rfc4474bis-01

IETF 90 (Toronto)

STIR WG

Jon

First principles (yet again)

Separating the work into two buckets:

1) **Signaling**
   - What fields are signed, signer/verifier behavior, canonicalization
2) **Credentials**
   - How signers enroll, how verifiers acquire credentials, how to determine a credential's authority for identity

- rfc4474bis (now a WG item!) is about (1)
  - But contains guidance for future specifications of (2)

Recap

- Identity signature over To, From, Method, and Date
- The mechanism works for TNs, could also work for SIP URIs
  - Would need to specify credential systems for greenfield IDs
- Optional Identity-Reliance header
  - Optional for signer to add, optional for verifier to check if present
- Identity-Info now much broader than in RFC4474
  - Acts as a selector if multiple parties can sign for the name
  - Not just for certificates per RFC4474
- Canonicalization (now not just a stub)
- Keeps much of the original RFC4474 apparatus
  - All the response codes, etc.

Canonicalization

- The high-level procedure (for To and From headers)
  - Strip special characters, append a country code if missing
  - End up with a format like:
    - 17004561000 (strip any +)
  - What if the E.164 format can't be inferred (at either side)?
    - Two possible options:
      - Guess that it's from this nation and append a cc, if the call is international, it fails
      - Leave it without a country code…?
  - What about special numbers?
    - Especially if we're canonicalizing To as well
    - Short codes, emergency codes, many corner cases
      - **Characters # and *, tones A B C D**

Canonicalization (2)

- rfc4474bis-01 adds a new Identity-Info param
  - “canon” with a value of tn-spec
  - Stores the canonical form of the TN created by the signer
    - Right now, actually vague about whether this is the To or From header field value – implied From
    - Should it include the To?
- Today, this is not under the signature
  - Why? Because intermediaries might change it
  - Should we protect it?

Baiting Attacks

- Raised on the list: REFER baiting
  - I REFER you to send a call through me (evil) to a chosen number
    - e.g. +17004561000@evil.com
  - I then copypasta the token into my own INVITE to that number with my media params
    - To +17004561000@gateway.com
  - I can thus impersonate you
- Arises due to several causes:
  - Because we don't protect SDP and thus media (invariant)
  - Because we limit TN signature scope to the TN only, not the domain
  - Because we lack a secure indication that a REFER induced this token

Fixing the baiting attack

- Fundamental SIP "perversion" as underlying cause
  - But we can't fix core SIP routing
  - Actually RFC3261 allows arbitrary location service decisions
- We could restrict REFER in some way
- We could protect media
  - Some kind of partial signature, even
- We could add a signed indication that a REFER induced this
  - Recipients can at least then decide to
- Other thoughts?

Open Issue: STIR scope

- REFER baiting is one of several attacks on the edges of our scope
- Should we protect mid-dialog requests?
  - Otherwise, forged BYEs can take down calls
  - It’s impersonation, but it isn’t robocalling or swatting
- Do we envision a later document explaining how to apply rfc4474bis to mid-dialog requests
  - Possibly outside STIR, in a successor?
  - Might explore connected identity revisions, even

Open Issue: Credential caching

- Should Identity-Info contain a credential hash
  - Let verifiers know that they already hold the credential
    - Remember multiple credentials might sign for the same number
      - So verifiers can't just tell from the From canonicalization
    - Some other form of UID for the credential also possible
  - Potentially complex interaction with caching
    - Verifiers can't assume the credential is still valid, so a lookup of some kind is still necessary
- But is there some value as an optimization?

Open Issue: Partial Body Protection

- Should we create a protection for subsets of SDP bodies
  - For example, signature over only hash of keying material for SRTP
    - Necessarily optional, since RTP doesn't require SRTP
    - Shouldn't be vulnerable to a bid down
  - Possibly other elements could be included as well
    - See earlier discussion of REFER baiting
- Will SBCs still violate that signature?
  - And if so, is that violation bad enough that verifiers should fail on it?
- Per STRINT, should something like this be mandatory?

Path Forward

- Editing still needed, big ideas now in place?
  - Some legacy RFC4474 language could use an update