CT for Binary Codes
draft-zhang-trans-ct-binary-codes-00

Dacheng Zhang

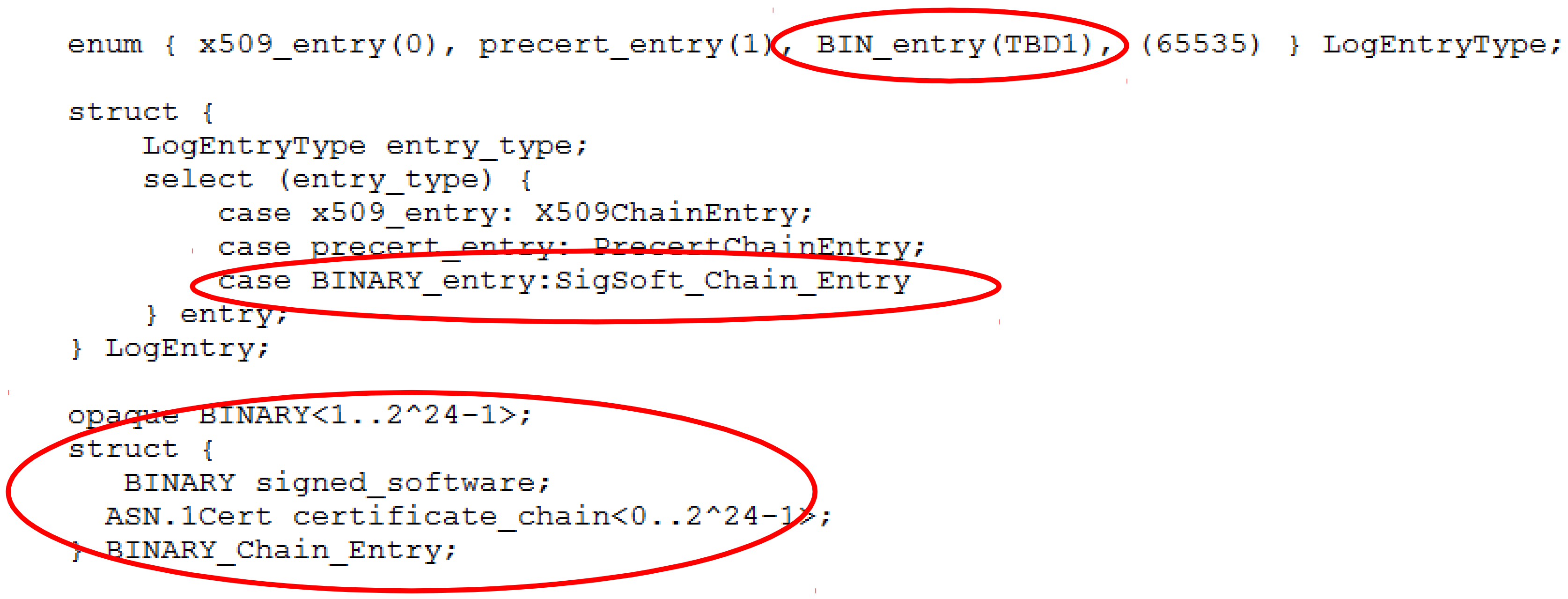
Huawei

IETF91

Motivations

- Digital signatures have been widely used in so ftware distributions to demonstrate the authe nticity of software.
- However, signatures cannot prevent a softwar e developer from distributing software with cu stomized backdoors/drawbacks.
  - In some circumstances, it may be hard for a user t o detect the differences between the software it g ot and the software provided to other users.

Extensions to Log Entries



```
enum { x509_entry(0), precert_entry(1), BIN_entry(TBD1), (65535) } LogEntryType;

struct {
    LogEntryType entry_type;
    select (entry_type) {
        case x509_entry: X509ChainEntry;
        case precert_entry: PrecertChainEntry;
        case BINARY_entry:SigSoft_Chain_Entry
    } entry;
} LogEntry;

opaque BINARY<1..2^24-1>;
struct {
   BINARY signed_software;
  ASN.1Cert certificate_chain<0..2^24-1>;
} BINARY_Chain_Entry;
```

•"signed_software" include the binary codes, the signature, and any other additional information used to describe the software and the signer publishing the software. The way of structuring such information will be left for future work.

•"certificate_chain" include the certificates constructing a chain from the certificate of signer to a certificate trusted by the

Extensions to SCT

```
struct {
    Version sct_version;
    LogID id;
    uint64 timestamp;
    CtExtensions extensions;
    digitally-signed struct {
        Version sct_version;
        SignatureType signature_type = DSRR_timestamp;
        uint64 timestamp;
        LogEntryType entry_type;
        select(entry_type) {
            case x509_entry: ASN.1Cert;
            case precert_entry: PreCert;
            case BINARY_entry: Binary_Codes;
        } signed_entry;
       CtExtensions extensions;
    };
} SignedCertificateTimestamp;
```

```
opaque digestcodes<0..2^24-1>;
struct {
   opaque issuer_key_hash[32];
   digestcodes binary_digest;
 } Binary_Codes;
```

Log Client Messages (1)

- Add Binary and Certificate Chain to Log
  - POST https://<log server>/ct/v1/add-Binary-chain
  - Inputs:
    - software the binary code, the signature, and the inform ation used to describe the software and the signer publi shing the software
    - chain: An array of base64-encoded certificates. The first element is the certificate used to sign the binary codes; the

Log Client Messages (2)

- Outputs:
  - sct_version: The version of the SignedCertificateTimesta mp structure, in decimal. A compliant v1 implementation MUST NOT expect this to be 0 (i.e., v1).
  - id: The log ID, base64 encoded.
  - timestamp: The SCT timestamp, in decimal. extensions: A n opaque type for future expansion. It is likely that not all participants will need to understand data in this field. Log s should set this to the empty string. Clients should decod e the base64-encoded data and include it in the SCT.
  - signature: The SCT signature, base64 encoded.

Open Questions

- The limitation on the size of binary codes
- Should we include the code or just the digest of the code
- Specify the way of present the information in “signed_software”

- Comments?