PERMANENT AUTHENTICATION PROOFS

INVENTOR:

MASSIMILIANO PALA

METHOD FOR PERMANENT AUTHENTICATION PROOFS

Defining authentication proofs for permanent and verifiable data origin

Abstract

This work describes a novel approach to track the origin of transferred data (e.g., configurations, documents, applications, etc.) independently from the producer.

Massimiliano Pala
m.pala@cablelabs.com

TABLE OF CONTENTS

1 Overview

In security, one important parameter for assuring and being able to verify that data, configurations, or applications come from authenticated and trusted origins. Today, we leverage the use of secure protocols such as TLS or SSH to make sure these properties are well verified - the data comes from the right internal/public server and the data has not been altered.

This leaves the user with the hard task of convincing 3rd parties that the data was downloaded from the right server or website. Since the same data could have been downloaded from a different site, the only type of validation that can be done at that point is to use some form of checking that the secure hash (or checksum) value calculated on the transferred data is correct. Very manual process, prone to error, and used only for software. Regrettably, because of the impracticality of the current approach, the origin validation is lost after the data is actually transferred.

1.1 Current Approach and Problem Statement

The problem with this approach is that once the data has been transferred from one entity to the other, the authentication and integrity information is lost. Specifically, under the assumption that we can identify the origin because the data was correctly decrypted and the encryption key was negotiated with a well-known server (i.e., a server whose certificate/key is verified and trusted), there is no possible way to retain that information after the data has been decrypted. In order to preserve that capability, the whole encryption key and negotiation (key-exchange) session would have to be stored (and shared with the receiving party).

This work changes the inability to retain the origin of data/documents/text/configurations /etc. by adding the concept of delivering a permanent authentication for the origin of the data.

2 Document Notation

The symbol (" | ") is used throughout this document to indicate concatenation of two values. Specifically, when indicating the concatenation of values A and B, this document uses the following notation:

{ A | B }

The symbols ("{") and ("}") indicate the beginning and end of a logical record.

3 The Invention Overview

In this invention we recognize the ("Server") and the ("Client") as the two parties involved in a secure communication. The terminology ("Client") and ("Server") will be used to help distinguishing the two parties of a communication (e.g., "Entity A" and "Entity B" can be referenced as "Client" and "Server"). This said, there are no special roles for server or client in

this work as the invention can be applied symmetrically – the only requirement for the patent idea to work is for the party where the data is transferred from (origin) can provide an authentication (e.g., a signature) tied to its own identity and the data that was transferred (e.g., a signature calculated over the transferred data).

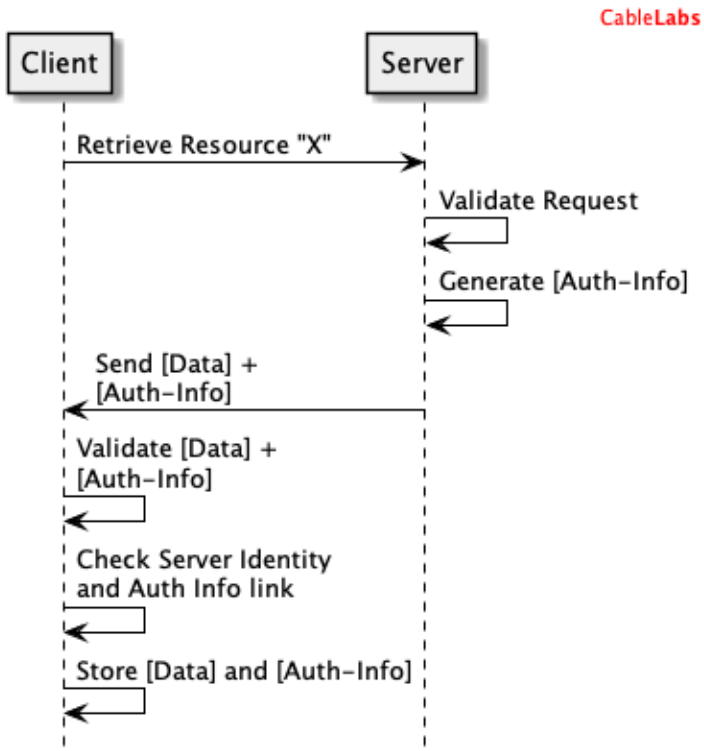
3.1 The Assumptions

In this invention, it is assumed that two parties are connected to each other (e.g., have the possibility to transfer data across each other), not necessarily through a secure protocol such as HTTPS or SSH. In the case of secure protocols, it is assumed that the originating party (i.e., the party where the data is transferred from) can generate the authentication information via its own private key or token.

