AN INTERNET OF BLOCK THINGS

Phillip H. Griffin
Griffin Information Security, United States

Abstract *– This paper defines extensible, distributed blocks of hash-linked data constructed using the cryptographic message syntax (CMS) SignedData message. The described SignedData blockchain allows each block to reside in a different physical location on the Internet of things (IoT). Each signed, time-stamped block content can combine data from multiple locations that are 'detached' from and remote to its block header. Two types of SignedData sidechains are described, ephemeral and fixed. Ephemeral sidechains can be added to any block at any time without affecting the integrity of the blockchain. They can also be removed without disruption, making them ideal for use in applications that must manage limited storage capacity or comply with right-to-be-forgotten privacy regulations. A simple blockchain example is presented using CMS SignedData for its block content and headers. This example is then extended to create doubly-linked blockchains and blockchain grids.*

Keywords – ASN.1, blockchain, IoT, sidechain, SignedData

1. INTRODUCTION

A blockchain can be described as a distributed series of signed, hash-linked, append only, timestamped sets of data, grouped into blocks. When viewed as an abstract data type, a blockchain is a limited *stack* implemented as a hash-linked list whose sole operation allows users to *push* blocks onto the top. Users may not modify the data content of any block or *pop* blocks off of the stack without detection, since any changes made to the content or its hash would compromise the integrity of its hash-linked blocks.

The SignedData message data type defined in the cryptographic message syntax (CMS) standard can be used to create extensible, distributed blockchains. CMS is a widely implemented key management standard whose messages are defined using Abstract Syntax Notation One (ASN.1) [1]. ASN.1 is a schema definition language defined in a series of international standards maintained jointly by ISO/IEC and ITU-T [2].

CMS is a mature schema that has been in use for over twenty-five years and employed in a broad range of applications. CMS messages have been standardized as "RSA Public Key Cryptography Standard (PKCS) #7, the Secure Electronic Mail (S/MIME) CMS standard defined by the Internet Engineering Task Force (IETF), and the X9.73 Cryptographic Message Syntax" [3] used in the financial services. A new international version of CMS has been developed in ITU-T Study Group 17 (SG17) and will be published as Recommendation X.894.

The attributes defined later in this paper and those referenced from CMS standards rely on the ASN.1 schema specified in the ITU-T X.500-series of Recommendations (The Directory standards). CMS attributes are compatible with those implemented in many authentication and identity management systems. The SignedData blockchain schema defined in this paper with ASN.1 can be input to tools that generate programming language code. This code can be used to exchange information on a wide range of platforms without consideration of specific programming language, hardware, or operating system characteristics.

2. SIGNED DATA BLOCKS

In a SignedData blockchain, the SignedData CMS message type serves as a container for the two basic components that make up the blocks of a Bitcoin blockchain as illustrated in Fig. 1.

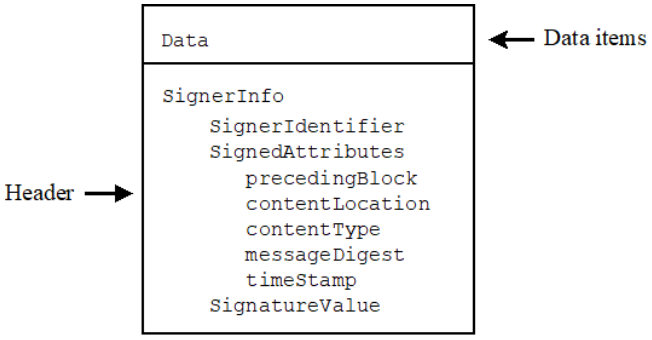

Fig. 1. SignedData blockchain block

These components include data in the form of a "block of items to be timestamped" and a "block header" [4]. The block of items to be timestamped and its associated header can be represented in a SignedData message to create a blockchain block.

In the SignedData type, the "block of items" [4] component is a value of type Data, an opaque string of octets. Type Data can contain information of any type or format. This information may contain flat or structured content, such as a set of transactions in a distributed ledger. However, for the purposes of SignedData message processing, the content is treated as unstructured and its structural details ignored. The "block header" [4] component of a SignedData block is a value of type SignerInfo. A series of block header and associated data components are illustrated in Fig. 2.

