MessageVortex

Transport Independent Messaging anonymous to 3^{rd} Parties

Inauguraldissertation
zur
Erlangung der Würde eines Doktors der Philosophie
vorgelegt der
Philosophisch-Naturwissenschaftlichen Fakultät
der Universität Basel
von

Martin Gwerder (06-073-787)
von Glarus GL

September 15, 2016

Genehmigt von der Philosophisch-Naturwissenschaftlichen FakultÃd’t
Auf Antrag von
Prof. Dr. Christian F. Tschudin
Prof. Dr. Heiko Schuldt
Basel, FIXME datum der fakultÃd’tsversammlung FIXME Signature line with Name of Dean of Faculty below

Contents

List of Tables

List of Figures

1. Introduction

Numerous events in present and past have shown that data is broadly collected in the internet. Wether this is a problem or not may be a disputable fact. Undisputed is however that if data is not handled with care peoples are accused with numerous “facts” that are more than questionable. To show that this may happen even under complete democratic control we might refer to events such as the “secret files scandal?? (or “Fichenskandal”) in Switzerland. In the years from 1900 to 1990 Swiss government collected 900’000 files in a secret archive (covering roughly 10% of the natural and juristic entities within Switzerland at that time).

A series of similar attempts to attack privacy on a global scale have been discovered by whistle blower Edward Snowden. The documents leaked in 2009 by him claim that there was a data collection starting in 2010. Since these documents are not publicly available it is hard to to prove claims based on these documents. However – due to the fact that the documents were screened by a significant number of journalists spanning multiple countries, the information seems credible.

According to these documents (verified by NRC) NSA infiltrated more than 50k computers with malware to collect classified information. They furthermore infiltrated Telecom-Operators (executed by british GCHQ) such as Belgacom to collect data and targeted high member of governments even in associated states (such as the germans president mobile phone). A later published shortened list of “selectors” showed 68 telephone and fax numbers targeting enconomy, finance and agricultural parts of the german government.

This list of events shows that big players are collecting and storing vast amounts of data for analysis or possibly future use. The list of events shows also that the use of this data has in the past been at least partially questionable. As a part of possible counter measures this work analyses the possibility of using state of the art technology to minimize the information footprint of a person on the internet.

We leave a vast information footprint in our daily communication. On a regular email we disclose everything in an “postcard” to any entity on its way. Even when encrypting a message perfectly with today’s technology (S/MIME[5] or PGP[4]) leaves at least the originating and the receiving entity disclosed or we rely on the promises of a third party provider which offers a proprietary solution. Even in those cases we leak informations such as “message subject”, “frequency of exchanged messages”, “size of messages”, or “client beeing used”. A good anonymity protocol has therefore far more attributes to cover than the message itself. It includes beside the message all metadata and the traffic flows. Furthermore a protocol anonymising messages should not rely on the trust of infrastructure other than the infrastructure under control of the sending or receiving entity.

Any central infrastructure is bound to be of special interest to anyone gathering data concerning the using entities of such a protocol. It furthermore may be manipulated in order to attack the messages or their flow. So central infrastructure has to be avoided.

In this work a new protocol is designed to allow message transfer through existing communication channels. These messages should be unobservable to any third party. This unobservability does not only cover the message itself but all metadata and flows associated with it. The protocol should be designed in such a way so that it is capable to use any type of transfer protocol. It should be even possible to switch protocols while have messages in transfer to allow media breaches (at least on a protocol level).

The new protocol should allow safe communication without the need of trusting the underlying transport media. Furthermore it is desirable that the usage of the protocol itself is possible without altering the immediate behaveour of the transport layer. That way it is possible to use the transport layers normal traffic to increase the noise in which information has to be searched.

This work splits into multiple parts. In the first part we are collecting available researches and technologies. We emphasize in all technologies on the strength and weaknesses relevant to this work. In the second part we reassemble the parts to a new protocol. In the third part we analyse the protocol for fitness of the purpose. We try to find weaknesses and work out recommendations for the protocol usage. In the last part we discuss the results and try to summarize the findings. Try to elaborate to what extend the protocol fullfills the requirements of this work.

