Cryptoki Authentication Models, v2.11? and v3.0

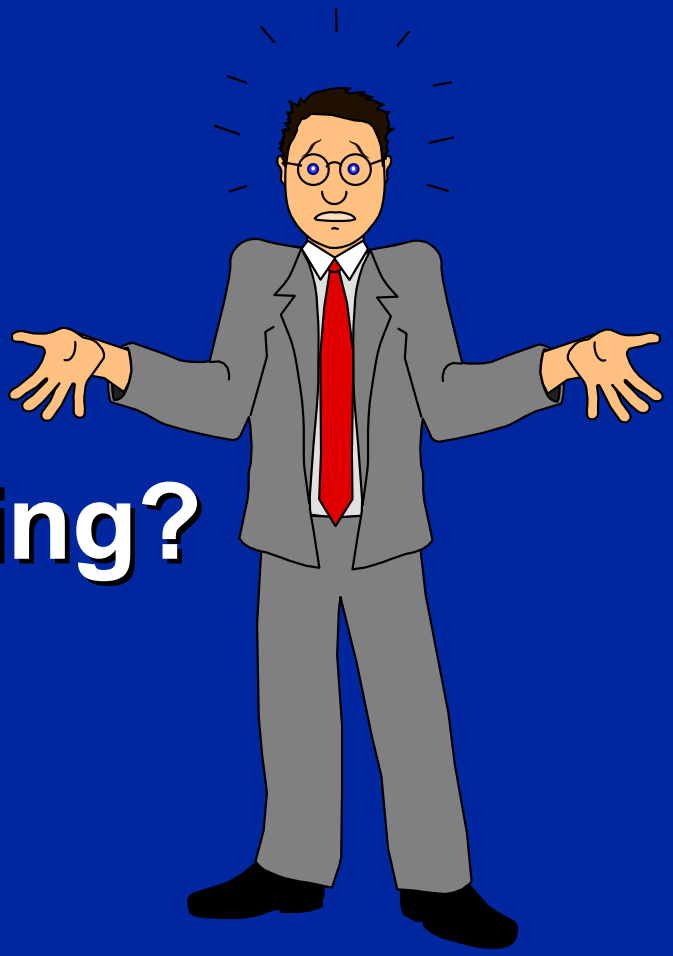
Matt Wood

Software Architect

Intel Corporation

Access Control in v2.10

- User/SO login
  - Public/private objects
- Secondary authentication



What's Missing?

What's Missing in v2.10?

1. Plausible method for a multiple PIN authentication mechanism
2. Complete support for existing PKCS #15
3. Any support for authentication mechanisms other than PIN (without protected *PIN* path)

Can the Gaps be Filled?

- Without breaking backward compatibility?
  - Maybe… but probably not
- Can support for a rich set of authentication mechanisms be supported?
  - Maybe… but probably not
- Should we spend a lot of time advancing the v2.x spec?
  - Maybe… but probably not

Proposed v3.0 Model

- Based on combination of CDSA and other models
- Assigns an ACL to each resource
- Authentication is specified using authentication objects
  - PINs
  - Biometrics
  - Others…

ACLs

- Control the access policy of an object
- Contain multiple entries with the following
  - Authorization list
  - Restrictions
  - Authentication mechanism
- Authorize an action if an entry exists that has a matching authentication mechanism and authorization, within the confines of the restrictions

Authorization Lists

- Contain object type specific actions
- Examples
  - Private keys
    - Sign, Decrypt, etc.
  - Data Objects
    - Read, Write, Execute, etc.

Restrictions

- Time based
- Usage based
- Etc.

Authentication Mechanisms

- PIN
- Biometric
- Public key/Certificate
- Threshold
  - *K* of *N*
  - Can be used as a grouping function (1 of *N*)