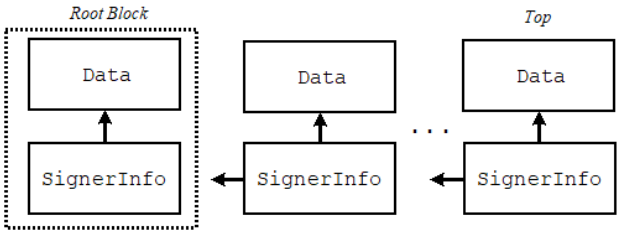

Fig. 2. SignedData block series

The SignerInfo header is a structured data type whose fields include a signing key identifier, a set of attributes to be signed, and the signature of the message signer over the signed attributes. Both the signature algorithm and digest algorithm identifier information are also included in type SignerInfo to provide algorithm independence and promote system resilience in coping with cryptographic change. The data content of a block is signed indirectly, by including a hash of the data in a messageDigest attribute. This required CMS attribute is signed to link the data component of a SignedData block to its SignerInfo header, as indicated by the up arrows in Fig. 2.

3. SIGNED ATTRIBUTES

3.1 Hash pointers

The SignerInfo block header illustrated in Fig. 1 contains a precedingBlock attribute. This attribute is signed to link a SignedData block to the previous block in the blockchain, as indicated by the left arrows in Fig. 2. Adjacent blocks in the blockchain are 'hash-linked' using a precedingBlock attribute value. This value contains a hash (message digest) of the SignerInfo header of the previous block in the blockchain.

A precedingBlock attribute is defined as a hash and a location indicator using ASN.1 as follows:

```
precedingBlock ATTRIBUTE ::= {
  WITH SYNTAX  HashPointer
           ID  id-PrecedingBlock
}

HashPointer ::= SEQUENCE {
   hash        DigestedData  OPTIONAL,
   pointers    Pointers  OPTIONAL
} (ALL EXCEPT ({ -- None present -- }))

Pointers ::=
   SEQUENCE SIZE(1..MAX) OF pointer Pointer

Pointer ::= CHOICE {
   uri         URI,
   rfid        RFID,
   gps         GPS,
   address     Address,
   dbRecord    DBRecord,

   ...   -- Expect other pointer types --

}
```

The precedingBlock attribute contains a value of type HashPointer that implements a Hash Pointer abstract data type. Type HashPointer can be used to specify the location of the previous block in a SignedData blockchain and to verify the integrity of the data content at that location. A value of this type contains "a pointer to the place where some information is stored" that is paired with a "cryptographic hash of the information" [5]. The

hash component of type HashPointer contains a value of the CMS message type DigestedData.

This CMS type specifies the message digest algorithm used to calculate the hash on the pointed to object. Type DigestedData also indicates the type of object being hashed is a SignedData block header, a value of CMS type SignerInfo. In a precedingBlock attribute the type of object is indicated by an ASN.1 information object identifier named id-signerInfo. Other types of objects can also be identified in the DigestedData message.

In a precedingBlock attribute, the optional value of type Pointers in type HashPointer may be omitted when adjacent block locations are known. This may be the case when adjacent blocks are not distributed and reside in a common file system or database. More than one type of Pointer value may be used in series to fully qualify the location of a distributed block. The defined set of pointer types is extensible to allow additional types of pointers to be added as needed by an application.

3.2 Detached content

The data content of a SignedData block is optional to include and need not be present in a given message. In a secure electronic mail (email) application that provides signed email messages using the SignedData type, the detached message content and the signature are located in separate parts of the same email message. This allows the content to be displayed for the email recipient even when an email agent cannot process signed email, or when verification of the digital signature on the detached content fails.