2. Main Research Question

The main topic of this thesis was defined as follows:

- Is it possible to have specialized messaging protocol used in the internet based on “state of the science” technologies offering a high level of unlikability (sender and receiver anonymity) towards an adverser with a high budget and privileged access to internet infrastructure?

Based on this main question there are several sub questions grouped around various topics:

1. What technologies and methods can be used to provide anonymity against a potential adverser. And what technologies may be used in order to mitigate already known weaknesses of this technologies?
2. How can entities utilizing these technologies and methods be attacked and what measures are available to circumvent such attacks?
3. How can attacks targeting anonymity of a sending or receiving entity be mitigated by design?

Part I.

Methodes

In this part of the dissertation we collect requirements, definitions, methods, and existing research relevant to the topic of this thesis. We start with collecting We start with researching existing technologies

3. Requirements for an anonymising Protocol

In the following sections we try to elaborate the main characteristics of the anonymising protocol. The main goal of the protocol is to enable Freedom of speech as defined in Article 19 of the International Covenant on Civil and Political Rights (ICCPR)[12].

> everyone shall have the right to hold opinions without interference

and

> everyone shall have the right to freedom of expression; this right shall include freedom to seek, receive and impart information and ideas of all kinds, regardless of frontiers, either orally, in writing or in print, in the form of art, or through any other media of his choice.

We imply that not all participants in the internet share this value. As of sept 1st 2016 Countries such as China (signatory), Cuba (signatory), Quatar, Saudi Arabia, Singapore, United Arab Emirates, or Myanmar did not ratify the ICCPR. Other countries such as United States or Russia did either put local laws in place superseding the ICCPR or made reservations rendering parts of it ineffective. We may therefore safely assume that freedom of speech is not given in the internet as a lot of countries explicitly supersede them.

We always have to keep in mind that we have no control over the flow of data packets in the internet. Packets may pass though any point of the world. It is not even possible to detect what way has a packet taken. The common network diagnostic tool `tracert` respectively `traceroute` tries to figure that out. But neither can a route of a packet beeing sent forseen nor can it be measured while or after sending. This is due to the fact that all routers along the way only decide for the next hop of a packet.

As an example of the problems analysing a packet route we may look at `traceroute`. The mapage of traceroute of Linux tells us that `traceroute` uses UDP, TCP, or ICMP packets with a short TTL and analyses the IP of the peer sending an TIME_EXCEEDED (message of the ICMP protocol). This information is then collected and shown as a route. This route may be completely wrong. Some of the possible cases are described in the manpage.

The output of `traceroute` is therefore not a reliable indication of route. Since routes do not have to be static and may be changed or even be alternating the output represents in the best case a snapshot of the current routing situation. We cannot be sure that data packets we are sending are passing only through countries accepting the ICCPR to the full extend.

3.1. Anonymizing and unlinking

If we are unable to limit the route of our packets through named jurisdictions we must protect ourselves from unintentionally breaking the law of a foreign country. Therefore we need to be anonymous when sending or receiving messages. Unfortunately most transport protocols (in fact almost all of them such as SMTP, SMS, XMPP or IP) use a globaly unique identifier for senders and receivers which are readable by any party which is capable of reading the packets.

As a result anonymisation of a sender or a receiver is not simple. If messages are being sent through a relay at least the original sender might be concealed (Sender anonymity). By combining it with encryption we may even achieve a very simple form of sender and receiver pseudonymity. If cascading more relay like infrastructures and combining it with cryptography we might even achieve sender and receiver anonymity. This approach has however serveral downsides (see 5.10.1.4 and 5.10.1.2 for details) and is easily attackable.

3.2. Censorship Resistant

In our scenario in 3 we defined the adverser as someone with superior access to the internet and its infrastructure. Such an adverser might attack a message flow in serveral ways:

- Identify sender
- Identify recipient
- Read messages passed or extract meta information
- Disrupt communication fully or partially

We furthermore have to assume that all actions taken by a potential adverser are not subject to legal prosecution. This is due to the fact that an adverser trying to establish censorship may be part of a jurisdictions government. We may safely assume that there are legal exceptions in some jurisdiction for such entities.