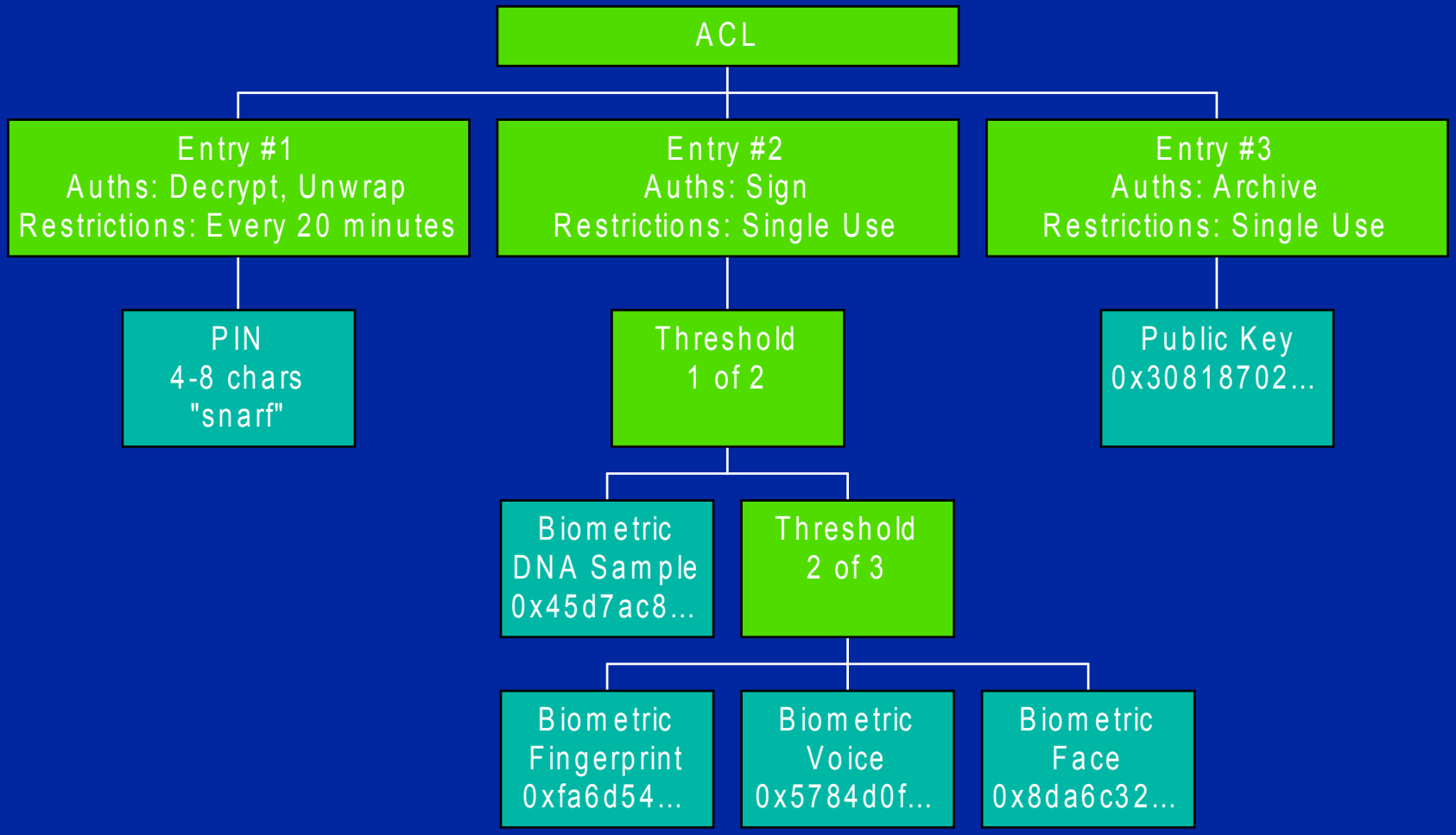
Simple Private Key ACL

- Single entry
  - Authorizations: Sign, Decrypt, Unwrap
  - Restrictions: None
  - Authentication: PIN{4-12 chars} = “snarf”

Not So Simple Private Key ACL

- Entry #1
  - Authorizations: Decrypt, Unwrap
  - Restrictions: Authenticate every 20 minutes
  - Authentication: PIN{4-8 chars} = “snarf”
- Entry #2
  - Authorizations: Sign
  - Restrictions: Single use
  - Authentication: Threshold{1 of 2}
    - Biometric{DNA Sample} = 0x45d7ac8…
    - Threshold{2 of 3}
      - Biometric{fingerprint} = 0xfa6d54…
      - Biometric{voice} = 0x5784d0f…
      - Biometric{face} = 0x8da6c32…
- Entry #3
  - Authorizations: Archive
  - Restrictions: Single use
  - Authentication: Public key = 0x30818702…

