Operations Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 14, 2014

S. Winter
RESTENA

February 10, 2014

A Configuration File Format for Extensible Authentication Protocol (EAP) Deployments

draft-winter-opsawg-eap-metadata-00

Abstract

This document specifies a file format for transfering configuration information of deployments of the Extensible Authentication Protocol (EAP). Such configuration files are meant to be discovered, consumed and used by EAP supplicant software to achieve secure and automatic EAP configuration on the consuming device.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 14, 2014.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Problem Statement 2
1.2. Other Approaches 4
1.3. Requirements Language 4
1.4. Terminology 4
2. XML Schema for EAP Metadata File Format 4
2.1. Location of XML Schema and Sample XML file 4
2.2. Description of Schema Elements 4
2.2.1. Overall structure 4
2.2.2. <AuthenticationMethods> 5
2.2.3. <ProviderInfo> 9
2.3. Internationalisation / Multi-language support 10
3. Issuer Authentication, Integrity Protection and Encryption of EAP Metadata configuration files 11
4. File Discovery 11
4.1. By MIME-Type: application/eap-config 11
4.2. By filename extension: .eap-config 11
4.3. By network location: SCAD 11
5. Existing Implementations 11
6. Design Decisions 12
6.1. Why XML and not $FOO? 12
6.2. Shallow vs. Deep definition of EAP method properties . . 12
6.3. EAP tunneling inside EAP tunnels 12
6.4. Placement of <OuterIdentity> inside <AuthenticationMethod> 12
7. Security Considerations 13
8. IANA Considerations 13
9. Contributors 14
10. References 14
10.1. Normative References 14
10.2. Informative References 14
Appendix A. Appendix A: MIME Type Registration Template 15

1. Introduction

1.1. Problem Statement

The IETF has produced the Extensible Authentication Protocol (EAP, [RFC3748] and numerous EAP methods (for example EAP-TTLS [RFC5281], EAP-TLS [RFC5216] and [RFC5931]); the methods have many properties which need to be setup on the EAP server and matched as configuration items on the EAP peer for a secure EAP deployment.

Setting up these configuration items is comparatively easy if the end-user devices which implement the EAP peer functionality are under central administrative control, e.g. in closed enterprise environments. Group policies or device provisioning by the IT department can push the settings to user devices.

In other environments, for example "BYOD" scenarios where users bring their own devices which are not under enterprise control, or in EAP-based WISP environments (see e.g. [HS20] and [I-D.wierenga-ietf-eduroam]) where it is not desired neither for the ISP nor for his user that the device control is in the ISPs hands, configuration of EAP is significantly harder as it has to be done by potentially very non-technical end users.

Correct configuration of all EAP deployment parameters is required to make the resulting authentications

- functional (i.e. the end user can authenticate to an EAP server at all)
- secure (i.e. the end user device can unambiguously authenticate the EAP server prior to releasing any sensitive client-side credentials)
- privacy-preserving (i.e. the end user is able to conceal his username from the EAP authenticator)

It would be desirable to be able to convey the EAP configuration information of a deployment in a machine parseable way to the end-user device, so that all the gory details need not be known/ understood by the user. Instead, the EAP peer software on the device could consume the configuration information and set up all EAP authentication details automatically.

However, there is currently no standard way of communicating configuration parameters about an EAP setup to the EAP peer.

This specification defines such a file format for EAP configuration metadata.

The specification allows for unique identification of an EAP identity provider by scoping it into a namespace and giving it a unique name inside that namespace. Using this unique identification, other configuration files (e.g. which detail an Enterprise Wi-Fi setup) can then refer to this particular instance of EAP identity information as authentication source.

1.2. Other Approaches

Device manufacturers sometimes have developed their own proprietary configuration formats, examples include Apple's "mobileconfig" (MIME type application/x-apple-aspen-config), Microsoft's XML schemata for EAP methods for use with the command-line "netsh" tool, or Intel's "PRO/Set Wireless" binary configuration files. The multitude of proprietary file formats and their different levels of richness in expression of EAP details create a very heterogenous and non-interoperable landscape.

