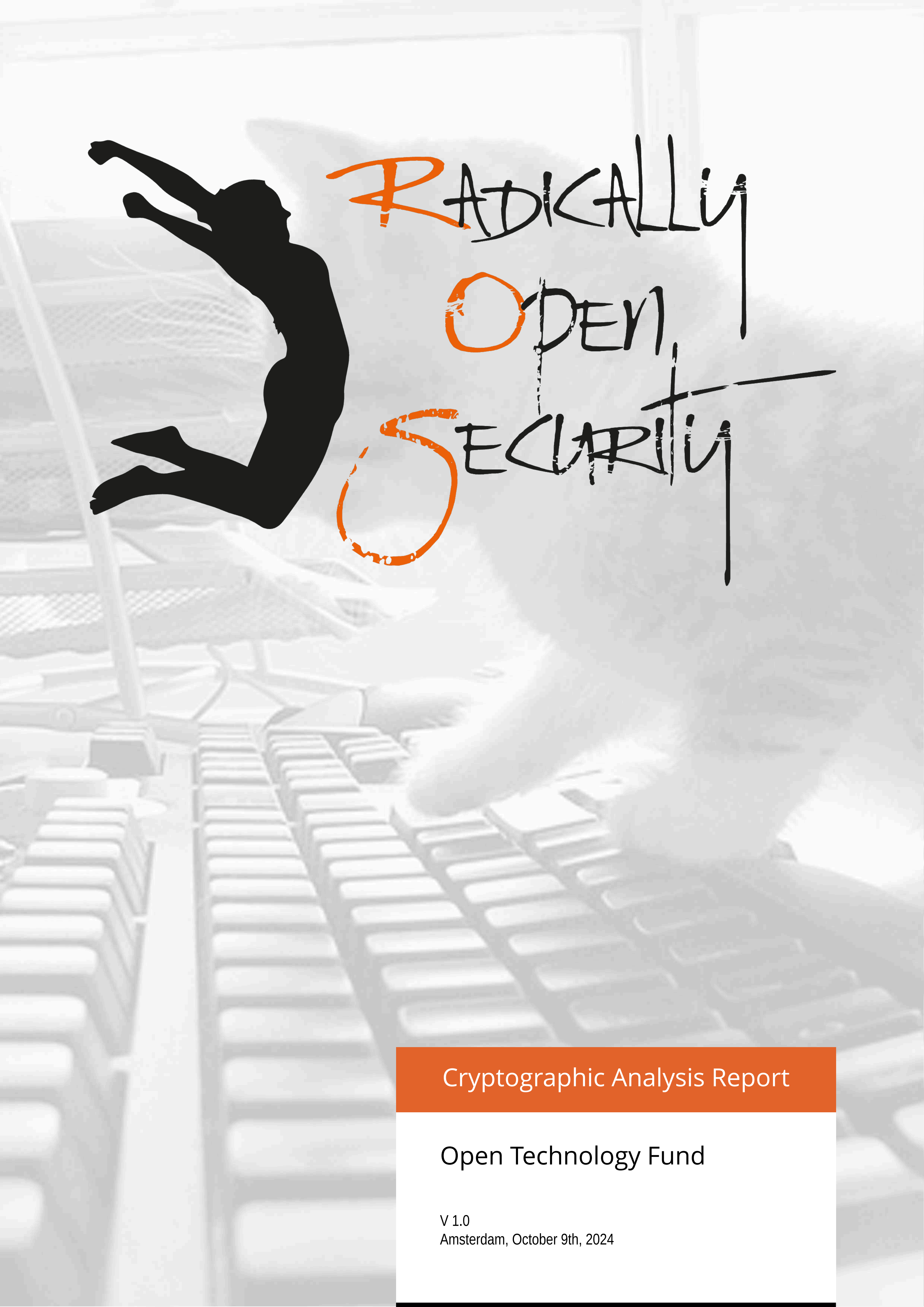


Radically Open Security

Cryptographic Analysis Report

Open Technology Fund

V 1.0
Amsterdam, October 9th, 2024

Document Properties

Client	Open Technology Fund
Title	Cryptographic Analysis Report
Target	The Awala network
Version	1.0
Pentester	Younes Talibi Alaoui
Authors	Younes Talibi Alaoui, Marcus Bointon
Reviewed by	Marcus Bointon
Approved by	Melanie Rieback

