Hidden Credentials

Paper by
Jason E. Holt, Robert W. Bradshaw, Kent E. Seamons, and Hilarie Orman

Presented by
Nicholas Farnan
9-30-09
Identity-based Encryption

Thanks to Yaronf (via Wikipedia) for the graphic
Credential Indistinguishability

- $P$ and $P'$ are possible credential policies
- $P \neq P'$
- $\text{CT} = \text{HC}_\text{simpleE}(R, \text{nym}, P)$
- $\text{CT}' = \text{HC}_\text{simpleE}(R, \text{nym}, P')$
- To anyone without $P$ or $P'$, CT and CT' must be indistinguishable
- Without the required credential, an malicious party cannot learn what credential is required to decrypt a message
FullIdent

- Boneh and Franklin 2001
- Cryptosystem chosen to provide the backbone of Hidden Credentials
- Note that any Identity-based Encryption scheme that has Credential Indistinguishability can be used
- Has already been shown to be secure against adaptive chosen ciphertext attacks (CCA2)
Mapping Hidden Credentials to FullIdent

- **CA_Create()**: Run by the "CA" to establish itself as, well, a CA

- **CA_Issue(nym, attribute)**: Issues a credential (private key) to nym stating that nym does, indeed, have the property attribute
Mapping Hidden Credentials to FullIdent

- $HC_{\text{simpleE}} (R, \text{nym}, P)$:
  - $R =$ the request for a resource, or the resource
  - $\text{nym} =$ the pseudonym of the user who should be allowed to access $R$
  - $P = \{\text{attribute},\ \text{PKG}_p\}$
  - $= \text{FullIdent}_E (\text{params} \ U \ \text{PKG}_p, \text{nym}||\text{attribute}, R)$

- The ID||attribute construct was suggested in FullIdent to implement key expiration
  - How could this be done?
How much are the PKG's required to share?

- params = \(<q, G_1, G_2, \hat{e}, n, P, H_1, H_2, H_3, H_4>\)

- params is stated to be agreed upon for use by all PKGs

- However, PKG\textsubscript{pub} = PKG\textsubscript{sec} P, and PKG\textsubscript{pub} is unioned with params in the mapping of HC\textsubscript{simple E} to FullIdent\textsubscript{E}

- PKGs all have a different P, but can they opt to use different versions of other parameters in params?
Mapping Hidden Credentials to FullIdent

- $HC_{\text{simpleD}} (CT, \text{Cred})$:
  - CT is the cyphertext from a past $HC_{\text{simpleE}}$
  - Cred is the credential the user agent is attempting to use to decrypt CT
- $= \text{FullIdent}_{D} (\text{params U PKG}_{pb}, CT, \text{Cred})$
Toward More Complex Policies

- If $P$ is a single credential:
  - $H_{E}(R, P) = H_{\text{simpleE}}(R, P)$

- What if $P$ is more than a single credential?
  - $H_{E}(R, P_1|P_2) = \{H_{\text{simpleE}}(R, P_1), H_{\text{simpleE}}(R, P_2)\}$
  - $H_{E}(R, P_1&P_2) = H_{\text{simpleE}}(H_{\text{simpleE}}(R, P_2), P_1)$

- Credential Indistinguishability protects each of the $P$s from being identified, but an attack may be able to see these policies as “$X$ or $Y$” or “$X$ and $Y$”, where $X$ and $Y$ are unknown credential requirements
  - Does this leak uncomfortable amounts of information?
Toward More Complex Policies

- What about everyone's favorite challenge: threshold policies?
  - $HC_E(R, MofN(m, P_1,...,P_n)) = \text{MofN}(R, m, n, HC_{\text{simpleE}}(s_1, P_1),...,HC_{\text{simpleE}}(s_n, P_n))$

- This, however, all depends on the security of the MofN to perform threshold encryption.
Example

- \( \text{CredAlice} = \{C2, C6, C7, C9\} \)
- \( \text{CredBob} = \{C1, C5, C7\} \)
- \( \text{PRequest} = [C1 \& (C2^* \mid C5)] \)
- \( \text{PResource} = [C6 \& C9] \)
- \( \text{PC5} = [C7^* \mid C8] \)

- Should Bob have added PC5 to PResource when the resource was sent?
Implementation and Performance

- Client proxy (Hidden Credential user agent)
- HC-aware HTTP server or server-side proxy
- Identity-based Encryption written in C
- Everything else in Java
- Tested on an 867Mhz G4 under OSX 10.2.6
Implementation and Performance

- 90-95% of time spent in IBE operations
- Single credentials policies
  - Single IBE encryption/decryption: 200ms
  - 450ms to decrypt request, encrypt response
- 6 credentials policies
  - 2500ms to decrypt request, encrypt response
- For a user with 6 credentials
  - 750ms to decrypt CT with 1 sensitive single credential policy
  - 1600ms for 2
Strengths

- Users can access protected resources without revealing that they have the required credentials
- Underlying encryption/decryption scheme can be swapped out for another IDB with Credential Indistinguishibility (threshold scheme is also swappable)
- Performance results
- GPL'd : )
Weaknesses

- If a server wants to accept encrypted requests, it must be willing to divulge that possesses the credentials required to read the request.
- Nyms must be a fixed length.
  - Why would HC be insecure without this?
- Rather hacked proposed implementation of greater than operator.