

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: 8 September 2022

J.-F. Rieckers  
DFN  
7 March 2022

Observations about EAP-NOOB ([RFC 9140](#))  
draft-rieckers-emu-eap-noob-observations-00

## Abstract

This memo is a random list of things the author noticed about EAP-NOOB when looking at the draft and running the implementation while capturing the packets (<https://github.com/Vogeltak/hostap>).

Most of the statements were written down before the author started the implementation. By the time of writing this draft, a mostly complete server implementation has been written. The implementation-specific remarks are mostly thoughts the author had while planning their own implementation.

## About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at <https://datatracker.ietf.org/doc/draft-rieckers-emu-eap-noob-observations/>.

Discussion of this document takes place on the EAP Method Update (emu) Working Group mailing list (<mailto:emu@ietf.org>), which is archived at <https://mailarchive.ietf.org/arch/browse/emu/>.

## 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 <https://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."

Internet-Draft

EAP-NOOB observations

March 2022

This Internet-Draft will expire on 8 September 2022.

## Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the [Trust Legal Provisions](#) and are provided without warranty as described in the Revised BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Areas of observations . . . . .	<a href="#">2</a>
<a href="#">2.1.</a>	Why use JSON? . . . . .	<a href="#">3</a>
<a href="#">2.2.</a>	IANA-Registries . . . . .	<a href="#">3</a>
<a href="#">2.3.</a>	Unnecessarily long messages . . . . .	<a href="#">4</a>
<a href="#">2.4.</a>	Unclear status of ServerInfo and PeerInfo . . . . .	<a href="#">4</a>
<a href="#">2.5.</a>	Number of messages exchanged . . . . .	<a href="#">5</a>
<a href="#">2.6.</a>	Reconnecting State in state machine . . . . .	<a href="#">6</a>
<a href="#">2.7.</a>	Missing Specification . . . . .	<a href="#">6</a>
<a href="#">3.</a>	IANA Considerations . . . . .	<a href="#">6</a>
<a href="#">4.</a>	Security Considerations . . . . .	<a href="#">6</a>
<a href="#">5.</a>	References . . . . .	<a href="#">6</a>
<a href="#">5.1.</a>	Normative References . . . . .	<a href="#">6</a>
<a href="#">5.2.</a>	Informative References . . . . .	<a href="#">7</a>
	Acknowledgements . . . . .	<a href="#">7</a>
	Contributors . . . . .	<a href="#">7</a>
	Author's Address . . . . .	<a href="#">7</a>

## [1.](#) Introduction

(see abstract)

## [2.](#) Areas of observations

### [2.1.](#) Why use JSON?

The author's initial reaction when they first read the draft was: Why use JSON and not a binary protocol? Almost no advantages of using JSON (dynamic keys, reordering, flexible and possible deep structure) are needed for or used in this protocol, while on the other hand the JSON serialization is problematic for the calculation of MAC/Hoob/... Especially the need to extract the JSON value of the PeerInfo and ServerInfo field may cause problems in the Hash/HMAC calculation, if peer and server do not agree on a JSON serialization.

The protocol has many static parts (always the field "Type", mostly fixed parameters depending on the type, ...), so it seems as if the number of bytes and the parsing effort could be significantly reduced if static protocol messages (e.g. like in the TLS Handshake messages) were used.

If the message format should be more flexible, CBOR seems to be a good idea as well. Here a simple map (best with numbers as keys) could be used, combined with deterministic encoding ([Section 4.2 of \[RFC8949\]](#)).

### [2.2.](#) IANA-Registries

EAP-NOOB has a striking number of IANA-registered protocol parameters with a dedicated IANA registry page only for EAP-NOOB. At first glance this seems a bit odd, since the protocol does not seem to be this complex.

Especially the choice for a designated registry for cryptosuites seems odd. With the current situation, every new curve has to be respecified instead of just using existing registries. This requires implementers to update their EAP-NOOB implementation instead of just updating the cryptographic library behind it. The current specification specifies only two cryptosuites, which may not be feasible for all use cases, especially if higher security levels are

required.