Version control

Version	Date	Author	Description
0.1	September 2nd, 2024	Younes Talibi Alaoui	Initial draft
0.2	September 17th, 2024	Younes Talibi Alaoui	Minor edits
0.3	October 2nd, 2024	Marcus Bointon	Review
0.4	October 5th, 2024	Younes Talibi Alaoui	Minor edits
1.0	October 9th, 2024	Marcus Bointon	1.0

Contact

For more information about this document and its contents please contact Radically Open Security B.V.

Name	Melanie Rieback
Address	Science Park 608 1098 XH Amsterdam The Netherlands
Phone	+31 (0)20 2621 255
Email	info@radicallyopensecurity.com

Radically Open Security B.V. is registered at the trade register of the Dutch chamber of commerce under number 60628081.

Table of Contents

1 Executive Summary

1.1 Introduction

Awala [1] is an overlay network designed to operate over the Internet when it is available, and switch to a sneakernet when the Internet is unavailable. Awala includes a channel session protocol [2], a Public Key Infrastructure [3] (Awala PKI), and various other protocols (see documentation [6] and corresponding repositories [4]) to ensure end-to-end encryption, even when using the sneakernet. The Awala network was designed by Relaycorp[1] .

1.2 Scope of work

This report provides a cryptographic analysis (as well as recommendations) of the Awala channel session protocol [2], and the usage of cryptographic primitives in the Awala network (Awala PKI, etc), with respect to the security guarantees it aims to fulfil. Additionally, we offer suggestions related to the Awala network's documentation. The content of this report can serve as additional material for developing Awala's threat model [5].

The analysis was constrained by a time limit of 11 days, two of which were dedicated to a code review of the implementation of the network. The analysis was conducted on the latest commits available at the start of the review[2] .

1.3 Summary of findings

Awala is built on well-cryptanalyzed cryptographic primitives, and makes extensive use of cryptographic standards. Through going over the Awala documentation, and detailed communication with Relaycorp, we identified several inconsistencies in the documentation, which have been addressed in subsequent commits. We also suggested adding more details to the documentation, which has been done in subsequent commits as well.

In addition to this, we provided several suggestions and recommendations, primarily focused on reducing the trust in the parties involved in Awala, some of which have already been considered for future implementation.

[1] `https://relaycorp.tech/`

[2] Latest commits before 2024-06-26

2 Overview of Awala

We go here over the relevant details for the cryptographic analysis, for more details one can refer to the Awala documentation [6]

2.1 The Awala network

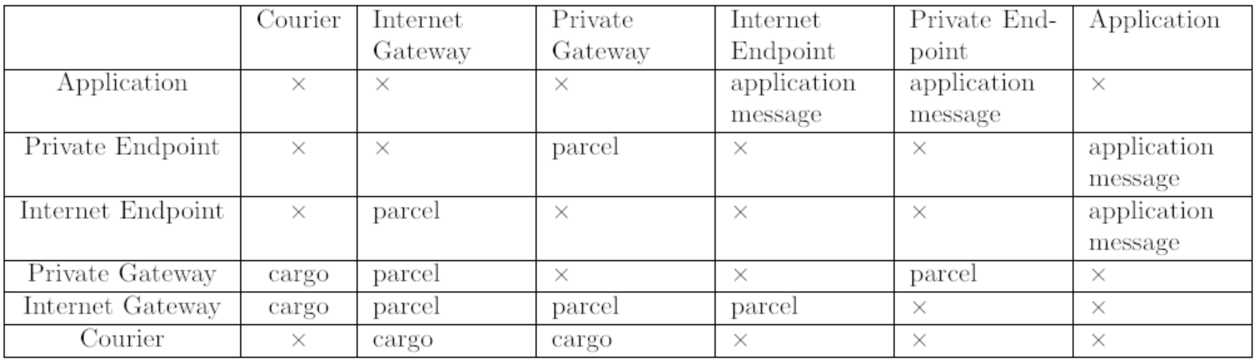
The Awala network is composed of four main components that exchange messages in different forms among themselves:

1. **Applications**: Applications are the components that wish to send each others messages using the network. We refer to these messages as application messages. That is, these are the components that would benefit from the Awala network to communicate in case the Internet is not accessible.
2. **Endpoints**: Endpoints act as brokers for the applications, so the applications can communicate. Endpoints can be either Internet endpoints or private endpoints. When an Endpoint receives a message from its application, it converts the message to a *Parcel*.
3. **Gateways**: Gateways act as a broker for endpoints to relay parcels. Gateways can be either private gateways that run on the user device to serve the endpoint on that device, or Internet gateways. When a Gateway receives a Parcel, if it has access to Internet, it forwards it to the next destination in the message's route. If the Gateway does not have Internet access, it encapsulates the Parcel in a *Cargo* and uses the sneakernet to send it.
4. **Couriers**: Couriers are actors that relay messages being sent in the form of Cargos when communication is happening in the sneakernet, i.e., when a Gateway that wants to send Parcels cannot do this through the Internet.

In accordance with Awala's documentation, we refer in what follows to Endpoints and Gateways as nodes in the network. We distinguish between two types of interaction that can happen between the different components of the network:

1. Adjacent-layer interaction, which is the interaction that happens between two successive components while a message is being routed in the network.
2. Same-layer interaction, which refers to the channels established between two components in the same layer (e.g. Alice's Endpoint with Bob's Endpoint).

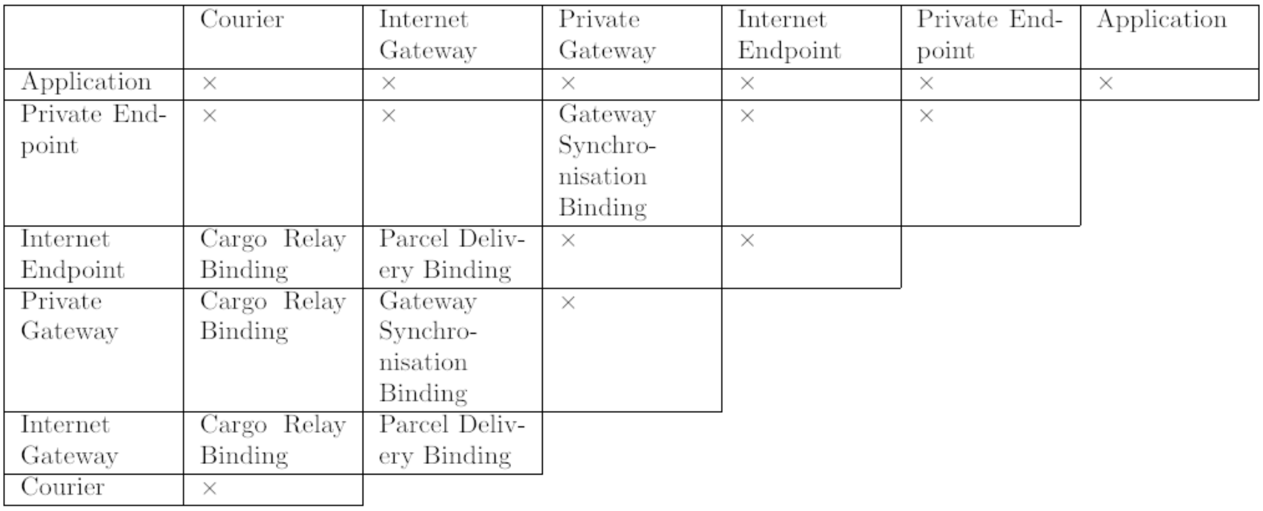
In table 1, we describe the forms of messages exchanged between the different components in an adjacent-layer interaction. Communication is directed from the component of the first column, to the component of the first row. Note that there are other types of messages (such as registration messages) that we have not included in the table. Also note that sending a message from an Internet Endpoint to an Internet Endpoint and from Courier to a Courier is not prohibited in the documentation, and might be supported by the software in future.