In order to be able to withstand an adverser outlined above the protocol needs certain attibutes. The message content needs to be unidentifiable by attributes or content. Whereas “Attributes” include meta information such as frequency, timing, message size, sender, protocol, ports, or recipient (list not conclusive).

3.3. Reliable

Any message sending protocol needs to be reliable in its functionality. If means of message transport are unreliable users tend to use different means for communication

3.4. Available

4. Existing Transport Layer Protocols

4.1. HTTP

The HTTP protocol allows message transfer from [illegible]d to a server and is specified in RFC2616
. It is not suitable as a communication protocol for messages due to the lack of notifications. There are some extensions which would allow such communications (such as WebDAV) but in general even those are not suitable as they require a continous connection to the server in order to get notifications. Having a "rollup" of notifications when connecting is not doable but could be programmed.

http server listen on standard ports 80 or 443 and listen to incomming connects. The port 443 is equivalent to the port 80 except for the fact that it has a wrapping encryption layer (usually TLS). The incomming connects (requests) must offer a header part and may contain a body part which would be suitable for transferring messages to the server. The reply onto this request is transferred over the same TCP connection containing the same two sections.

The main disadvantage in terms as message transport protocol is that this protocol is not symetrically. This means that a server is always just "serving requests" and not sending actively information to peers. This Request-Reply design makes the protocol unsuitable for message transport.

4.2. MQTT

MQTT is an ISO standard (ISO/IEC PRF 20922:2016) and was formerly called MQ Telemetry Transport. The protocol runs by default on the two ports [illegible]83 and 8883 and may be encrypted with TLS.

4.3. Advanced Message Queuing

4.4. Constrained Application Protocol

4.5. Web Application Messaging Protocol

4.6. XMPP (jabber)

4.7. SMTP

4.8. SMS

SMS c[illegible]bility was introduced in the SS7 protocol
. This protocol allows the message transfer of messages not bigger than 144 character. Due to this restriction in size it is unlikely to be suitable for this type of communication as the keys beeing required are already sized similarly leaving no space for Messages or routing information.

4.9. MMS

The Multimedia Messaging Service (MMS) is maintained by 3GPP (3rd Generation Partnership Project)

5. Existing Research and Implementations on the Topic

5.1. Anonymity

As Anonymity we take the definition as specified in [9].

> Anonymity of a subject means that the subject is not identifiable within a set of subjects, the anonymity set.[1]

and

> Anonymity of a subject from an attacker's perspective means that the attacker cannot sufficiently identify the subject within a set of subjects, the anonymity set.[1]

Whereas the anonymity set is defined as the set of all possible subjects.

Especially the anonymity of a subject from an attacker's is very important to this paper.

5.1.1. k-Anonymity

k-anonymity is a term introduced in [1]. This work claims that no one might be held responsible for an action if the action itself can only be identified as an action which has been taken by one unidentifiable entity out of k entities.

The Document distinguishes between *Sender k-anonymity* where the sending entity can only be narrowed down to a set of k entities and *Receiver k-anonymity*

5.1.2. ℓ-Diversity

In [8] an extended model of k-anonymity. According to the authors it is possible to break a k-anonymity set if there is additional Information available which may be merged into a data set so that a special entity can be filtered from the k-anonymity set. In other words if an anonymity set is to tightly specified a single additional background information might be sufficient to identify a specific entity in an anonymity set.

While it might be arguable that a k-anonymity in which a member is not implicitely k-anonymous still is sufficient for k-anonymity in its sense the point made in this work is definitely right and should be taken into account.

[1] footnotes omitted in quote

Their approach is to introduce an amount of invisible diversity into k-anonymous sets so that simple background knowledge is no longer sufficient to isolate a single member.

5.1.3. t-Closeness

While ℓ-diversity protects the identity of an entity it does not prevent information gain. A subject which is in a class has the same attributes. This is where t-closeness[7] comes into play. t-closeness is defined as follows:

> An equivalence class is said to have t-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the attribute in the whole table is no more than a threshold. A table is said to have t-closeness if all equivalence classes have t-closeness.

5.2. Zero Trust

Zero trust is not a truly researched model in systems engineering. It is however widely adopted.