As a possible alternative, the COSE algorithms registry provides short identifiers for many relevant parameters.

Another odd thing is the Server-/PeerInfo Registry, see [Section 2.4](#).

### [2.3](#). Unnecessarily long messages

It seems that the protocol uses an unnecessarily high number of bytes to transfer the information. Though this should not lead to issues on the EAP/WPA2 level, it seems that the protocol is not well-suited for constrained devices.

This is especially prominent for the encoding of the ServerInfo and PeerInfo fields. Since the IANA policy is "Specification required", shorter identifiers could have been used instead of human-readable names, leading to shorter messages.

The author was also a little confused about the reason why the PeerId is always retransmitted. Other EAP-Methods do just fine without any explicit information to identify the exact EAP instance running, so it seems a bit odd that the server and the peer keep telling each other what the PeerId is. From the author's understanding, the PeerId only has to be transmitted once, either in the first EAP-NOOB message from the peer to the server, if the PeerId was assigned in a previous EAP-NOOB conversation, or in the second EAP-NOOB message from the server to the peer, if a new PeerId has to be allocated. The PeerId can then be saved to the current EAP state information. This also seems useful to prevent bad implementations where the PeerId is not checked every time (since it is always the same anyway), but for the calculation of Hoob/..., only the one transmitted last is used, which an attacker could have modified. However, this is a random thought of how broken implementations could be; the author has not yet analyzed the actual attack vector or its effects.

In the server implementation by the author, the PeerId sent by the client is ignored in all messages except the aforementioned.

#### [2.4.](#) Unclear status of ServerInfo and PeerInfo

The specification is somewhat ambiguous regarding the ServerInfo and PeerInfo fields. [Section 3.3.2 of \[RFC9140\]](#) specifies them as "The format and semantics of these objects MUST be defined by the application that uses the EAP-NOOB method." On the other hand, specific fields are defined in [Section 5.4](#) and 5.5. In [Section 6.7](#) (Channel Binding) the RFC states: "The peer MAY include in PeerInfo any data items that it wants to bind to the EAP-NOOB association and to the exported keys."

This is especially interesting for the calculation of Hoob/MACs/MACp/... since the PeerInfo and ServerInfo fields are encoded as JSON, both the Server and the Peer have to somehow agree on a serialization of the JSON object inside. This could become extremely

tricky once either side sends attributes the other one does not expect, e.g. a field with an additional object inside. Since it is JSON and not a JSON-encoded string, the client has to parse the whole JSON tree in order to compute the correct Hoob/MACs/MACp/... value. Again, this could be easier with other data representation formats. With CBOR, the specification could simply use the bytes, which the CBOR parser could easily distinguish. Additionally, [\[RFC8949\]](#) specifies a canonical encoding, so both peer and server can generate the Hash/HMAC deterministically, even if the implementation only saves the content of the fields, not the encoded message.

Since the IANA registration procedure for PeerInfo/ServerInfo is "Specification required", additional use cases for these fields cannot easily be added. For a Masters project at the University of Bremen, students are looking into MUD ([\[RFC8520\]](#)), and it would be interesting to transmit the MUD URL in the PeerInfo, or the server could transmit some initial (public) configuration parameters for the device, e.g. the URI for a SmartHome controller. Since [RFC 9140](#) does not specify how to deal with unexpected or unknown fields, this could lead to unexpected behavior, if the peer does not understand the server's parameters.

## [2.5.](#) Number of messages exchanged

The number of messages exchanged seems to be on the high side.

For the initial exchange this leads to 4 roundtrips, for the reconnect exchange 5 roundtrips, which seems unnecessarily high. (The counts given start with the EAP-Response/Identity packet.)

Let's describe one straightforward way to reduce the number of messages needed, primarily for the initial handshake:

The server could send its supported Versions, Cryptosuites and Directions in the first EAP-NOOB message.