	Courier	Internet Gateway	Private Gateway	Internet Endpoint	Private Endpoint	Application
Application	×	×	×	application message	application message	×
Private Endpoint	×	×	parcel	×	×	application message
Internet Endpoint	×	parcel	×	×	×	application message
Private Gateway	cargo	parcel	×	×	parcel	×
Internet Gateway	cargo	parcel	parcel	parcel	×	×
Courier	×	cargo	cargo	×	×	×

Figure 1: Adjacent-layer interaction messages

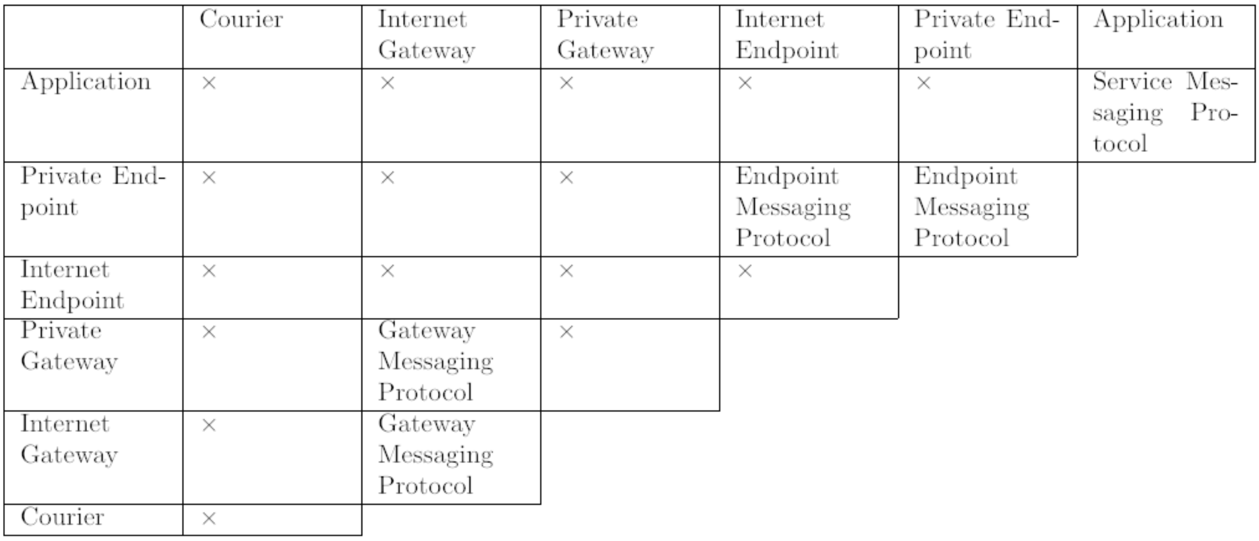
In table 2, we outline the protocols governing adjacent-layer interaction, which are referred to as bindings. One can find more details about Parcel delivery binding, Gateway synchronisation binding and Cargo relay binding in [6]. Note that communication between two Couriers is not prohibited in the documentation, and might be supported by the software in future.



	Courier	Internet Gateway	Private Gateway	Internet Endpoint	Private Endpoint	Application
Application	×	×	×	×	×	×
Private Endpoint	×	×	Gateway Synchronisation Binding	×	×	
Internet Endpoint	Cargo Relay Binding	Parcel Delivery Binding	×	×		
Private Gateway	Cargo Relay Binding	Gateway Synchronisation Binding	×			
Internet Gateway	Cargo Relay Binding	Parcel Delivery Binding				
Courier	×					

Figure 2:Adjacent-layer interaction protocols

In table 3, we describe the protocols governing same-layer interaction. These protocols are referred to as messaging protocols. One can find more details about, service messaging protocol, Endpoint messaging protocol and Gateway messaging protocol in [6]. Note that communication between two Internet Endpoints is not prohibited in the documentation, and might be supported by in future.