NOTE WELL: If a non-secure protocol is used, the binding with the origin is only through the additional information provided by the party and it provides weaker security properties (i.e., no network-binding properties such as DNS names can be associated with the additional authentication information because it cannot be validated by the client).

3.2 The Workflow

The workflow is depicted in the next figure:



Specifically, in this work, when the ("Client") requests a specific resource from the ("Server"), the server returns the resource together with the additional source validation information or "Permanent Authentication Proofs" (PAPs).

The PAPs are signed tokens that can be instantiated with different technologies and formats. It is suggested that a standard format is used to facilitate interoperability across systems and environments. An example is the Cryptographic Message Syntax (CMS) format that is suitable for carrying the origin information when a certificate or a key is used as the origin's identity.

After the data and PAP are transferred, they can be used together to provide both origin and integrity information. For example, when CMS is used, the authentication information should have the signerInfo data structure configured for the origin's identity details, together with all the required certificates or keys needed to build the chain to a root source of trust such as a trusted root CA or root key. Additional information such as OCSP responses or CRLs can be also added to the authentication information for proof of validity at the time of transferring the data.

4 PAPs Generation Examples for different Protocols

Although this work is independent from the specific protocol used underneath to transfer the data, it is important to provide some technical aspects for the major protocols in use today across the world.

4.1 HTTP/S PAPs

In the HTTP/S world, when a resource is requested, it is possible to provide the additional authentication information via a multi-part message where the authentication data can be received and saved separately from the data to be authenticated.

