Domain Name Assertions (DNA)

Joe Hildebrand

Problem

- Hosting providers can’t hold customer certs
  - –Too much responsibility
  - –Not allowed by customers
- Too many connections between servers
  - –Two for each domain pair
  - –E.g.: 10k domains each side = 200 million sockets

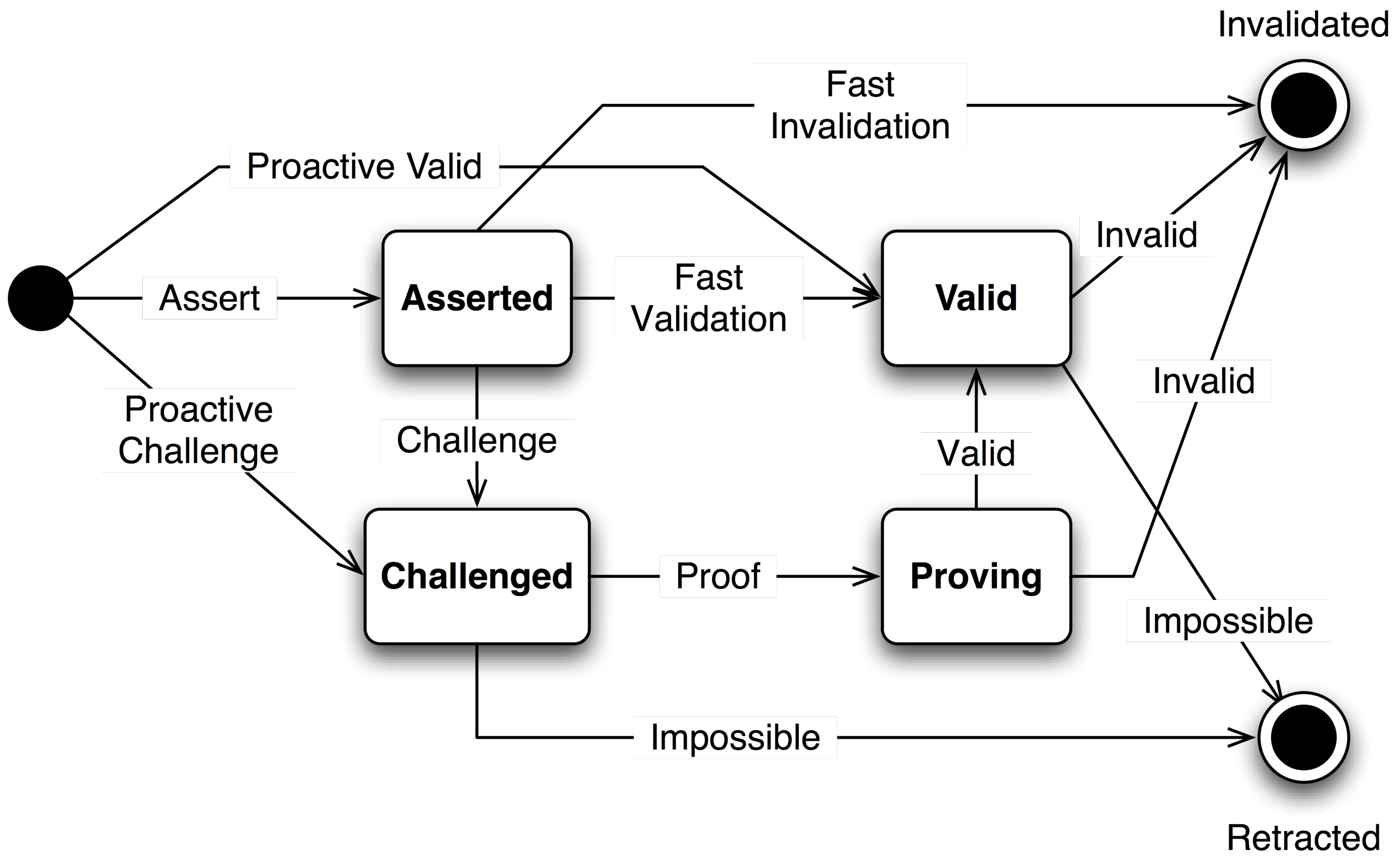
Approach

- Assert domain names
  - –OUTSIDE start-TLS
  - –At the application level
- Verify domains with extensible proof
  - –One such proof: Attribute Certificates (RFC 3281)
  - –Others (such as SAML) can be added later
  - –Custom assertions possible

Server-to-Server example

```
T: <stream:stream from='target.tld' to='originator.tld'>

T: <stream:features>
     <assert xmlns='urn:xmpp:dna:0' from='target.tld'/>
   </stream:features>

O: <challenge xmlns='urn:xmpp:dna:0' to='target.tld'>
     <proof type='urn:xmpp:dna:proof:attribute-cert'/>
   </challenge>

T: <proof xmlns='urn:xmpp:dna:0' from='target.tld'>
       ascii-armored attribute certificate
   </proof>

O: <valid xmlns='urn:xmpp:dna:0' to='target.tld'/>
O: <assert xmlns='urn:xmpp:dna:0' from='originator.tld'/>
...
```

State Transitions



HTTPS Proof?

- Proof URL like: https://target.tld/delegate-xmpp.xml
- Serve up a doc with delegation
- Check domain of cert offered by HTTPS according to XMPP rules (with “www.”+target.tld option)
- Deployable ✔
- Is this different than OAuth?

OAuth Proof

- Domain owner: User
- Asserting entity: Consumer
- Validating entity: Service Provider

Client-to-server

- Same problem as S2S, but easier
  - –One domain
  - –No modifications
- Client suspends judgment on certificate names
  - –Looks for assertion in stream:features

Other protocols

- Could be used for SMTP, IMAP, etc.
- Each needs its own syntax (as for SASL)
- States, proof types stay the same