	Courier	Internet Gateway	Private Gateway	Internet Endpoint	Private Endpoint	Application
Application	×	×	×	×	×	Service Messaging Protocol
Private Endpoint	×	×	×	Endpoint Messaging Protocol	Endpoint Messaging Protocol	
Internet Endpoint	×	×	×	×		
Private Gateway	×	Gateway Messaging Protocol	×			
Internet Gateway	×	Gateway Messaging Protocol				
Courier	×					

Figure 3: Same-layer interaction protocols

For the sake of clarity, as one can deduce from the tables, note that communication between Alice and Bob does not need to involve all types of interaction, nor all types of components, for instance we can have a scenario where communication between Alice and Bob involves a private Endpoint and Gateway from each side, and one Internet Gateway. See more examples in https://awala.network/tech-overview.

Awala defines a channel session protocol for encrypting communication for same-layer interaction (more details given in [6]), and defines a serialization for messages. This serialization is called the Awala Abstract Message Format (RAMF for short). More details about this serialization, what it contains, and where it is used are given in the following sections.

2.2 PKI

The Awala network comes with a PKI which governs messages sent in the RAMF serialization. Every node in Awala has a long-term identity key pair that is used to issue certificates and sign RAMF messages. That is, an RSA key pair. This key might be held in more than one certificate, either self-signed, or issued by another node. Also, each node has an Id, which is the hash of the node's public key. A certificate in the Awala PKI is distinguished by the fact that the Common Name (CN) is a node's Id. This way, a self-signed certificate can be trusted by other parties, as the public key it attests to be associated with the CN of the certificate is the pre-image of the CN itself, and therefore, a party that did not generate the key pair corresponding to a CN cannot issue a valid signature for the self-signed certificate, as this requires the private key of the key pair.

Using Awala PKI certificates, nodes can authorize each other to communicate. We distinguish between two types of authorizations: Parcel Delivery Authorization (PDA) and Cargo Delivery Authorization (CDA). The PDA authorization is generated by an Endpoint to allow another Endpoint to communicate with it. The CDA authorization is generated by a Private Gateway to allow Internet Gateways to communicate with it. More details can be found in [3].

2.3 Awala channel session protocol

This protocol is used to encrypt the content encapsulated in a RAMF message to provide end-to-end encryption. Authenticated encryption is being used, specifically AES in GCM mode. The messages are also signed, to authenticate the sender. The protocol is based on the extended triple Diffie-Hellman protocol and the double ratchet algorithm from the Signal project [7], [8].

In the Awala channel session protocol, Alice and Bob derive a key `K` using Diffie-Hellman over ephemeral keys. The key `SK` with which they encrypt messages is calculated by applying a Key Derivation Function (KDF) over the key `K`. When Alice (and Bob) applies Diffie-Hellman to encrypt a message, she uses the last public key received from Bob (from Alice). The encrypted message is attached (among other things) to the public key corresponding to the private key used by Alice (by Bob) while applying Diffie-Hellman. The encrypted message is then signed with the Alice's (Bob's) identity key following the RAMF serialization. The nonce for AES in the GCM mode is generated following the RBG construction (Section 8.2.2 from [9]).

The protocol is described in details in [2]. Discussion of the guarantees of this protocol is deferred to the next section.

3 Analysis

3.1 Awala channel session protocol

In this section, we discuss the security guarantees provided by the Awala Channel Session Protocol combined with the RAMF serialization. It is important to note, however, that this analysis is not based on formal verification of the protocol but rather on a high-level evaluation, similarly to sections 4 and 6 from the Signal extended triple Diffie-Hellman protocol and double ratchet algorithm documentation, described in [7] and [8] respectively.

3.1.1 Authentication

As mentioned earlier, parties authenticate each other using signatures. That is, every message encrypted with the Awala channel session protocol is attached to its signature generated by the identity key of the node sending the message. The RAMF serialization also includes the necessary certificates following the Awala PKI described in section 2 to authenticate the message. The public key that the receiver uses to verify the signature must be the one corresponding to the CN of the leaf certificate (or the self-signed certificate if no certificate path is sent). The CN, as mentioned earlier, is the hash of the public key itself.