In other application contexts, such as the cloud and Internet of things (IoT) environments, the location of detached SignedData content must be identified and made available for verification of the signature on the message content. When the location of detached data content is not known to blockchain participants, a SignedData message signer can include a content location attribute in the signed attributes of the block header. This attribute can be used to locate detached data content distributed in one or more locations. A content location attribute can be defined as a series of one or more uniform resource identifier (URI) [6] values in ASN.1 as follows:

```
contentLocations ATTRIBUTE ::= {
  WITH SYNTAX URIs
           ID id-ContentLocations
}

URIs ::= SEQUENCE SIZE(1..MAX) OF uri URI

URI ::= UTF8String (SIZE(1..MAX))
```

A collection of items are signed by first calculating a message digest over type ContentToBeSigned defined as follows:

```
ContentToBeSigned ::= SEQUENCE
   SIZE(1..MAX)OF content LocatedValue

LocatedValue ::= OCTET STRING
```

Each value in type ContentToBeSigned is the data located by one URI value in the contentLocations attribute. Each URI contains a generalized form of a uniform resource locator (URL).

3.3 SignedData required attributes

CMS requires at least two attributes to be present if any signed attributes are included in a SignedData message. These attributes are defined in CMS and named contentInfo and messageDigest. When SignedData is used as a blockchain block, the contentInfo attribute value is set to indicate the type of content in the block is ordinary data, rather than one of the other cryptographic message types defined in CMS.

The required messageDigest attribute contains the hash of the content. The content is bound to the other signed attributes cryptographically under the digital signature of the signer. There may be any number of additional attributes included in this binding at the signers choice, and these may be of any type or format.

3.4 Blockchain required attributes

In a SignedData blockchain, both the timeStamp attribute and precedingBlock attributes shown in Fig. 1 must be included in the SignerInfo block header. The value of a timeStamp attribute in a given block header may contain a choice of either a locally sourced synchronized time, or a trusted timestamp token. A trusted timestamp is based on a coordinated time source, such as one of those specified in the X9.95 standard [7]. The type of time value used may vary by block, but should meet the requirements agreed to by the blockchain participants. For blocks associated with tagged,

physical objects on the Internet of things (IoT) or other resource constrained enviroments, a local time value may be the only possibility.

A timestamp attribute can be defined as an extensible pair of choice alternatives in ASN.1 as follows:

```
timeStamped ATTRIBUTE ::= {
   WITH SYNTAX TimeStamped ID id-TimeStamped
}

TimeStamped ::= SEQUENCE {
   timeStampValue    TimeStamp,
   timeStampService  URI  OPTIONAL
}

TimeStamp ::= CHOICE {
   timeStampToken  TimeStampToken,
   localTimeStamp  GeneralizedTime,

   ...  -- Expect additional time types --
}
```

A value of type TimeStamped contains two components, a required timeStampValue and an optional timeStampService. The timeStampValue component is a value of type TimeStamp, a choice between an X9.95 [7] trusted timestamp token, and a value from a local time source. Trusted timestamps can provide greater assurance of the validity of the ordering of block content.

The optional timeStampService component of type TimeStamped is a value of type URI. This value indicates the location of a timestamp authority (TSA) that can issue and verify the timestamp token. This component should be omitted when a local time source is used, or when the location of a timestamp service is required but known as a system default.

A TSA ensures that "an independent third party can audit and validate the controls over the use of a time stamp process [7]. Unlike locally sourced time that must be continuously synchronized by blockchain participants, a TSA relies on time sourced from a national measurement institute (NMI) or other "Master Clocks upstream from a TSA that provides time calibration services" [7]. The time source for an NMI is the "Bureau International des Poids et Mesures (BIPM) near Paris, France" which calibrates the NMI clocks used to calibrate a TSA [7].

4. DISTRIBUTED BLOCKS

Blockchain has been described as a promising technique "to create a decentralized, peer-to-peer trust network" [8]. Due to the success of Bitcoin, a blockchain-based cryptocurrency, blockchain has emerged as a distributed platform capable of providing a ledger for payments and other types of transaction data. It is widely believed that blockchain will play an important role in securing cyber physical systems, and that blockchain will be applied to Internet of things (IoT) applications from cloud computing to "home gateway" and "edge computing" envionments [8].

Part of the appeal of blockchain lies in its promise to provide a secure design for supporting a "distributed computing system with high Byzantine fault tolerance", an efficient distributed system that can be operated "without depending on a central authority" [8]. One property used to quantify the scalability of a blockchain architecture is the "decentralization of block production (DBP)" [9]. This term has been defined as " the number of block producers" [9].