We refer in this work to the zero trust model when denying the trust in any infrastructure not directly controlled by the sending or receiving entity. This distrust extends especially but not exclusively to the network transporting the message, the nodes storing and forwarding messages, the backup taken from any system, and software, hardware, and operators of all systems not explicitly trusted.

5.3. Pseuidonymity

As Pseudonymity we take the definition as specified in [9].

> A pseudonym is an identifier of a subject other than one of the subject's real names. The subject which the pseudonym refers to is the holder of the pseudonym. A subject is pseudonymous if a pseudonym is used as identifier instead of one of its real names.[1]

5.4. Undetectability

As undetectability we take the definition as specified in [9].

> Undetectability of an item of interest (IOI) from an attacker's perspective means that the attacker cannot sufficiently distinguish whether it exists or not.[2]

5.5. Unobservability

As unobservability we take the definition as specified in [9].

> Unobservability of an item of interest (IOI) means
>
> - undetectability of the IOI against all subjects uninvolved in it and
> - anonymity of the subject(s) involved in the IOI even against the other subject(s)
>
> involved in that IOI.

This part is very important. As we are heading for a censorship resistant solution unobservability is a key. As mentioned in this paper unobservability raises the required attributes again (⇒ reads "implies"):

$$\begin{aligned} Censorship\ resistance &\Rightarrow unobservability \\ unobserability &\Rightarrow undetectability \\ unobserability &\Rightarrow anonymity \end{aligned}$$

So this m[illegible]s that we have to use an undetectable

5.5.1. Ephemeral Identity

In this work we use accounting on various levels. While we are dealing with anonymity accounting has still to be linked to some kind of identity for this reason we are introducing a term called "ephemeral identity".

> A Ephemeral identity is a temporary identity which is defined by the following attributes:
>
> - It is an artificial identity
> - It is only used for a short timespan
> - It is not linkable to another identity

The key in this definition is the last point is crucial and at the same time hard to achieve.

[2] footnotes omitted in quote

5.6. Single Use Reply Blocks and Multi Use Reply Blocks

The use of single use reply blocks were first introduced by Chaum in [2]. The concept is that we have in our case a routing block which mi[illegible] be used up to n times ($0 < n < FIXME$)

5.7. Censorship

5.7.1. Censorship Resistant

5.7.2. Parrot Circumvention

5.7.3. Censorship Circumvention

5.7.3.1. Covert Channel

5.7.3.2. Spread Spectrum

5.8. Cryptography

5.8.1. Symmetric Encryption

5.8.2. Asymmetric Encryption

5.8.2.1. RSA

[Rivest:1978:MOD:359340.359342]

5.8.2.2. Elliptic Curve Cryptogaphy

5.8.3. Homomorphic encryption

5.9. Routing

5.9.1. Mixing

5.9.2. Onion Routing

5.9.3. Crowds

5.9.4. Mimic routes

5.10. System Implementations

The following sections describe

5.10.1. Concepts

5.10.1.1. DC Networks

5.10.1.2. MIX Networks

5.10.1.3. Onion Routing

5.10.1.4. Remailer

5.10.2. Implementations

The following sections emphasize on implementations of anonymising (and related) protocols regardless of their usage in the domain of messaging. It is a list of system classes or their specific implementations together with a short analysis of strength and weaknesses. Wherever possible we try to refer to original sources. This is however not always possible since some of these systems are no longer in use.

5.10.2.1. Pseudonymous Remailer

5.10.2.2. Babel

5.10.2.3. Cypherpunk-Remailer

5.10.2.4. Mixmaster-Remailer

5.10.2.5. Mixminion-Remailer

5.10.2.6. Crowds

5.10.2.7. Tarzan

5.10.2.8. Herbivore

5.10.2.9. Dissent

5.10.2.10. P5

5.10.2.11. Gnutella

5.10.2.12. Gnutella2

5.10.2.13. Freenet

5.10.2.14. Darknet

5.10.2.15. Sneakernet

5.10.2.16. Hordes

5.10.2.17. Salsa

5.10.2.18. Hydra-Onion

5.11. Known Attacks

In the following sections we emphasize on possible attacks to an anonymity preserving protocols. In the following sections we describe classes of attacks. These attacks may be used to attack the anonymity of any entity involved in the message channel. In a later stage we test the protocol for immunity against these classes of attacks.