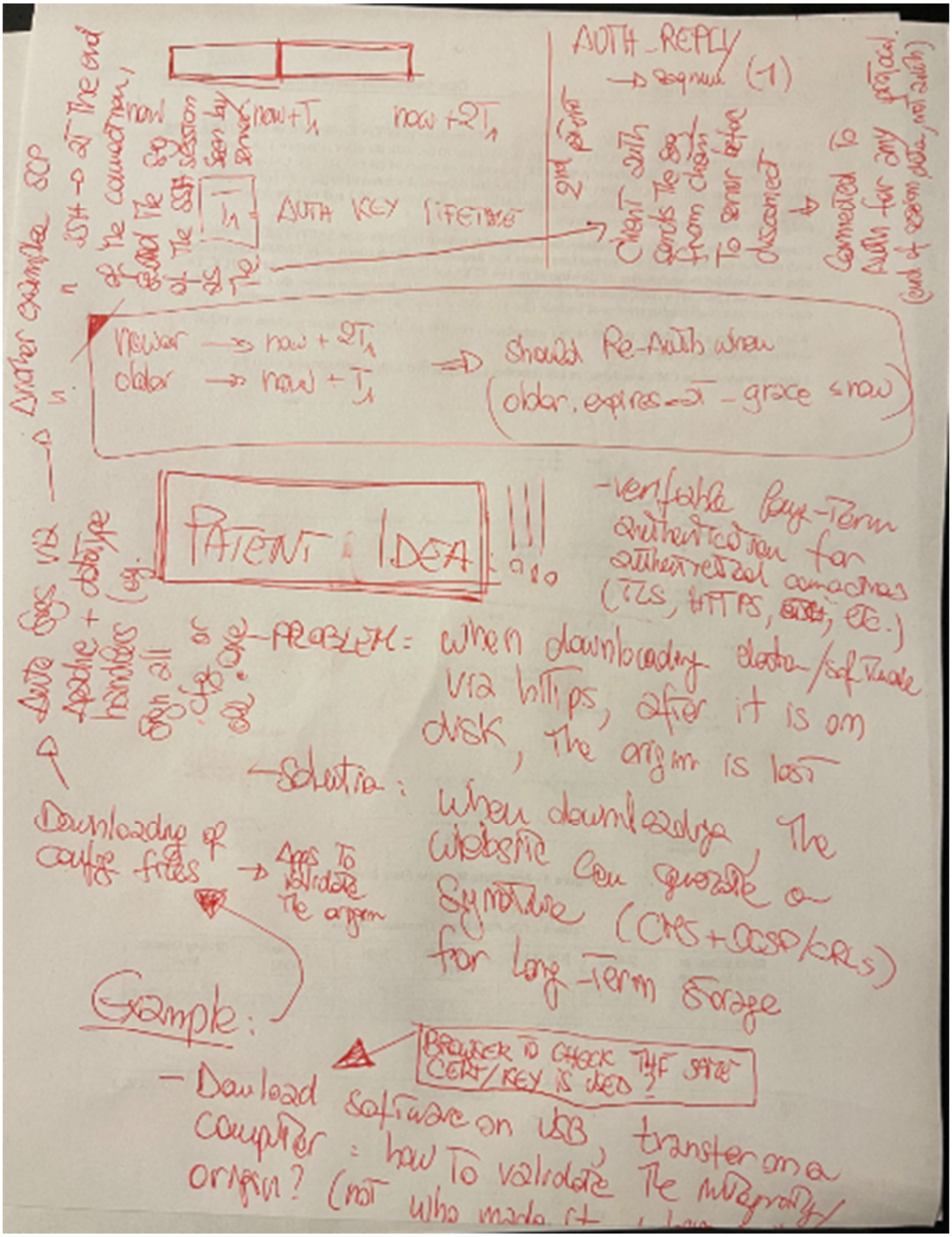
Alternatively, the server can be configured to generate a data+signature single file (e.g., by using a CMS signature where the signed content is actually provided within the data structure itself in a so called non-detached signature)