A set of SignedData blockchain blocks can be distributed in whole, just as Bitcoin blockchains can be distributed. Additionally, in a SignedData blockchain, each of the blocks can be separately distributed. Each block can be signed by a different signer using a different signing key. The signer of each block can use a different message digest and signature algorithm, perhaps to meet specific industry or regulatory requirements.

Each block signer can include any number or type of signed attributes along with those required. Each distributed block can serve as an independent block producer. A SignedData blockchain can be extended with additional blocks, and each block can spawn a series of sidechains, blockchains associated with a parent block. This capability allows SignedData blockchain users to create and manage flexible blockchain grids.

5. SIDECHAINS

5.1 Defintions

Many definitions of the term 'sidechain' come from descriptions of blockchain architectures used to transfer cryptographic currencies, such as Bitcoin [4]. In their strong federation paper, Dilley,

Poelstra, Wilkins, Piekarska, Gorlick, and Friedenbach describe a sidechain as an interoperable blockchain solution that can be used to decentralize risk and enhance security [10]. In their proposed blockchain-based system, sidechains provide a mechanism for moving "assets to and from other blockchains" [10]. Once an asset is moved to a sidechain, the sidechain participants can manage their own operational environment without affecting any parent blocks.

This splitting of operations "between entities" serves to "limit the damage an attacker can cause" to the overall system [10]. This mechanism also provides additional system agility benefits. Though the "chains are still attached" the sidechain can be used to test new system features and to isolate a set of logical activities "without harming the main network should vulnerabilities arise" [11]. The blocksigners "who sign blocks of transactions on the sidechain" and define "its consensus history" can use a different consensus mechanism and different cryptographic algorithms than those used on the parent blockchain [10].

5.2 Fixed sidechains

One or more fixed sidechains can be added to a new top block of a parent SignedData blockchain as the block is created. Pointers to these fixed sidechains can be included in the parent block header using the signed attribute, sidechains. The sidechains attribute is cryptographically bound to the new top block of the parent blockchain under a digital signature to link sidechains to the parent.

Fixed sidechain pointers cannot be modified or removed from the signed attributes of the parent block header without detection. Each parent block header is a value of type SignerInfo, which includes a precedingBlock attribute and a timestamp. This signed attribute hash-links the parent blocks of the blockchain together, and cannot be altered without loss of blockchain integrity. Each sidechain pointer can locate a new sidechain root block, or point to any block that already exists. Pointers to existing blocks may locate blocks within the parent blockchain, perhaps to associate related information contained in preceding blocks. A sidechain pointer can also point to a distributed block, a block whose physical location differs from that of the parent block.

A sidechains attribute is defined using ASN.1 as follows:

```
sidechains ATTRIBUTE ::= {
  WITH SYNTAX  Sidechains
          ID  id-Sidechains
}

Sidechains ::=
  SEQUENCE SIZE(0..MAX) OF linked Sidechain

Sidechain ::= HashPointer
```

The syntax of a sidechains attribute is a series of values of type Sidechain. A value of type Sidechain links a parent block to a sidechain. This linkage relies on type HashPointer defined in section 3.1. Type Sidechain contains two optional components, named hash and pointers. In a value of type Sidechain, the pointers component must be present. When the message digest of the sidechain block header can be calculated, the hash component of type Sidechain can also be included.

When a new sidechain root block is created, it may be doubly linked to point back to the parent block as shown in Fig. 3.

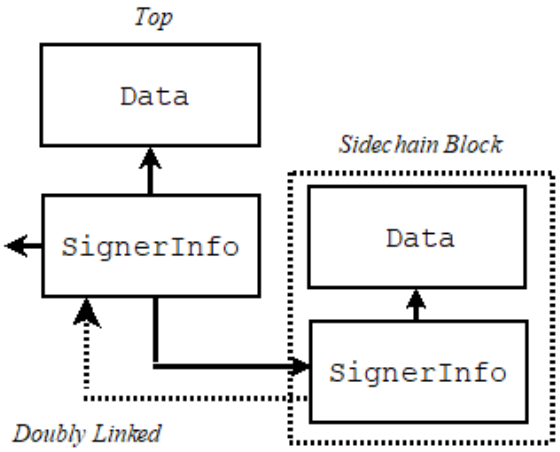

Fig. 3. Fixed SignedData sidechain

To link the sidechain back to the parent block, a parentBlock attribute must be included in the root block of the new sidechain. The parentBlock attribute is similar to the precedingBlock attribute defined in section 3.1. The parentBlock and precedingBlock attributes have different identifier names so that applications can readily distinguish between these two types of links.