5.11.1. Broken Encryption Algorithms

5.11.2. Attacks Targeting Anonymity

5.11.2.1. Hotspot Attacks

5.11.2.2. Message Tagging and Tracing

5.11.2.3. Side Channel Attacks

5.11.2.4. Timing Attacks

5.11.2.5. Sizing Attacks

5.11.2.6. Bugging Attacks

5.11.3. Denial of Service Attacks

5.11.3.1. Censorship

Where as traditional censorship is widely regarded as selective information filtering and alteration a very repressive censorship can even include denial of information flows in general. Any anonymity system not offering the possibility to hide in legitimate information flows is therfore not censorship resistant.

5.11.3.2. Credibility Attack

Another type of DoS attack is the credibility attack. While not necessarily a technical attack it is very effective. A system not having a sufficiently big user base is offering thus a bad level of anonymity due to the fact that the anonymity set is too small or the traffic concealing message flow is insufficient.

In a credibility attack a systems reputation is degraded in such a way that the system is no longer used. This may be achieved in serveral ways. This is usually done by reducing the reputation of a system.

This may be achieved in several ways. Examples:

- Disrupt functionality of a system.
- Publicly dispute the effectiveness of a system.
- Reduce the effectiveness of a system.
- Dispute the credibility of the system founders.
- Dispute the credibility of the infrastructure.

6. Applied Methodes

Dased in the findings in this chapter we used the following methodology:

1. Identify problem hotspots for a new protocol
2. Design a protocol which adresses the previously identified hotspots
3. Build a protocol prototype
4. Analyse the protocol for weaknesses using attack schemes
   a) Tagging/Bugging attacks
   b) Tracing attacks

6.1. Problem Hotspots

6.2. Protocol Design

6.3. Protocol Analysis

Part II.

Results

To verify the hypothesis made in this paper, and to analyse properties of the protocol in a real world scenario a library was implemented in Java which was capable of handling all message packets and the routing stack as a whole. The following paragraphs describe the protocol developed in general as a generic approach. Appendix IV gives the full ASN.1 representation of the protocol.

It is important to notice that ASN.1 has no mean to express encrypted structures. Due to this fact we defined all encrypted fields as `OCTET STRING`. The protocol offers according to the ASN.1 the possibility to store onionized information in an unencrypted form. This is meant for debuing purposes. At no point this should be used in a production environment.

The protocol described in the next chapter is independent from routing. At the moment capabilities include SMTP and XMPP. The protocol may be extended by adding new transport layer capabilities and their addressing schemes.

7. MessageVortex - Transport Independent Messaging anonymous to 3rd Parties

7.1. Protocol Description

7.2. Accounting

7.3. Message Flows

7.4. Considerations for Building Messages

7.4.1. Timing of messages

7.4.2. Building Diagnostic Paths

7.4.2.1. Implicit Diagnostic

7.4.2.2. Automatic Explicit Diagnostic

7.4.2.3. On-Demand Explicit Diagnostic

7.5. Real World Considerations

This approach is heavily dependent of the transport protocol and builds on top a new obfuscating/routing layer. For this system to become a real peer-to-peer approach some additional quirks are required. A message-Vortex-Account needs always an active routing handler. This routing handler may be introduced by new server capabilities or by having a device handling the routing from the client side. For this reason we built a RaspberryPi appliance capable of connecting to one (or more) accounts fetching incomming mails, analysing them and reroute them if necessary. Although the system is designed to be run on a RaspberryPi the software might be installed to any Java capable client. The RaspberryPi is just an affordable lightweight device which offers all required capabilities.

8. Security Analysis

9. Additional Considerations

9.1. Storage of Messages and queues

The storage of messages sent though MessageVortex should be handled with great care. It seems on the first sight a good idea to merge all messages in a globally available storage such as the mail account of the receiving entity. However – In doing so we would discover the message content to the providing party of a mail account. Since we handled the message with great care and tremendous costs up until this point it would be careless doing so.

