High-Assurance Re-Direction (The HARD problem)

Richard Barnes

Peter St. Andre

A Common Design Pattern

- a.com outsources services to b.com
- User directs client to connect to a.com
- Client gets redirect to b.com and connects

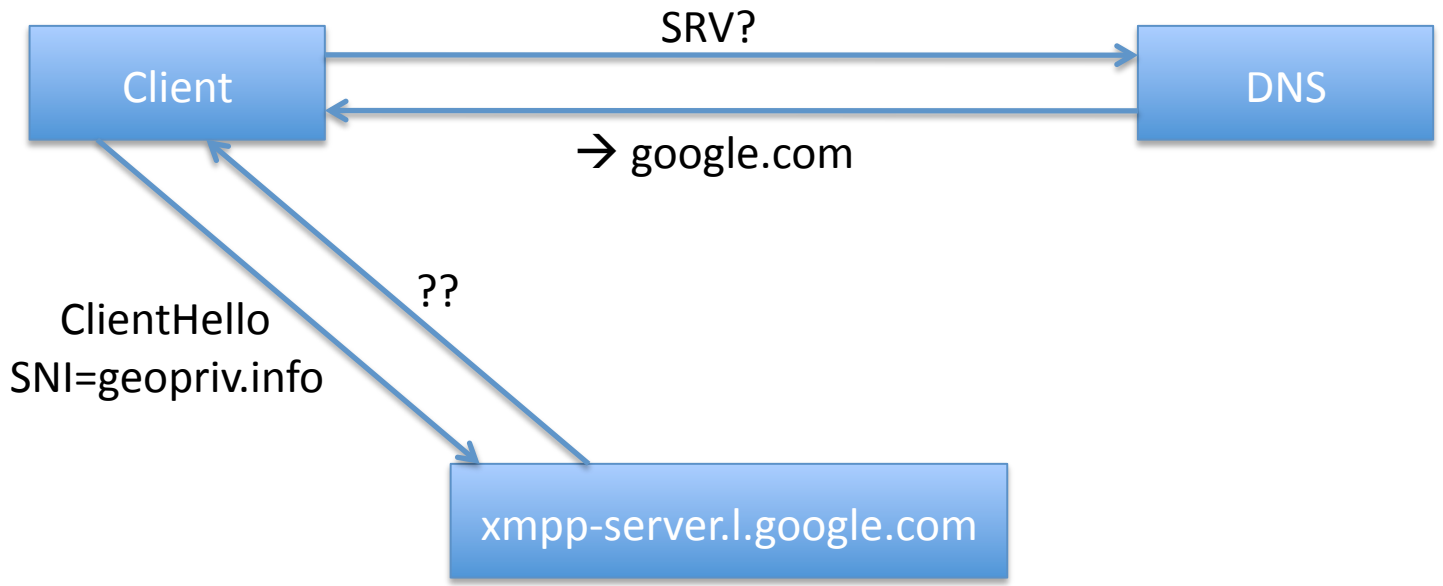
- How does the client know he's talking to the right server for a.com?

Scope

- Many different applications
  - HTTP, SMTP/IMAP/POP3, SIP, XMPP
  - LoST, HELD, ALTO?
- Different embodiments of a redirect
  - DNS-layer: CNAME, MX, SRV, NAPTR, etc.
  - Application-layer: 3XX responses (for example)

In XMPP, for example…

```
_xmpp_server._tcp.geopriv.info.   14400 IN SRV  5 0 5269
   xmpp-server.l.google.com
xmpp-server.l.google.com. 300 IN A   74.125.45.125
```

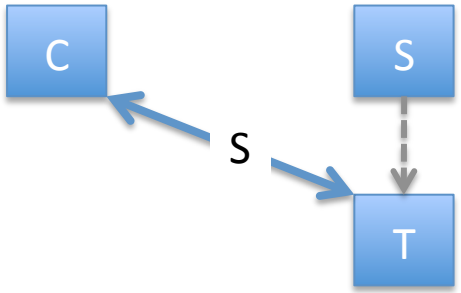


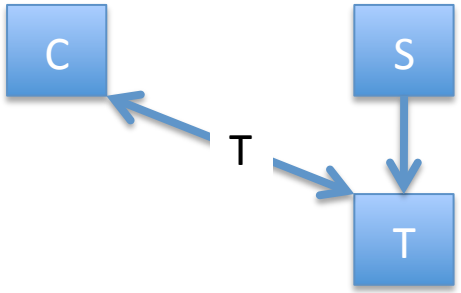
Terminology

- Source domain (a.com)
  - The domain the user sees
  - The domain that is delegated to the service provider
- Target domain (b.com)
  - The domain providing the service
  - The domain to which the service is delegated

Redirected Authentication Options

1. Ignore the redirect
   - Client expects the server to authenticate as the source domain (a.com)
   - Server authenticates as source domain
2. Authenticate the redirect
   - Source domain signs the redirect
   - Server authenticates as the target domain
   - Client validates the signature and expects the server to authenticate as the target domain





Ignoring the Redirect

- State of the art for DNS-based redirection
  - Follow the MX / SRV record
- Security issues
  - a.com might not trust b.com with credentials to authenticate as a.com
  - b.com might not want to have to protect customer credentials
- Operational issues
  - b.com needs to choose which identity to present (requires TLS SNI or equivalent)
  - Inter-provider connections go as $O(n^2)$