Subsequent blocks added to a sidechain root block must include a precedingBlock attribute. This attribute should not be included in the root block

of the sidechain. The absence of a precedingBlock attribute or the presence of a parentBlock attribute indicate that a sidechain root block has been located when traversing the chain of blocks.

The parentBlock attribute is defined using ASN.1 as follows:

```
parentBlock ATTRIBUTE ::= {
  WITH SYNTAX  ParentBlock
           ID  id-ParentBlock
}

ParentBlock ::= HashPointer
```

The syntax of a parentBlock attribute is a value of type ParentBlock. Type ParentBlock relies on type HashPointer defined in section 3.1. Type ParentBlock contains two optional components, named hash and pointers. In a value of type ParentBlock, the pointers component must be present. When the message digest of the parent block header can be calculated, the hash component of type ParentBlock can also be included.

5.3 Ephemeral sidechains

The CMS SignedData message schema can support multiple content signers through a series of SignerInfo values. There is one SignerInfo value for each co-signer of the message content or signed attributes. Each SignerInfo value in the series is independent of all of the others. Each co-signer can choose their own signature and hash algorithms, use their choice of signing key, and include any number of attributes of any type of format that they wish. The CMS standard only requires that the contentType and messageDigest attributes also be included, as described in section 3.4.

The SignedData message serves as a container for the blockchain block header and data components. In a SignedData block only the first SignerInfo value is used as a block header that hash-links the block into the blockchain. The signature on the attributes in the block header serves to cryptographically bind "the contents of the block, a timestamp, and the previous block header" to form a *chain* of data that can provide "a well-defined ordering for transactions" [12]. The SignedData block header and the other SignerInfo values in the series are illustrated in Fig. 4.

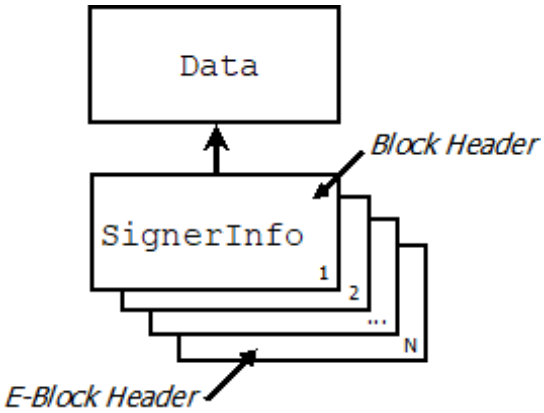

Fig. 4. SignedData SignerInfo series

The e-block header SignerInfo values in the series depicted in Fig. 4 can be used for other purposes. Their use is specified by a SignedData blockchain application. Their use has no affect on the first SignerInfo value, the block header used to create the hash-linked blockchain. The e-block header values can be used to manage a series of ephemeral sidechains.

An e-block header value can be added to any SignedData container at any time, even after the block header hash-links have been signed. They can be removed without loss of blockchain integrity. Though they are part of a SignedData container, they are not connected operationally to the blockchain block header.

E-block headers can be used by an application in a number of ways. They can be used to form a completely new and independent root block of an ephemeral sidechain, a block not connected in any way to the SignedData blockchain block header. This could be accomplished by not including any hash pointer back to the parent SignedData block header in the sidechain e-block header. As an alternative, the parentBlock attribute described in section 5.2 could be included in the e-block header.

SignedData ephemeral sidechains may be retained for as long as needed, but they can be considered as being temporary objects. Ephemeral sidechains can be deleted from a blockchain once they are no longer needed by the application. This feature can help SignedData blockchain applications to comply with some aspects of privacy laws and regulations, such as the *right-to-be-forgotten* requirements of the European Union (EU) General Data Protection Regulation (GDPR).