Storing them in a localized and receiving entity controlled storage is definitely a good idea but leaves security considerations like a backup possibly to an end user. This might be better but in effect a questionable decision. There is however a third option. By leaving the message unhandled on the last entity of the MessageVortex chain we may safely backup the data without disclosing the message content. Merging the content then dynamically through a specialized proxy would allow the user tu have a unified view on his without compromising the security.

9.2. Economy of transfer

Part III.

Discussion

10. Anonymity

10.1. Effects of anonymous communication on behaveour

[10]

Part IV.

Appendix

ASN.1 representation of the protocol

Listing 1: ASN.1 representation of the protocol

```
-- FIXME ADAPT NeoScrypt

-- encryption: as specified in the key. If not specified default mode is ECB and default padding is PKCS1Padding

-- States: Tuple()=Value() [vallidity; allowed operations] {Store}
-- - Tuple(identity)=Value(messageQuota,transferQuota,sequence of Routingblocks for Error Message Routing) [validity; Requested at creation; may be extended upon request] {identityS
-- - Tuple(Identity,Serial)=maxReplays ['valid' from Identity Block; from First Identity Block; may only be reduced] {IdentityReplayStore}

Message-Blocks DEFINITIONS EXPLICIT TAGS ::=
BEGIN

  -- define constants
  maxSerial             INTEGER ::= 4294967295    -- maximum serial number
  maxChunkSize          INTEGER ::= 4294967295    -- maximum size of a message chunk
  maxNumberOfReplays    INTEGER ::= 65535         -- maximum number of replys
  maxNumberOfRequests   INTEGER ::= 8             -- maximum number of administrative requests

  Message     ::= SEQUENCE {
    header          IdentityBlock,
    blocks          CHOICE {
        encrypted  [1101] OCTET STRING,     -- contains encrypted UnencryptedBlocks structure; Decryption key is in identity block [decryptionKey]
        -- it is not allowed to use plain in production environments; This is for testing and analysis only
        plain      [1102] UnencryptedBlocks -- should not be used except for internal diagnostic purposes
    }
  }

  -- represents the identity of the rights owner
  IdentityBlock ::= SEQUENCE {
    headerKey             [1000] OCTET STRING OPTIONAL,  -- contains SymmetricKey encrypted with recipient nodes public key
    identityBlock         CHOICE {
        encrypted   [1001] OCTET STRING,
        -- it is not allowed to use plain in production environments; This is for testing and analysis only
        plain       [1002] IdentityPayloadBlock,
    },
    identitySignature OCTET STRING   -- contains signature of Identity [as stored in identityBlock;signed identityBlock without Tag]
  }

  IdentityPayloadBlock ::= SEQUENCE {
    -- Public key of the identity representing this transmission
    identityKey         AsymmetricKey,

    -- serial identifying this block
    serial              INTEGER (0..maxSerial),

    -- number of times this block may be replayed (Tuple is identityKey,serial while
    maxReplays          INTEGER (0..maxNumberOfReplays),

    -- subsequent Blocks are not processed before valid time.
    -- Host may reject too long retention. Recomended validity support >=1Mt.
    valid               UsagePeriod,

    -- represents the chained secret which has to be found in subsequent blocks
    -- prevents reassembly attack
    forwardSecret [2000] ChainSecret OPTIONAL,

    -- contains SymmetricKey encrypted with private key of identityKey
    -- encryption is done as proof of identity (identity hijack protection)
    decryptionKey       OCTET STRING,   -- contains (encrypted) DER encoded ASN1BitString with key representation

    --  contains the MAC-Algorithm used for signing
    hash                MacAlgorithm,

    -- contains administrative requests such as quota requests
    requests            SEQUENCE (SIZE (0..maxNumberOfRequests)) OF HeaderRequest ,

    -- padding and identitifier required to solve the cryptopuzzle
    identifier     [2001] INTEGER (0..maxSerial) OPTIONAL,
    padding        [2002] OCTET STRING OPTIONAL  -- This is for solving crypto puzzles
  }

  UnencryptedBlocks ::= SEQUENCE {
    -- contains routing information (next hop) for the payloads
    -- FIXME how handle multiple payloads
    routing      [3000] RoutingBlock OPTIONAL,

    -- contains encrypted log data of the data traveling
    routingLog   [3010] RoutingLogBlock OPTIONAL,

    -- contains replys to header requests (eg. quota and identity handling)
    reply        [3020] ReplyBlock OPTIONAL,

    -- contains the actual payload
    payload      [3100] SEQUENCE (SIZE (0..128)) OF PayloadChunk
  }


  -- represents the building and sending process for the next hop
  RoutingBlock ::= SEQUENCE {

    -- contains the next recipient in sequence
    recipient NodeSpec,

    -- contains the period when the payload should be processed
    -- Router might refuse to long queue retention
    -- Recommended support for retention >=1h
    queueTime UsagePeriod,

    nextHop             SEQUENCE (SIZE (0..128)) OF NextHopBlock,   -- encrypted next RoutingBlocks for the payload

    -- contains the secret of the identity block (if any)
    forwardSecret       [111]   ChainSecret OPTIONAL,