New devices which would like to benefit from machine-parseable EAP configuration currently either have to choose to follow a competitor's approach and use that competitor's file format or have to develop their own. This situation is very unsatisfactory.

1.3. Requirements Language

In this document, several words are used to signify the requirements of the specification. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119]

1.4. Terminology

2. XML Schema for EAP Metadata File Format

2.1. Location of XML Schema and Sample XML file

The schema files are currently hosted on this preliminary location:

- Schema: http://ticker.eduroam.lu/cat/EAP-metadata/eap-metadata.xsd
- Sample: http://ticker.eduroam.lu/cat/EAP-metadata/eap-metadata.xml

2.2. Description of Schema Elements

2.2.1. Overall structure

The root element is the <EAPIdentityProviderList> tag, which contains a sequence of <EAPIdentityProvider> elements; these carry the actual installer information. In most practical applications, the <EAPIdentityProviderList> will contain only a single element; a longer list can be used for metadata transfers between systems or to allow users to select from a set of providers in one file.

Every <EAPIdentityProvider> has two attributes which make it globally unique: one attribute is the 'namespace' attribute which defines the namespace inside which this EAPIdentityProvider is unique; the other attribute is the 'ID' attribute which specifies the unique name inside the namespace. The element contains the following sub-elements:

- zero or one <ValidUntil> timestamp with an indication of possible expiry of the information in the configuration file. EAP peers importing the configuration file can use this information for example to re-assess whether the account is still valid (e.g. if the ValidUnil timestamp has passed, and authentication attempts consistently fail, the supplicant should consider the information stale and ask the user to verify his access authorisation with the EAP identity provider)

- exactly one <AuthenticationMethods> block contains a list of EAP methods which the EAPIdentityProvider supports. This element is described in more detail in section Section 2.2.2

- zero or one <ProviderInfo> blocks provide additional information about the EAPIdentityProvider, e.g. a logo to allow visual identification of the provider to the user in a user interface, or Acceptable Use Policies pertaining to the use of this EAP identity. This element is described in more detail in section Section 2.2.3

- zero or more <VendorSpecific> elements with undefined structure for cases where particular implementations of this specification need to convey additional data which is not covered by the other elements of this specification and does not require cross-vendor interoperability. The attribute "vendor" of the element MUST contain the vendor's IANA Enterprise Number.

2.2.2. <AuthenticationMethods>

<AuthenticationMethods> is a sequence of <AuthenticationMethod> elements. Each such element specifies the properties of one supported authentication method with various elements. These elements are enumerated in section Section 2.2.2.1 The set of configuration parameters depends on the particular EAP method to be configured.

For instance, EAP-PWD [RFC5931] does not require any server certificate parameters; EAP-FAST and TEAP are the only ones making use of Protected Access Credential (PAC) provisioning. On the other hand, properties such as outer ("anonymous") identity or the need for a trusted root Certification Authority are common to several EAP

methods. The server- and client-side credential types of EAP methods are defined as a flat list of elements to choose from (see <ServerSideCredential> and <ClientSideCredential> below); see section Section 6.2 for a rationale.

Where the sequence of <AuthenticationMethod> elements contains more than one element, the order of appearance in the file indicates the server operator's preference for the supported EAP types; occurences earlier in the file indicate a more preferred authentication method.

When a consuming device receives multiple <AuthenticationMethod> elements, it should attempt to install more preferred methods first. If the configuration information for that method is insufficient (e.g. the <AuthenticationMethod> is EAP-TLS, but the configuration file does not contain the client certificate/private key and the device's credential store is not pre-loaded with the client's certificate), the device should query whether the more preferred method should be used (requiring the user to supplement the missing data) or whether a less-preferred method should be configured. In non-interactive provisioning scenarios, all methods should be tried in order until one method can be installed; if no method can be installed in a fully automated way, provisioning is aborted.