Authenticating the Redirect

- Basic requirements:
  - Source domain (a.com) makes a signed statement of the redirect
  - Client can locate and validate this statement
- Nice to have:
  - Generality to address multiple applications
  - Simple for source domain to provision
- Different for application-layer vs. DNS redirect

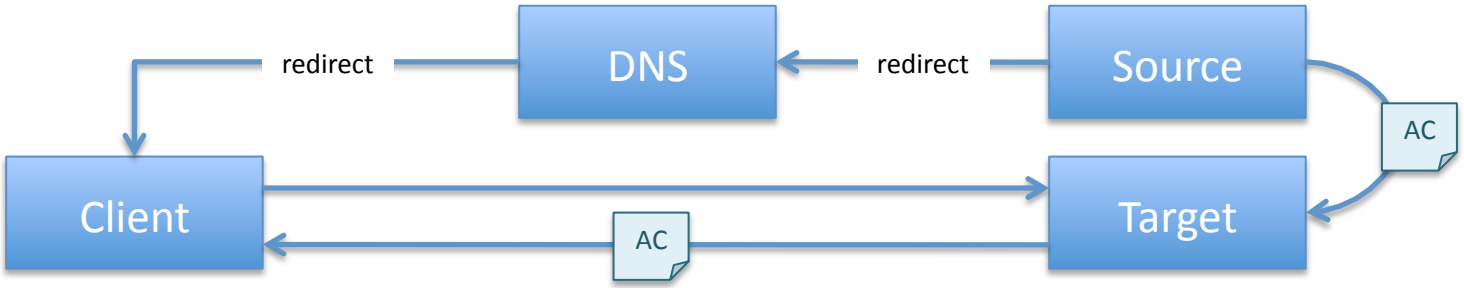
Application-Layer Redirection

- Client has an application-layer interaction with the source domain (a.com) that redirects it to the target domain (b.com)
  - E.g., HTTP 301 over TLS
- Application mechanisms to authenticate source domain, thus the redirect
- Drawbacks:
  - Each application has a separate mechanism
  - Source domain can't completely offload service

DNS-Based Redirection

- Client gets redirect information from DNS, without connecting to source domain at the application layer
- DNSSEC solves this case, since client can verify that redirect record is signed by source domain
- XMPP discussions on interim mechanisms (Domain Name Assertions – DNA)
  - Attribute certificates [draft-ietf-xmpp-dna]
  - External DNSSEC trust anchors [draft-barnes-xmpp-dna]
  - A couple more on the XMPP list

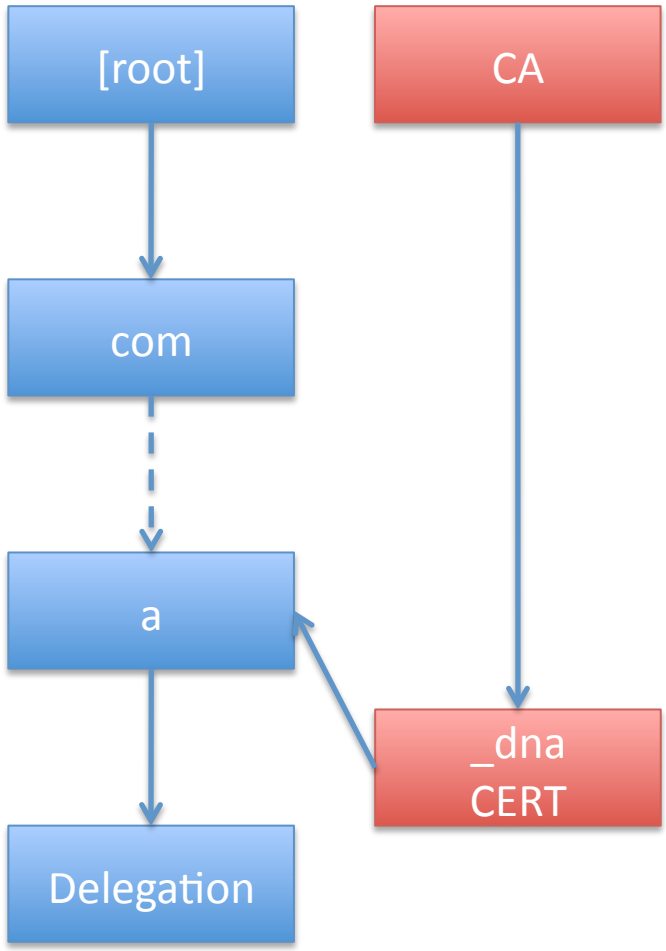
Attribute Cert DNA



- Source domain signs an attribute cert authorizing target domain to provide XMPP services
- Client connects directly to target domain
- Target domain provides attribute cert as proof of authorization
- Costs:
  - Source domain has to sign an attribute cert
  - Requires application-layer protocol changes

Local DNSSEC

- Much of the barrier to DNSSEC deployment is the lack of chains to the root
- Use DNSSEC locally:
  - Sign the delegation zone with DNSSEC
  - Bind the key to the parent domain with an X.509 DV cert
- Very general across applications ….
- … but requires specialized DNSSEC signing and validation



Summary & Questions

- More and more applications are facing the HARD problem
- Applications need to think about how to secure their redirects
- A generalized solution might lead to more consistent behavior and enhanced security