```

```
    -- contains a routing block which may be used when sending error messages back to the quota owner
    -- this routing block may be cached for future use
    replyBlock [131]     RoutingBlock OPTIONAL,

    -- This section is required if payload is routed with a prebuilt RB (
    -- Messages MAY always request recompression (otherwise the message is identifyable from a non reply routing block)
    decryptionKey        [200] SEQUENCE (SIZE (1..2)) OF SymmetricKey OPTIONAL,
    encryptionKey        [201] SymmetricKey OPTIONAL,

    -- contains information for building replys
    cascade              [300] SEQUENCE(SIZE (0..255)) OF CascadeBuildInformation
  }

  NextHopBlock ::= SEQUENCE {
    nextIdentityBlock     [13100] OCTET STRING,
    nextRoutingBlock      [13200] OCTET STRING,
    nextReplyBlock        [13300] OCTET STRING OPTIONAL,
    nextErrorReplyBlock   [13400] OCTET STRING OPTIONAL
  }

  PayloadOperation ::= SEQUENCE{
    operation             PayloadType,
    thisid                [12010] SEQUENCE (0..128) OF INTEGER (0..65535)
  }

  PayloadType ::= CHOICE {
        randomPayload     [100] RandomPayloadOperation,
        splitPayload      [150] SplitPayloadOperation,
        mergePayload      [200] MergePayloadOperation,
        xorPayload        [250] XorPayloadOperation,
        encryptPayload    [300] EncryptPayloadOperation,
                ...
  }

  CascadeBuildInformation ::= SEQUENCE {
    encryptionKey         SymmetricKey,
    secret                ChainSecret,
    payloadOp             PayloadOperation,
    ...
  }

  PercentSizeType ::= SEQUENCE {
    fromPercent           REAL (0..100),
    toPercent             REAL (0..100)
  }

  AbsoluteSizeType ::= SEQUENCE {
    fromAbsolute          INTEGER (0..maxChunkSize),
    toAbsolut             INTEGER (0..maxChunkSize)
  }

  SizeType ::= SEQUENCE{
        reference                 INTEGER (0..65535),
        size CHOICE {
                percent       [15001] PercentSizeType,
                absolute      [15101] AbsoluteSizeType
        }
  }

  RandomPayloadOperation ::= SEQUENCE {
    size                  SizeType
  }

  SplitPayloadOperation ::= SEQUENCE {
    originalId                           INTEGER (0..65535),
    firstSize                  SizeType,
        secondId                                 INTEGER (0..65535)
  }

  MergePayloadOperation ::= SEQUENCE {
    -- FIXME
  }

  XorPayloadOperation ::= SEQUENCE {
    -- FIXME
  }

  EncryptedRoutingLogBlock ::= OCTET STRING -- contains symmetrically encrypted RoutingLogBlock
  RoutingLogBlock ::= SEQUENCE {
    routingLog             SEQUENCE (SIZE (0..16)) OF RoutingLog,
    nestedRoutingInformationBlock EncryptedRoutingLogBlock
  }

  RoutingLog ::=SEQUENCE   {
    nodeIdentifier      IA5String,
    time                GeneralizedTime,
    code                ErrorCode,
    information         IA5String
  }

  IdentityReplayStore ::= SEQUENCE {
        replays          SEQUENCE (SIZE (0..4294967295)) OF IdentityReplayBlock
  }

  IdentityReplayBlock ::= SEQUENCE {
    identity              AsymmetricKey,
    valid                 UsagePeriod,
        replaysRemaining          INTEGER (0..4294967295)
  }

  IdentityStore ::= SEQUENCE {