Article 17 of the GDPR regulation, "Data Erasure", entitles data subjects to the "right to be forgotten" [13]. This privacy right "entitles the data subject to have the data controller erase his/her personal data" [13]. Careful data architecture design, where the privacy sensitive data of an individual is stored in one or more ephemeral sidechains, would make it possible for a SignedData blockchain application to comply with a data erasure request. Compliance would simply require deleting any sidechains containing data subject information.

6. CONCLUSION

In this paper blockchain blocks constructed using the cryptographic message syntax SignedData type were described. These blocks were specified using the ASN.1 schema definition language, a standard for the generation of programming language tools from ASN.1 syntax. ASN.1 provides applications with platform-independent information exchange that can enhace the chances of interworking systems.

Signed attributes required by the CMS standard or needed to implement a Bitcoin style blockchain were described in this paper using ASN.1. Two types of sidechains and their schema were also described, fixed and ephemeral. The paper discussed how fixed sidechain attributes could be cryptographically bound to a parent blockchain block under a digital signature.

Ephemeral sidechains loosely coupled to a parent block using an e-block header SignerInfo value were shown to be operationally disjoint from the block header used to hash-link a parent block to its blockchain. The paper described how ephemeral sidechains could be added to or deleted from a parent block at any time without affecting the data integrity of the parent blockchain. This feature could be used to design blockchain systems that could comply with the requirements of privacy regulations.

ACKNOWLEDGEMENT

This paper could not have been produced without the guidance and influence of Bancroft Scott, who led me to a new world of intellectual opportunities, and John Larmouth, who encouraged me to dream of new things and taught me how to make them real. Special thanks to OSS Nokalva for use of the ASN.1 tools used to perfect the schema available at http://phillipgriffin.com/SignedDataBlocks.asn.

REFERENCES

[1] J.L. Larmouth, “ASN.1 Complete,” San Francisco: Morgan Kaufmann Publishers, 2000. Retrieved March 17, 2018, from http://www.oss.com/asn1/resources/books-whitepapers-pubs/larmouth-asn1-book.pdf

[2] ITU-T X.680, "Information Technology – Abstract Syntax Notation One (ASN.1) Specification of basic notation," 2017.

[3] P.H. Griffin, "Telebiometric Security and Safety Management," Proceedings of ITU Kaleidoscope 2013 Conference – Building Sustainable Communities, Kyoto, Japan, 2013.

[4] S. Nakamoto, "Bitcoin: A Peer-To-Peer Electronic Cash System," 2008.

[5] A. Narayanan, J. Bonneau, E. Felten, A. Miller, & S. Goldfeder, "Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction", Princeton University Press, 2016.

[6] ITU-T X.672, "Information technology – Open systems interconnection – Object identifier resolution system (ORS)".

[7] Accredited Standards Committee (ASC) X9 Financal Services, "American National Standard (ANS) X9.95 Trusted Time Stamp Management and Security", 2016.

[8] Opportunities and Use Cases for Distributed Ledger Technologies in IoT. GSMA (2018). Retrieved September 22, 2018, from https://www.gsma.com/iot/opportunities-and-use-cases-for-distributed-ledger-technologies-in-iot/

[9] K. Samani, "Models for Scaling Trustless Computation," MultiCoin Capital , 2018.

[10] J. Dilley, A. Poelstra, J. Wilkins, M. Piekarska, B. Gorlick, & M. Friedenbach, "Strong Federations: An Interoperable Blockchain Solution to Centralized Third Party Risks," *arXiv preprint arXiv:1612.05491*, 2016. Cornell University Library Cryptography and Security archive.

[11] A. Hertig, "The Sidechains Breakthrough Almost Everyone in Bitcoin Missed," CoinDesk, 2018.

[12] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, ... & P. Wuille, "Enabling blockchain innovations with pegged sidechains," 2014.

[13] EU-GDPR, "European Union (EU) General Data Protection Regulation (GDPR) Portal, http://www.eugdpr.org, 2018.