2.2.2.1. Authentication Method Properties

The <AuthenticationMethod> element contains

- exactly one <EAPMethod> element, which is an integer of the EAP method identifier as assigned by IANA
- zero or one <ServerSideCredential> elements which are a complex type containing elements which define means to authenticate the EAP server to the EAP peer (for a list of these elements, see section Section 2.2.2.2)
- zero or one <ClientSideCredential> elements which are a complex type containing elements which define means to authenticate the EAP peer to the EAP server (for a list of these elements, see section Section 2.2.2.3)
- zero or more <InnerAuthenticationMethod> elements. Elements of this type indicate that a tunneled EAP method is in use, and that further server-side and/or client-side credentials are defined inside the tunnel. The presence of more than one InnerAuthenticationMethod indicates that EAP Method Chaining is in use, i.e. that several inner EAP methods are to be executed in sequence inside the tunnel.

The <InnerAuthenticationMethod> element itself contains the same <EAPMethod>, <ServerSideCredentials> and <ClientSideCredentials> as described in the preceding list, but differs in two points:

- It can optionally contain the element <NonEAPAuthMethod> (an enumerated integer of authentication methods not based on EAP) instead of <EAPMethod> because some tunneled EAP types do not necessarily contain EAP inside the tunnel (e.g. TTLS-PAP, TEAP). Note that the XML Schema formally allows to specify both <EAPMethod> and <NonEAPAuthMethod>. This situation MUST NOT occur in configuration files to ensure deterministic interpretability.

- It can NOT contain further <InnerAuthenticationMethod> elements because establishing a secure tunnel inside an already established secure tunnel is considered a pathological case which needs not be considered. See section Section 6.3 for a rationale.

2.2.2.2. <ServerSideCredential> Properties

The server-side authentication of a mutually authenticating EAP method is typically based on X.509 certificates, which requires the EAP peer to be pre-provisioned with one or more trusted root Certification Authority prior to authenticating. A server is uniquely identified by presenting a certificate which is signed by these trusted CAs, and by the EAP peer verifying that the name of the server matches the expected one. Consequently, a (set of) CAs and a (set of) server names make up the ServerSideCredentials block.

Note that different EAP methods use different terminology when referring to trusted CA roots, server certificates, and server name identification. They also differ or have inherent ambiguity in their interpretation on where to extract the server name from (e.g. is the server name the CN part of the DistinguishedName, or is the server name one of the subjectAltName:DNS entries; what to do if there is a mismatch?). This specification introduces one single element for CA trust roots and naming; these notions map into the naming of the particular EAP methods very naturally. This specification can not remove the CN vs. sAN:DNS ambiguity in many EAP methods.

- zero or more <CA> elements: a Certification Authority which is trusted to sign the expected server certificate. The set of <CA> elements SHOULD contain self-signed root certificates to establish trust, and MAY contain additional intermediate CA certificates which ultimately root in these self-signed root CAs. A configuration file can, but SHOULD NOT include only an intermediate CA certificate (i.e. without also including the corresponding self-signed root) because trusting only an

intermediate CA without being able to verify to a self-signed root is an unsupported notion in many EAP peers.

- zero or more <ServerID> elements: these elements contain the expected server names in incoming X.509 EAP server certificates. For EAP methods not using X.509 certificates for their mutual authentication, these elements contain other string-based handles which identify the server (Example: EAP-pwd).

2.2.2.3. <ClientSideCredential> Properties

There is a variety of means to identify the EAP peer to the EAP server. EAP methods use a subset of these criteria. As with server-side credentials, the terminology for the credential type may differ slightly between EAP types. The naming convention in this specification maps nicely into the method-specific terminology. Not all the criteria make sense in all contexts; for EAP methods which do not support a criterion, configuration files SHOULD NOT contain the corresponding elements, and consumers of the file MUST ignore these elements.