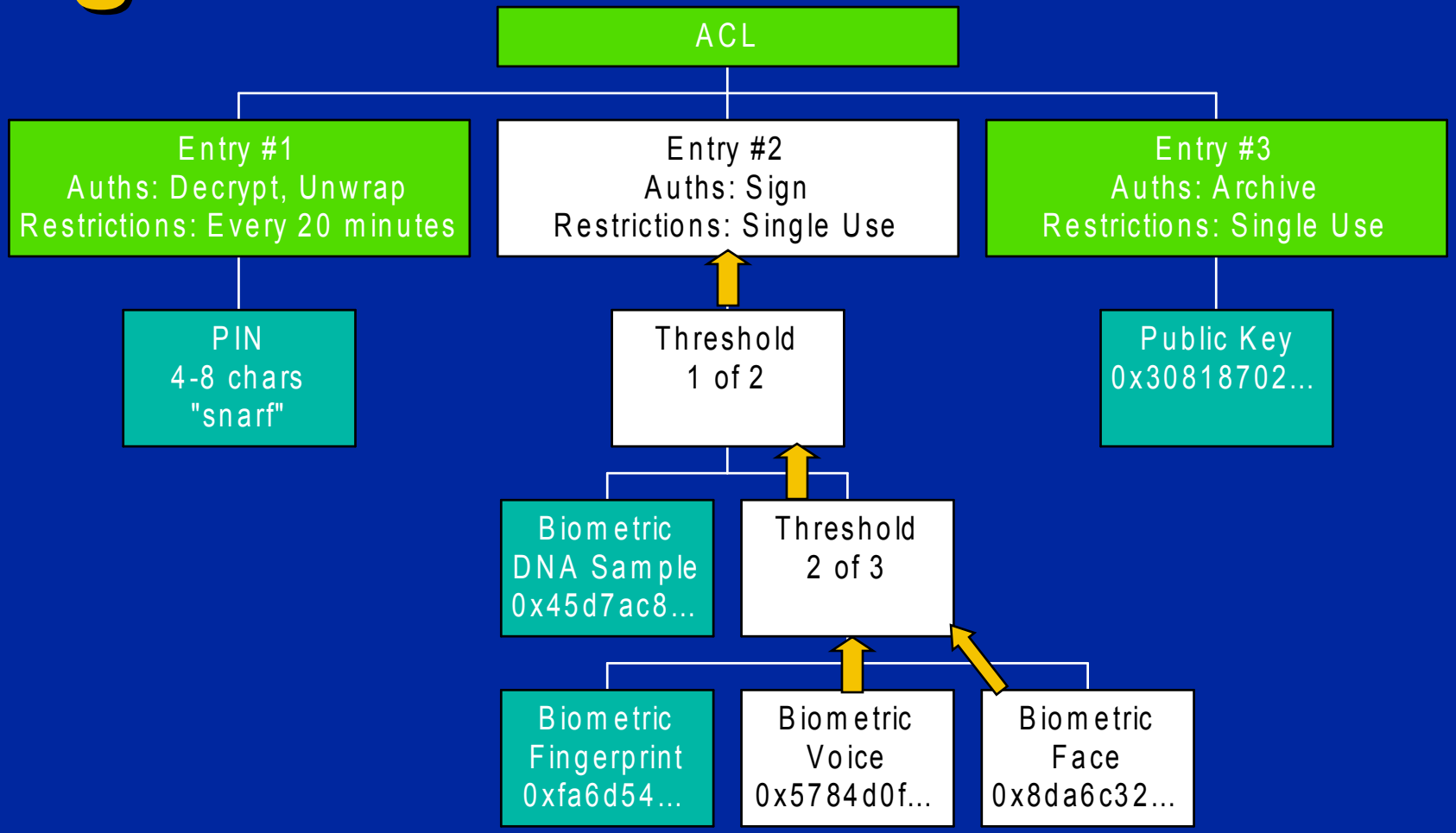
ACL Diagram


ACL
Entry #1
Auths: Decrypt, Unwrap
Restrictions: Every 20 minutes
Entry #2
Auths: Sign
Restrictions: Single Use
Entry #3
Auths: Archive
Restrictions: Single Use
PIN
4-8 chars
"snarf"
Threshold
1 of 2
Public Key
0x30818702...
Biometric
DNA Sample
0x45d7ac8...
Threshold
2 of 3
Biometric
Fingerprint
0xfa6d54...
Biometric
Voice
0x5784d0f...
Biometric
Face
0x8da6c32...

How Do You Assert an Authorization?

- Recursive assertion of nodes in the authentication tree to get authentication handles
- Handles expire once the restriction has been met

Recursive ACL Assertion for Sign



Question…

- How should the authorizations be applied to actions?
  a. Set an attribute of the session used to perform the action
     - May require the authorization handle attribute to be set for every operation
  b. Pass the authentication handles to APIs that require them
     - All access controlled APIs must have an added parameter

Previous Answers

- At the April workshop, the first method was suggested
- CDSA took the second option

Discussion