If the peer is in the initial state, in the last message of the common handshake, the peer already knows that it wants to negotiate a new set of connection parameters, so it could already transmit its ECDHE parameters along with the PeerInfo. If the server had already sent its supported versions, cryptosuites and directions, it could already choose the suitable methods.

In the answer the server could then transmit its ServerInfo and its ECDHE parameters. This saves at least one roundtrip. It also gives the server the ability to send different ServerInfo parameters depending on the type of the peer device. This would also enable configuration provisioning; for instance, if a light bulb is

connected, the server could already send the address of the light controller to which the bulb should connect to once the EAP-NOOB connection is established. (See previous point on ServerInfo and PeerInfo.)

If the peer is already connected, sending the supported versions and cryptosuites in the first message would also give the peer enough information to decide whether a renegotiation with a newer version or updated cryptographic keys is necessary and could already transmit new ECDHE parameters.

Details of these thoughts have been put into the draft [I-D.[draft-rieckers-emu-eap-ute](#)]. In this draft, all standard exchanges consist of 3 roundtrips.

## [2.6.](#) Reconnecting State in state machine

The existence of the "Reconnecting" state in the state diagram is a bit confusing, since the differentiation between Reconnecting and Registered State does not serve a further purpose other than to determine if the peer is currently connected to a network. For the server, the difference between Reconnecting and Registered state should not be relevant at all, since it is never the initiator of the EAP conversation. Once the peer loses its connection, a reconnect attempt has to be started anyway, so the peer should never start an EAP-NOOB handshake when in state 4 (this behavior is also specified). Specifying this state may help with implementation, but it might also confuse the implementer.

## [2.7.](#) Missing Specification

[RFC 9140](#) never explicitly specifies which value should be used for the Protocol Version field. Only example values are given in [Section 3.3.2](#) for Vers and Verp, from which the version number 1 could be guessed.

## [3.](#) IANA Considerations

This memo includes no request to IANA.

## [4.](#) Security Considerations

This document discusses a specification that is intended to provide security when initiating Internet access.

## [5.](#) References

### [5.1.](#) Normative References

Rieckers Expires 8 September 2022 [Page 6]

---

Internet-Draft EAP-NOOB observations March 2022

[RFC9140] Aura, T., Sethi, M., and A. Peltonen, "Nimble Out-of-Band Authentication for EAP (EAP-NOOB)", [RFC 9140](#), DOI 10.17487/RFC9140, December 2021, <<https://www.rfc-editor.org/info/rfc9140>>.

### [5.2.](#) Informative References

[I-D.[draft-rieckers-emu-eap-ute](#)]

Rieckers, J.-F., "User-assisted Trust Establishment (EAP-UTE)", Work in Progress, Internet-Draft, [draft-rieckers-emu-eap-ute-00](https://www.ietf.org/archive/id/draft-rieckers-emu-eap-ute-00), 2022, <<https://www.ietf.org/archive/id/draft-rieckers-emu-eap-ute-00.txt>>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", [RFC 8520](https://www.rfc-editor.org/info/rfc8520), DOI 10.17487/RFC8520, March 2019, <<https://www.rfc-editor.org/info/rfc8520>>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, [RFC 8949](https://www.rfc-editor.org/info/rfc8949), DOI 10.17487/RFC8949, December 2020, <<https://www.rfc-editor.org/info/rfc8949>>.

## Acknowledgements

TBD

## Contributors

Carsten Bormann  
Universität Bremen TZI  
Postfach 330440  
D-28359 Bremen  
Germany  
Phone: +49-421-218-63921  
Email: [cabo@tzi.org](mailto:cabo@tzi.org)

## Author's Address

Jan-Frederik Rieckers  
Deutsches Forschungsnetz | German National Research and Education Network  
Alexanderplatz 1  
10178 Berlin  
Germany  
Email: [rieckers@dfn.de](mailto:rieckers@dfn.de)  
URI: [www.dfn.de](http://www.dfn.de)