Specifying any one of these elements is optional and they can occur at most once. Consumers of configuration files MUST be able to fall back to user-interactive configuration for these parts if they are not specified (e.g. ask for the username and password for an EAP method during import of the EAP configuration data). Configuration files which do contain sensitive elements such as <Password> MUST be handled with due care after the import on the device (e.g. ensure minimal file permissions, or delete the source file after installing). The <ClientSideCredential> element has an attribute 'allow_save'; if it is set to false, sensitive parts of the client-side credentials MUST NOT be permanently saved on the device. See also section Section 3 for transport security considerations.

<OuterIdentity> is typcially used on the outside of a tunneled EAP method and allows to specify which user identity should be used outside the tunnel. This string is not used for actual user authentication, but may contain routing hints to send the request to the right EAP server.

<UserName> contains the actual username to be used for user authentication. For tunneled EAP methods, this element SHOULD only occur in the <InnerAuthenticationMethod>'s <ClientSideCredentials> - if differing outer identities are not desired in the deployment, the <OuterIdentity> element should be populated for the <AuthenticationMethod> element; but may contain the actual username then.

<ClientCertificate> contains a X.509 certificate and private key; if the key is protected, the <Passphrase> element MAY be used to indicate the passphrase, see below

<Passphrase> contains the passphrase needed to unlock a cryptographic credential internally on the device (i.e. it is not used itself for the actual authentication during the EAP conversation)

<Password> contains the user's password, or an otherwise secret string which the user needs to authenticate to the EAP server

<PAC> contains the Protected Access Credential, typically used in EAP-FAST and TEAP.

<ProvisionPAC> is a boolean which indicates whether a PAC should be provisioned on the first connection. Note that the specification allows to use <ProvisionPAC> without a CA nor ServerID in <ServerSideCredential>. While this allows the operation mode of "Anonymous PAC Provisioning" as used in EAP-FAST, due to the known security vulnerabilities of anonymous PAC provisioning, this combination SHOULD NOT be used.

2.2.3. <ProviderInfo>

This specification needs to consider that user interaction during the installation time may be required; the user at the very least must be empowered to decide whether the configuration file was issued by a provider he has an account with; the provider may have hints for the user (e.g. which password to use for the login), or may want to display links to helpdesk pages in case the user has problems with the setup or use of his identity.

The <ProviderInfo> element allows to specify a range of potentially useful information for display to the user (some of which is relevant only during installation time, other pieces of information could be retained by the EAP peer implementation and displayed e.g. in case of failed authentication):

- <DisplayName> specifies a user-friendly name for the EAP Identity Provider. Consumers of this specification should be aware that this is simple text, and self-asserted by the producer of the configuration file. If more authoritative information about the issuer is available (e.g. if the file is signed with S/MIME and carries an Organisation name (O attribute) in the signing certificate) then the more authoritative information should be displayed with more prominence than the self-asserted one.

- <Description> specifies a generic descriptive text which should be displayed to the user prior to the installation of the configuration data.

- <ProviderLocation> specifies the approximate geograhic location(s) of the EAP Identity Provider and/or his Points of Presence. This can be useful if the configuration file contains multiple <EAPIdentityProvider> elements; the user device can then make an informed guess which of the Identity Providers could be a good match to suggest to the user

- <ProviderLogo> specifies the logo of the EAP Identity Provider. The same self-assertion considerations as for <DisplayName> above apply.

- <TermsOfUse> contains terms of use to be displayed to and acknowledged by the user prior to the installation of the configuration on the user's system

- <Helpdesk> is a complex element with three possible sub-elements: <EmailAddress>, <WebAddress> and <Phone>, all of which can be displayed to the user.