From a performance standpoint, signing every message leads to reduced efficiency in terms of bandwidth and computation, especially when compared to end-to-end messaging protocols that rely on symmetric key cryptography. However, it is important to note that given the nature of the network, namely an overlay network designed for use when the Internet is unavailable, the primary concern is maintaining node communication rather than optimizing the performance of an end-to-end encryption protocol.

3.1.2 Forward and future secrecy

Forward and future secrecy, as defined in [10], are guaranteed, thanks to the ratcheting procedure that occurs within the protocol. Note that the ratcheting process happens not only over sessions (i.e. at most one ratchet per session), but also at the message level (i.e. multiple ratchets within the same session). That is, the key by which every message is encrypted is renewed whenever the public key of a new key pair generated by one party is received by the other party. Receiving a new key can occur for every message, which means that within the same session, ratchets are performed multiple times.

Note however that in the case where Bob is a server, Bob's initial Diffie-Hellman key will be the same in all sessions, and with all other nodes. The fact that this key is static and not ephemeral will affect the forward and future secrecy guarantees; if the corresponding private key is compromised, the adversary will be able to decrypt every first message sent and will be sent to Bob by any other node, until Bob's key is replaced, and the node sending messages to Bob updates this key.

Note also that as mentioned in [2] it could be the case that some third-party application using the Awala network only supports unidirectional communication, which means that in these cases a compromise of the private key of the node that is only receiving messages and not sending them will break forward and future Secrecy. Note however that the current Awala-compatible applications are not unidirectional.

3.1.3 Replay attacks

Replay attacks in Awala are avoided thanks to the usage of signatures and timestamping. That is, the RAMF serialization includes message creation and validity times (which are signed), and every message received by a node is saved until its validity time expires. Then, whenever a node receives a message, if the validity time of this message has expired, the node discards the message. If the message is valid, it checks whether this same message was sent before, and if so, discards it.

3.1.4 Deniability

Due to how the identity key of a node is used in the Awala channel session protocol, deniability in the Awala network is not satisfied. That is, as every message is signed, and the corresponding public key is transmitted in clear following the RAMF serialization, a node that sent a message cannot deny this message in the future. Note however that this property was not aimed for by the Awala network.

3.1.5 Out-of-order messages

Given the nature of the Awala network, the channel session protocol was designed to be tolerant of delays, as messages are most likely to arrive out of order.

Each message sent through the protocol specifies the public key of the sender, as well as the key pair of the receiver that was involved in deriving the key `SK`. Thus, a node receiving a message will be able to decrypt it, as long as it still stores the private key corresponding to its public key that was used to derive `SK`.

The `IV` used while encrypting (As mentioned above AES in the GCM mode is being used; the encryption process involves an Initialization Vector `IV`) is also forwarded to the receiver so that it can use the correct `IV` to decrypt.

3.1.6 Compromise of identity and ephemeral keys

While the compromise by an adversary of the private key corresponding to an ephemeral key will have limited damage on the protocol (the adversary will be only able to decrypt messages that involved the compromised key in the ratcheting process of the corresponding encryption key), any compromise of the private key corresponding to an identity key will have severe consequences. This is due to the role given to the identity key of a node in the protocol.

Because authenticating messages sent by nodes all along the protocol relies on message signing, if the private key corresponding to the identity key of a node is compromised by an adversary, they will be able to impersonate this node to the other nodes by sending messages on behalf of the victim node as they will be capable of signing messages with the victim node's private key.

The adversary will be able to not only create new sessions with other nodes using the identity of the victim node, but also intervene in ongoing sessions. There is no explicit termination of a session in the protocol, which makes the damage even worse. It is worth noting however that session keys have a validity time, so the adversary will not be able to impersonate the victim node within an ongoing session where the last ephemeral keys used while ratcheting exceeded their validity time.

3.2 PKI

As explained earlier, a certificate in the Awala PKI uses the hash of the public key corresponding to this certificate as its CN. These certificates are used in the RAMF serialization, such as messages exchanged within the Awala channel session protocol. A certificate can be either self-signed, or issued by another node. Certificates issued by other nodes allow forming authorizations, which consist of a chain of certificates. If node `A` attaches an authorization to send messages to node `B` (see section 2 for when authorizations are needed), messages sent from `A` to `B` will be forwarded in the network, otherwise these messages will be discarded. These authorizations have a validity time.