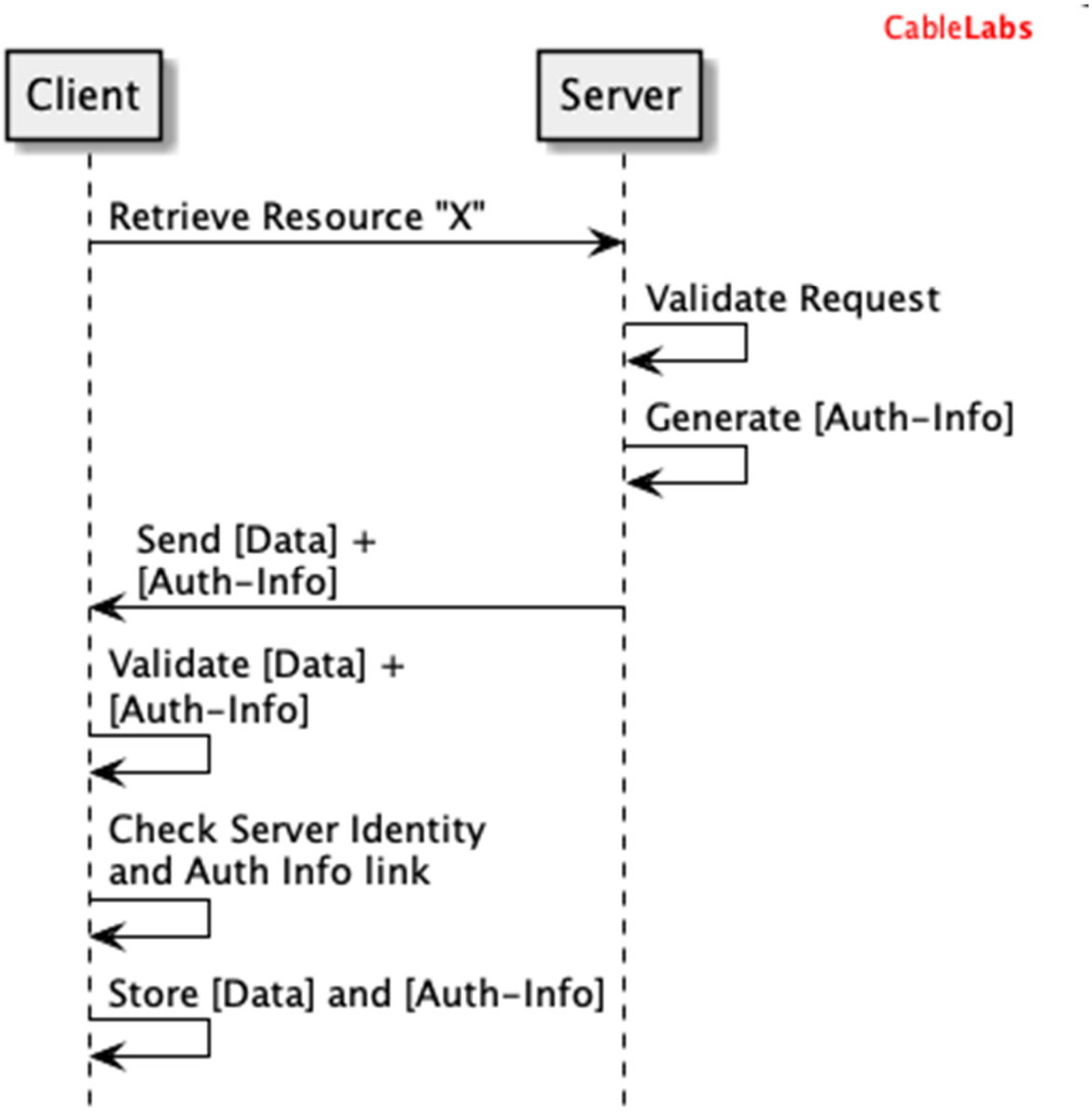
4.2 SCP PAPs

When transferring data via SCP, if a certificate is used by the origin party (i.e., the party where the data is transferred from) to authenticate itself, the same approach as in HTTP/S PAPs can be used where the authentication data is provided embedded within the authentication information (e.g., a CMS structure)

4.3 SSH PAPs

This idea can also be applied to retain an authenticated log of the operations executed during the session. Specifically, before closing the SSH session, an SCP transfer is initiated to retrieve the session's log/text. The received text




CableLabs
Client
Server
Retrieve Resource "X"
Validate Request
Generate [Auth-Info]
Send [Data] +
[Auth-Info]
Validate [Data] +
[Auth-Info]
Check Server Identity
and Auth Info link
Store [Data] and [Auth-Info]

METHOD FOR PERMANENT AUTHENTICATION PROOFS

Defining authentication proofs for permanent and verifiable data origin

Abstract

This work describes a novel approach to track the origin of transferred data (e.g., configurations, documents, applications, etc.) independently from the producer.

Massimiliano Pala
m.pala@cablelabs.com

TABLE OF CONTENTS

LIST OF ACRONYMS

NONCE – Unique Session Random Value

Bibliography

The Internet Engineering Task Force (IETF) – IETF RFC 6066. *Transport Layer Security (TLS) Extensions: Extension Definitions*, edited by D. Eastlake 3rd et al., January 2011. Also available at https://datatracker.ietf.org/doc/rfc6066

WI-FI Alliance – *Device Provisioning Protocol Specification Version 1.1*, WiFi Alliance. Also available at https://www.wi-fi.org/download.php?file=/sites/default/files/private/Device_Provisioning_Protocol_Specification_v1.1_1.pdf

1 Overview

In security, one important parameter for assuring and being able to verify that data, configurations, or applications come from authenticated and trusted origins. Today, we leverage the use of secure protocols such as TLS or SSH to make sure these properties are well verified - the data comes from the right internal/public server and the data has not been altered.

This leaves the user with the hard task of convincing 3rd parties that the data was downloaded from the right server or website. Since the same data could have been downloaded from a different site, the only type of validation that can be done at that point is to use some form of checking that the secure hash (or checksum) value calculated on the transferred data is correct. Very manual process, prone to error, and used only for software. Regrettably, because of the impracticality of the current approach, the origin validation is lost after the data is actually transferred.