2.3. Internationalisation / Multi-language support

Some elements in this specification contain text to be displayed in User Interfaces; depending on the user's language preferences, it would be desirable to present the information in a local language. Other elements contain contact information, and those contact points may only be able to handle requests in a number of languages; it may be desirable to present only contact points to the user which are compatible with his language capabilities.

All elements which either contain localisable text, or which point to external resources in localised languages, have an optional "lang" attribute. The elements can occur more than once in the specification, which enables an iteration of the element in all applicable languages. If the "lang" attribute is omitted or "lang" is set to "C", the instance of the element is considered a default choice which is to be displayed if no other instance is a better match.

If the entire file content consistently uses only one language set, e.g. all the elements are to be treated as "default" choices, the language can also be set for the entire <EAPIdentityProvider> element in its own "lang" attribute.

3. Issuer Authentication, Integrity Protection and Encryption of EAP Metadata configuration files

S/MIME or underlying transport security. Nuff said :-)

4. File Discovery

4.1. By MIME-Type: application/eap-config

For transports where the categorisation of file types via MIME types is possible (e.g. HTTP, E-Mail), this document assigns the MIME type

application/eap-config

Edge devices can associate this MIME type to incoming files on such transports, and register the application which can consume the EAP Metadata as the default handler for this file type. By doing so, for example a single click or tap on a link to the file in the device's browser will invoke the configuration process.

This method of discovery is analogous to the Apple "mobileconfig" discovery on recent versions of Mac OS and iOS.

4.2. By filename extension: .eap-config

In situations where file types can not be determined by MIME type meta-information (e.g. when the file gets stored on a local filesystem), this document RECOMMENDs that EAP Metadata configuration files be stored with the extension

.eap-config

to identify the file as containing EAP Metadata configuration information. Edge devices can register the application which can consume the EAP Metadata with this file extension. By doing so, for example a single click or tap on the filename in the device's User Interface will invoke the configuration process.

4.3. By network location: SCAD

5. Existing Implementations

Producers of the configuration files

- eduroam Configuration Assistant Tool: this existing tool already produces EAP configuration files in various proprietary formats for hundreds of EAP Identity Providers. The authors of this

specification will add a module which will produce configuration files in the file format as specified in this document.

Consumers of the configuration files

- Android: the authors of this specification are currently developing an App for the Android operating system (compatible with API level 18 of Android, i.e. version 4.3 and above) which can consume the file format as defined in this draft specification and configure EAP via the WifiEnterpriseConfig API.

- Linux: the authors of this specification are currently developing an application for UNIX-like operating systems which configure enterprise networks via the NetworkManager daemon; the application can consume the file format as defined in this draft specification and configure the settings via Networkmanager's D-BUS interface.

6. Design Decisions

6.1. Why XML and not $FOO?

XML is a popular choice for EAP configurations: Microsoft's "netsh" files, Apple's "mobileconfig" files, the Wi-Fi Alliance's "PerProviderSubscription Managed Object", and other vendor/SDO definitions are all using XML.

Other possibilities which will be duly considered if sufficient interest warrants it include, but are not limited to:

- JSON (less rich expressions; no verification of conformity such as with XML Schema - but it doesn't need many resources to parse and may thus be advantageous for constrained devices)

- YANG (very rich feature set, and tools can produce automatic conversions to both XML and JSON - but not as well understood by the author, and unlikely to be natively supported on consumer devices)

6.2. Shallow vs. Deep definition of EAP method properties

6.3. EAP tunneling inside EAP tunnels

6.4. Placement of <OuterIdentity> inside <AuthenticationMethod>

7. Security Considerations

8. IANA Considerations

IANA is requested to allocate the MIME type "application/eap-config" in the MIME Media Types / application registry (see section Section 4.1). The allocation should contain the following values:

- Name: eap-config
- Template: see Appendix A (RFC editor note: remove this appendix prior to publication; replace this line with the URL to the application as posted online)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

IANA is requested to allocate the location "TBD" in the "well-known URIs" registry. The allocation should contain the following values:

- URI Suffix: TBD
- Change Controller: IETF
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)
- Related Information: none

IANA is requested to register the XML namespace "urn:ietf:params:xml:ns:eap-config" in the "IETF XML Registry / ns". The allocation should contain the following values:

- ID: eap-config
- URI: urn:ietf:params:xml:ns:eap-config
- Filename: https://www.iana.org/assignments/xml-registry/ns/eap-config.txt (to be created by IANA)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

IANA is requested to register the XML schema "urn:ietf:params:xml:schema:eap-config" in the "IETF XML Registry / schema". The allocation should contain the following values:

- ID: eap-config

- URI: urn:ietf:params:xml:schema:eap-config
- Filename: https://www.iana.org/assignments/xml-registry/schema/eap-config.xsd (to be created by IANA; current XSD file is linked to in section Section 2.1)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

9. Contributors

Tomasz Wolniewicz of Nicolaus Copernicus University in Torun, Poland, provided significant input into this specification.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, August 2008.

[RFC5931] Harkins, D. and G. Zorn, "Extensible Authentication Protocol (EAP) Authentication Using Only a Password", RFC 5931, August 2010.

[I-D.wierenga-ietf-eduroam] Wierenga, K., Winter, S., and T. Wolniewicz, "The eduroam architecture for network roaming", draft-wierenga-ietf-eduroam-02 (work in progress), January 2014.

[HS20] Wi-Fi Alliance, "Hotspot 2.0 Technical Specification", 2012, <https://www.wi-fi.org/hotspot-20-technical-specification-v100>.

Appendix A. Appendix A: MIME Type Registration Template

The following values will be used for the online MIME type registration at https://www.iana.org/form/media-types

Your Name: Stefan Winter

Your Email Address: stefan.winter@restena.lu

Media Type Name: Application

Subtype name: (Standards tree) eap-config

Required parameters: (none)

Optional parameters: (none)

Encoding Considerations: 8-Bit text

Security Considerations: This file type carries configuration information for consumer devices. It has the potential to substantially alter the consumer's device; particularly to install a new trusted Certification Authority. Applications consuming files of this type need to be cautious to explain to the end user what is being altered, so that they understand the consequences. For further explanations, see Section 7 of draft-winter-opsawg-eap-metadata. (Note to IANA: replace this reference with the RFC number of this document once known)

Interoperability Considerations: The file content is XML version 1.0 or later. The encoding SHOULD be UTF-8, but implementations consuming the file SHOULD be prepared to encounter different encodings.

Published Specification: draft-winter-opsawg-eap-metadata (Note to IANA: replace this reference with the RFC number of this document once known)

Applications which use this media type: files of this type are intended for consumption by sortware on edge devices; they consume the information therein to configure authentication parameters (EAP protocol and EAP method payload configurations) which are then applied to network or application authentication scenarios.

Fragment Identifier Considerations: files of this type are expected to be transmitted in their entirety. If a reference to a specific part of the content is to be made, XML XPath expressions

are to be used. I.e. fragment identifier formats are not expected to be used.

Restrictions on Usage: none

Provisional registration: initial submission of this form will be executed after adoption in the IETF; it will be a provisional registration. Final registration will be done after IESG review.

Additional information:

Deprecated alias types for this name: none

Magic numbers: none

File extensions: eap-config

Macintosh File Type Codes: TBD

Object Identifiers or OIDs: none

Intended Usage: Common (no further provisions)

Other Information/General Comment: none

Person to contact for further information:

Name: Stefan Winter

E-Mail: stefan.winter@restena.lu

Author/Change controller: IETF

DATA

Author's Address

Stefan Winter
Fondation RESTENA
6, rue Richard Coudenhove-Kalergi
Luxembourg 1359
LUXEMBOURG

Phone: +352 424409 1
Fax: +352 422473
EMail: stefan.winter@restena.lu
URI: http://www.restena.lu.