The chain of trust for these authorizations is established similarly to the Internet PKI, in the sense that a certificate has to be signed by the public key of the parent certificate in the certificate path, except for the root certificate which is self-signed. When the necessary checks on an authorization go through, this will attest to the node performing these checks that the node whose id is the CN of the leaf certificate (the leaf certificate is the one supposed to hold the public key of node `A`) is allowed to send messages to the node whose id is the CN of the certificate supposed to hold the public key of node `B` (e.g. for a PDA authorization, this is the second certificate in the certification path from the leaf certificate; see [3]).

However, as the CN of such a certificate does not contain a domain name, associating public keys with domains for the case of Internet Endpoints and Gateways is established differently. That is, the connection parameters of a node (the Internet address of the node, the identity public key of the node, and the initial Diffie-Hellman public key of the node) are retrieved by a user either (1) by the entity operating the node in question publishing the connection parameters on the Internet (not necessarily signed), which is the case for nodes operated by Relaycorp, which includes the default Internet Gateways, or (2) by a node sending the connection parameters of another node to the user (such as in Letro, following the protocol used to retrieve the connection parameters for an Awala Internet Endpoint [11]), or (3) distributing them during the migration protocol (Private node registration protocol [6]) from the default Internet Gateway to the Gateway the user decides to switch to.

In this regard, we recommend reducing trust on nodes when distributing the connection parameters, as distributing wrong connection parameters might lead to severe consequences, such as impersonation attacks. For example, if a user is provided with incorrect connection parameters for an Internet Endpoint and their Internet Gateway is compromised with access to the corresponding private keys, the Gateway could generate Parcels and interact with the user, impersonating the intended Internet Endpoint.

In order to reduce trust, when the connection parameters are retrieved from the ones published on the Internet (case 1), we recommend for instance signing the connection parameters. This will hold the entity in question liable for its actions, whether it is acting maliciously, has been compromised, or is subject to man-in-the-middle attacks.

Similarly, we recommend reducing trust on the node forwarding connection parameters to a user (case 2). Measures for this have already been considered by Relaycorp, such as starting to use an authentication protocol (VeraID [12]) for this purpose.

In addition, the Awala network depends on utilizing the Internet whenever available. The successful delivery of a Parcel, assuming that no malicious nodes were involved in its forwarding, ultimately relies on communication occurring over the Internet at some point in the process. This, along with the potential for users to receive incorrect connection parameters, could severely undermine the network's guarantees. Therefore, we recommend adding a thorough analysis of the Awala network's guarantees to the threat model for the case where there is an extensive lack of Internet access for nodes involved in Parcel forwarding.

3.3 Documentation

We spotted some inconsistencies in the documentation, mainly due to it being outdated. We also provided suggestions to what to add in the documentation, such how the `IV` for AES in GCM mode is generated and whether it is communicated. These inconsistencies and suggestions were addressed in subsequent commits[3]

3.4 Code Review

The analysis also included a code review of some of the relevant repositories [4] [5] [6] [7] [8], targeting the implementation and usage of cryptographic primitives.

This code review took two working days, and we did not identify any findings. However, we still recommend performing a more extensive code review in future, given how large the project is.

[3] Commits between 2024-06-26 and 2024-07-29 from `https://github.com/AwalaApp/specs/commits/master/`.

[4] `https://github.com/relaycorp/relaynet-core-js`

[5] `https://github.com/relaycorp/awala-jvm`

[6] `https://github.com/relaycorp/awala-endpoint-internet-js`

[7] `https://github.com/relaycorp/awala-gateway-internet`

[8] `https://github.com/relaycorp/awala-endpoint-internet`

4 Conclusion and future work

In this report, we provided a cryptographic analysis of the Awala network, specifically the Awala channel session protocol, the Awala PKI, and a limited code review.

