: Eavesdropper



28975 94192 58648 49124
90751 39974 56541 73127
87582 79665 56880 61421

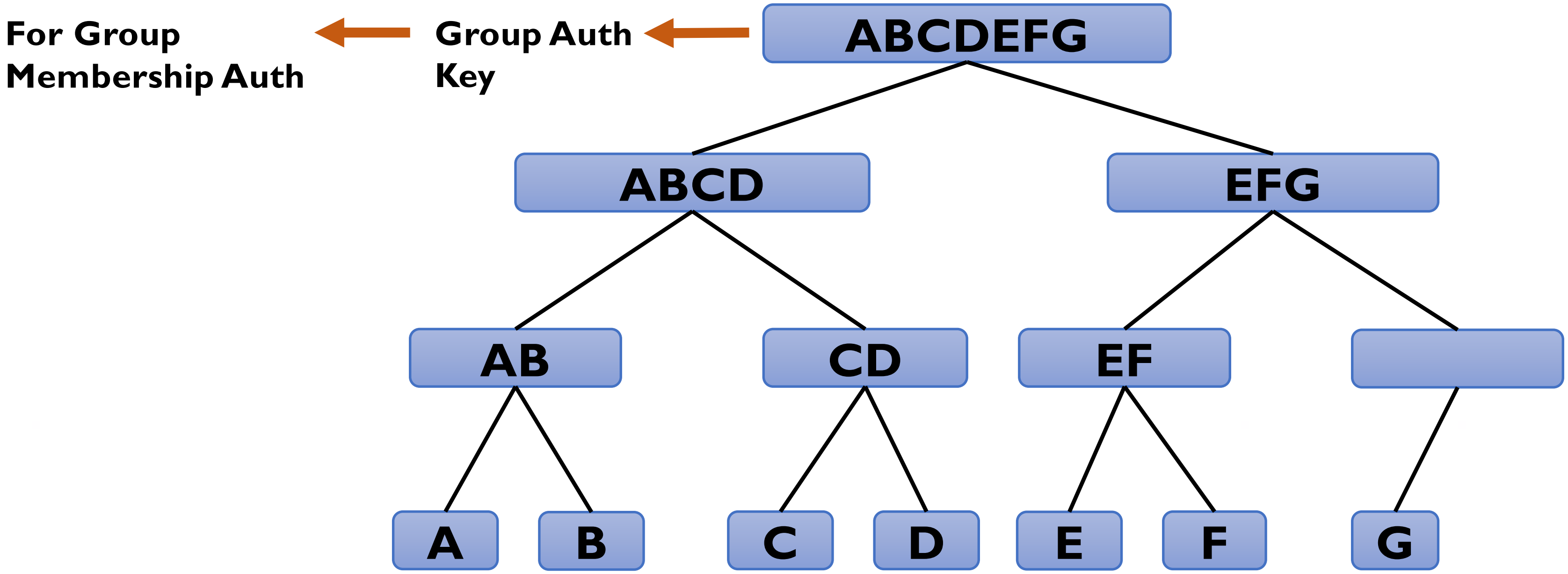


28975 94192 58648 49124
90751 39974 55981 22467
62909 67409 03621 18502



```
               init_secret_[n-1] (or 0)
                     |
                     V
update_secret -> HKDF-Extract = epoch_secret
                     |
                     +--> Derive-Secret(., "sender data", GroupContext_[n])
                     |    = sender_data_secret
                     |
                     +--> Derive-Secret(., "handshake", GroupContext_[n])
                     |    = handshake_secret
                     |
                     +--> Derive-Secret(., "app", GroupContext_[n])
                     |    = application_secret
                     |
                     +--> Derive-Secret(., "confirm", GroupContext_[n])
                     |    = confirmation_key
                     |
                     V
               Derive-Secret(., "init", GroupContext_[n])
                     |
                     V
               init_secret_[n]
```

Epoch-Level Authentication
Group-Level Authentication



For Group
Membership Auth

Group Auth
Key

ABCDEFG

ABCD

EFG

AB

CD

EF

A

B

C

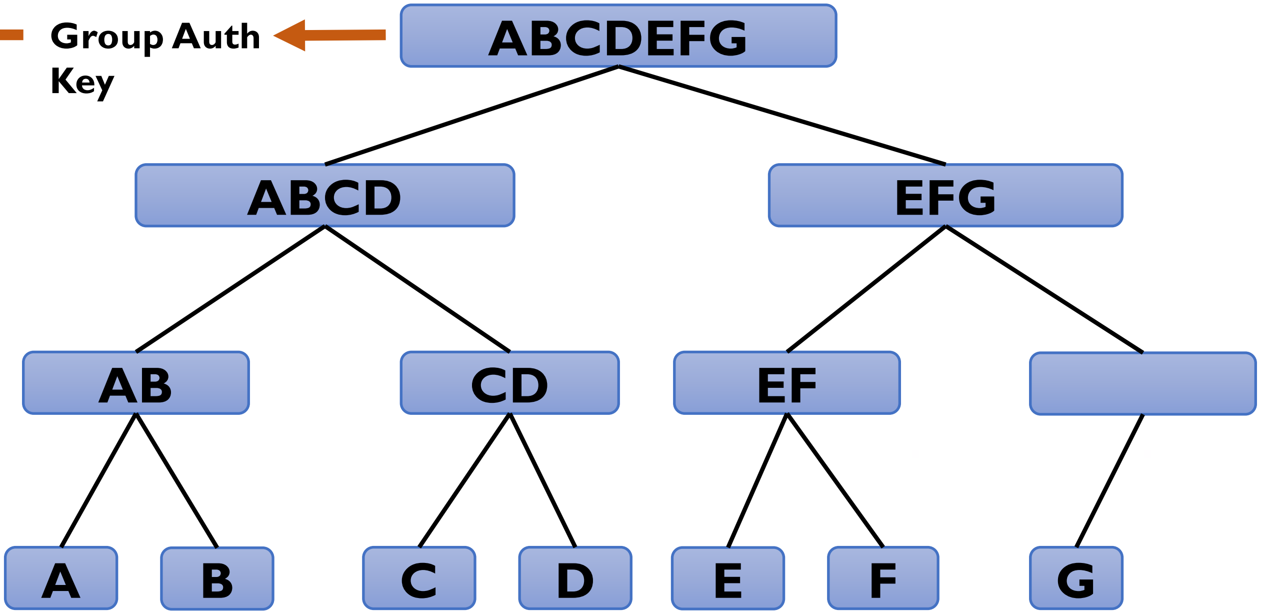
D

E

F

G

For Group Membership Auth ← Group Auth Key ←



For Entity Auth? ← SigKeyA

Entity Auth breaks under static long-term SigKey