CT for Binary Codes

draft-zhang-trans-ct-binary-codes-03
Dacheng Zhang
Daniel Kahn Gillmor
Changes since IETF91(1)

• Motivation
  – Add a use case
    • This mechanism can also show some advantages in the scenarios where the signer does not realize that their keys have been compromised. If their update system requires using a CT log they could find out about their compromise.
Changes since IETF91(2)

- Extend the software entry type.

```c
enum { x509_entry(0), precert_entry(1), BIN_entry(TBD1), (65535) } LogEntryType;
enum { binary(TBD3), binary_digest(TBD4) } Signed_Type;

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

opaque BINARY<1..2^24-1>;
struct {
    Signed_Type signed_type;
    BINARY signed_software;
    ASN.1Cert certificate_chain<0..2^24-1>;
} BIN_Chain_Entry;
```
Changes since IETF91(3)

• The software SHOULD encapsulated and signed following the ways specified in CMS[RFC5652]. If signed_type is TBD3, the software is encapsulated in this field. If signed_type is TBD4, the SHA-256 digest of software is encapsulated in this field.

• "certificate_chain"
  – If the information chain is provided in the signed_software field, this field is set to empty.
Unaddressed issue

• Still not specify the information besides the software/digest which should be signed.
  – Vendor name
  – Software name and version number
  – Architecture/Platform/Distribution
  – Data and time when the signature is generated
  – Any more?

• Do we need to support PGP?
Thanks