TLS 1.2 Status

Eric Rescorla
Network Resonance
ekr@networkresonance.com
DH Group/Exponent Checking

- Basic issue: TLS uses arbitrary DH groups
  - Chosen by the server
  - What if the server chooses an unsafe group?
- Proposed solution
  - Client SHOULD verify DH group correctness and modulus size
    * Need a reference for how to check. Help!
  - Server MAY use a known group
    * But no hint that this is a known group
DigestInfo

- We now use DigestInfo with RSA signatures
- How is Parameters in AlgorithmIdentifier encoded?
  - MUST be NULL
  - MUST be accepted as empty

- Outcome of a decision in Prague:
Cert Hash Types

• Some comments on the list…
• Generalized to signature hash types
• Added a preference order
• Any fatal alert MUST be sent
  – Used to be optional

• Error alerts sent before closure MUST be fatal

• But alerts don’t have to be sent

Whenever an implementation encounters a condition which is defined as a fatal alert, it MUST send the appropriate alert prior to closing the connection. In cases where an implementation chooses to send an alert which MAY be a warning alert but intends to close the connection immediately afterwards, it MUST send that alert at the fatal alert level.

If an alert with a level of warning is sent and received, generally the connection can continue normally. If the receiving party decides not to proceed with the connection (e.g., after having received a no renegotiation alert that it is not willing to accept), it SHOULD send a fatal alert to terminate the connection.
Signature Hash Agility (I)

- Need a hash indicator
- Also needs to be indicated in cert
- Proposed new struct

```c
struct {
    select (SignatureAlgorithm) {
        case anonymous: struct { };  
        case rsa:
            HashType digest_algorithm;   // NEW  
            digitally-signed struct {  
                opaque hash[Hash.length];
            };
        case dsa:
            HashType digest_algorithm;   // NEW  
            digitally-signed struct {   
                opaque sha_hash[20];
            };
    }
} Signature;
```
Signature Hash Agility (II)

• Pasi suggests that `digest_algorithm` be an `AlgorithmIdentifier`
  – Allows carrying parameters

• This isn’t the ordinary TLS style
  – But we may need parameters

• Proposal: pack into the signature if required