The Awala channel session protocol, combined with the RAMF serialization, provides authentication, forward and future secrecy. It also provides a mechanism to address replay attacks and out-of-order messages. However, these guarantees rely heavily on the fact that messages are signed using node identity key. As such, any compromise of these keys will result in severe consequences. For future work, we recommend performing formal analysis on the protocol, similar to [14], [15] and [16].

The Awala PKI governs messages sent in the RAMF serialization, and implements measures to restrict communication between only the nodes that have authorized each other (if an authorization is needed), thanks to issuing authorizations through certificates. For future work, we recommend reducing trust on the nodes when it comes to linking domain names with public keys, as this might become a point of failure in the chain of trust. Note however that some measures have already been considered for future implementation, such as the usage of VeraID.

In the implementation and usage of the cryptographic primitives within some of the repositories implementing the Awala network, we did not identify any findings during our code review. However, as we only spent two working days on this, we recommend performing a more in-depth code review, targeting all the repositories.

Finally, we recommend performing more audits of the Awala network in general in the future.

5 Acknowledgement

We would like to thank Gus Narea from Relaycorp for his prompt answers, and the extensive explanations he provided about the Awala network throughout the analysis.

6 Bibliography

[1] *Awala Network*. https://awala.network/. Accessed: 2024-06-26.

[2] *Awala Channel Session Protocol*. https://specs.awala.network/RS-003. Accessed: 2024-06-26.

[3] *Awala PKI*. https://specs.awala.network/RS-002. Accessed: 2024-06-26.

[4] *Awala repositories*. https://github.com/relaycorp. Accessed: 2024-06-26.

[5] *Awala Threat Model*. https://specs.awala.network/RS-019. Accessed: 2024-06-26.

[6] *Awala Documentation*. https://specs.awala.network/RS-000. Accessed: 2024-06-26.

[7] *Signal X3DH Key Agreement Protocol*. https://signal.org/docs/specifications/x3dh/. Accessed: 2024-06-26.

[8] *Signal Double Ratchet Algorithm*. https://signal.org/docs/specifications/doubleratchet/. Accessed: 2024-06-26.

[9] *GCM with an RBG-based Construction*. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf. Accessed: 2024-06-26.

[10] *SoK: Secure Messaging*. https://www.ieee-security.org/TC/SP2015/papers-archived/6949a232.pdf. Accessed: 2024-06-26.

[11] *Connection params retrieval using the Letro application*. https://docs.relaycorp.tech/letro-server/connection-params-retrieval. Accessed: 2024-06-26.

[12] *VeraId protocol*. https://veraid.net/. Accessed: 2024-06-26.

[13] *Awala Abstract Message Format*. https://specs.awala.network/RS-001. Accessed: 2024-06-26.

[14] *Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, Douglas Stebila. Formal Security Analysis of the Signal Messaging Protocol*. https://eprint.iacr.org/2016/1013.pdf. Accessed: 2024-06-26.

[15] *Benjamin Dowling, Paul Rösler, and Jörg Schwenk. Flexible Authenticated and Confidential Channel Establishment (fACCE): Analyzing the Noise Protocol Framework*. https://casa.rub.de/fileadmin/img/Publikationen_PDFs/2020_Flexible_Authenticated_and_Confidential_Channel_Establishment__fACCE__Analyzing_the_Noise_Pr Accessed: 2024-06-26.

[16] *Benjamin Lipp, Bruno Blanchet, Karthikeyan Bhargavan. A Mechanised Cryptographic Proof of the WireGuard Virtual Private Network Protocol*. https://inria.hal.science/hal-02396640/document. Accessed: 2024-03-07.

Appendix 1 Testing team

Younes Talibi Alaoui	Younes received his PhD. in cryptography from KU Leuven, with a focus on privacy-preserving technologies such as multi-party computation. Since his PhD, he has worked as a cryptographer across various companies.
Melanie Rieback	Melanie Rieback is a former Asst. Prof. of Computer Science from the VU, who is also the co-founder/CEO of Radically Open Security.

Front page image by Slava (https://secure.flickr.com/photos/slava/496607907/), "Mango HaX0ring",
Image styling by Patricia Piolon, https://creativecommons.org/licenses/by-sa/2.0/legalcode.