1.1 Current Approach and Problem Statement

The problem with this approach is that once the data has been transferred from one entity to the other, the authentication and integrity information is lost. Specifically, under the assumption that we can identify the origin because the data was correctly decrypted and the encryption key was negotiated with a well-known server (i.e., a server whose certificate/key is verified and trusted), there is no possible way to retain that information after the data has been decrypted. In order to preserve that capability, the whole encryption key and negotiation (key-exchange) session would have to be stored (and shared with the receiving party).

This work changes the inability to retain the origin of data/documents/text/configurations /etc. by adding the concept of delivering a permanent authentication for the origin of the data.

2 Document Notation

The symbol (" | ") is used throughout this document to indicate concatenation of two values. Specifically, when indicating the concatenation of values A and B, this document uses the following notation:

{ A | B }

The symbols ("{") and ("}") indicate the beginning and end of a logical record.

3 The Invention Overview

In this invention we recognize the ("Server") and the ("Client") as the two parties involved in a secure communication. The terminology ("Client") and ("Server") will be used to help distinguishing the two parties of a communication (e.g., "Entity A" and "Entity B" can be referenced as "Client" and "Server"). This said, there are no special roles for server or client in

this work as the invention can be applied symmetrically – the only requirement for the patent idea to work is for the party where the data is transferred from (origin) can provide an authentication (e.g., a signature) tied to its own identity and the data that was transferred (e.g., a signature calculated over the transferred data).

3.1 The Assumptions

In this invention, it is assumed that two parties are connected to each other (e.g., have the possibility to transfer data across each other), not necessarily through a secure protocol such as HTTPS or SSH. In the case of secure protocols, it is assumed that the originating party (i.e., the party where the data is transferred from) can generate the authentication information via its own private key or token.

NOTE WELL: If a non-secure protocol is used, the binding with the origin is only through the additional information provided by the party and it provides weaker security properties (i.e., no network-binding properties such as DNS names can be associated with the additional authentication information because it cannot be validated by the client).

3.2 The Workflow

The workflow is depicted in the next figure:

Specifically, in this work, when the ("Client") requests a specific resource from the ("Server"), the server returns the resource together with the additional source validation information or "Permanent Authentication Proofs" (PAPs).

The PAPs are signed tokens that can be instantiated with different technologies and formats. It is suggested that a standard format is used to facilitate interoperability across systems and environments. An example is the Cryptographic Message Syntax (CMS) format that is suitable for carrying the origin information when a certificate or a key is used as the origin's identity.

After the data and PAP are transferred, they can be used together to provide both origin and integrity information. For example, when CMS is used, the authentication information should have the signerInfo data structure configured for the origin's identity details, together with all the required certificates or keys needed to build the chain to a root source of trust such as a trusted root CA or root key. Additional information such as OCSP responses or CRLs can be also added to the authentication information for proof of validity at the time of transferring the data.

4 PAPs Generation Examples for different Protocols

Although this work is independent from the specific protocol used underneath to transfer the data, it is important to provide some technical aspects for the major protocols in use today across the world.

4.1 HTTP/S PAPs

In the HTTP/S world, when a resource is requested, it is possible to provide the additional authentication information via a multi-part message where the authentication data can be received and saved separately from the data to be authenticated.

Alternatively, the server can be configured to generate a data+signature single file (e.g., by using a CMS signature where the signed content is actually provided within the data structure itself in a so called non-detached signature)

4.2 SCP PAPs

When transferring data via SCP, if a certificate is used by the origin party (i.e., the party where the data is transferred from) to authenticate itself, the same approach as in HTTP/S PAPs can be used where the authentication data is provided embedded within the authentication information (e.g., a CMS structure)

4.3 SSH PAPs

This idea can also be applied to retain an authenticated log of the operations executed during the session. Specifically, before closing the SSH session, an SCP transfer is initiated to retrieve the session's log/text. The received text