    identities SEQUENCE (SIZE (0..4294967295)) OF IdentityStoreBlock
  }

  IdentityStoreBlock ::= SEQUENCE {
    valid                 UsagePeriod,
    messageQuota          INTEGER (0..4294967295),
    transferQuota         INTEGER (0..4294967295),
    identity              [1001] AsymmetricKey OPTIONAL,    -- if omitted this is a node identity
    nodeAddress           [1002] NodeSpec       OPTIONAL,   -- if ommited own identity key
    nodeKey               [1003] SEQUENCE OF AsymmetricKey OPTIONAL,    -- Contains the identity of the owning node; May be ommited if local node
    routingBlocks         [1004] SEQUENCE OF RoutingBlock OTIONAL
    ...
  }

  -- contains a node spec of a routing point
  -- At the moment either smtp:<email> or xmpp:<jabber>
  NodeSpec ::= IA5String

  ChainSecret ::= INTEGER (0..4294967295)
```

```
  -- FIXME define requests
  HeaderRequest ::= CHOICE {
    identity       [0] HeaderRequestIdentity,
    capabilities   [1] HeaderRequestCapability,
    messageQuota   [2] HeaderRequestIncreaseMessageQuota,
    transferQuota  [3] RequestIncreaseTransferQuota,
    quotaQuery     [4] HeaderRequestQueryQuota,
    ...
  }

  ReplyBlock ::= CHOICE {
    identity       [0] ReplyIdentity,
    capabilities   [1] ReplyCapability,
    ...
  }

  HeaderRequestIdentity ::= SEQUENCE {
    identity AsymmetricKey,
    period UsagePeriod,
    ...
  }

  ReplyIdentity ::= SEQUENCE {
    challenge      BIT STRING, -- bit sequence at beginning of hash from encrypted identity block
    hash           MacAlgorithmIdentifier,
    valid          UsagePeriod,
    identifier     INTEGER (0..4294967295),
    ...
  }

  HeaderRequestQueryQuota ::= SEQUENCE {
    identity AsymmetricKey,
    ...
  }

  HeaderRequestIncreaseMessageQuota ::= SEQUENCE {
    identity AsymmetricKey,
    messages INTEGER (0..4294967295),
    ...
  }

  RequestIncreaseTransferQuota ::= SEQUENCE {
    identity AsymmetricKey,
    size     INTEGER (0..4294967295),
    ...
  }

  HeaderRequestCapability ::= SEQUENCE {
    period UsagePeriod,
    ...
  }

  ReplyCapability ::= SEQUENCE {
    cypher            SEQUENCE (SIZE (2..256)) OF CypherSpec,
    maxTransferQuota  INTEGER (0..4294967295),
    maxMessageQuota   INTEGER (0..4294967295),
    supportedProtocol SEQUENCE OF Protocol,
    ...
  }

  CypherSpec ::= SEQUENCE {
      asymmetric AsymmetricAlgorithmIdentifier,
      symmetric  SymmetricAlgorithmIdentifier,
      mac        MacAlgorithmIdentifier
  }

  Protocol ::= ENUMERATED {
    smtp  (100),
    xmmp  (110),
    ...
  }

  ErrorCode ::= ENUMERATED {

    -- System messages
    ok                     (2001),
    transferQuotaStatus    (2101),
    messageQuotaStatus     (2102),

    -- protocol usage failures
    transferQuotaExceeded  (3001),
    messageQuotaExceeded   (3002),
    identityUnknown        (3101),
    messageChunkMissing    (3201),
    messageLifeExpired     (3202),

    -- Mayor host specific errors
    hostError              (5001),
    ...
  }

  PayloadChunk ::= SEQUENCE {
    offset                  INTEGER (0..MAX),
    routingBlockIdentifier  [0]   SEQUENCE (SIZE (0..255)) OF INTEGER, -- FIXME limit integer range
    payload                 [100] OCTET STRING
  }

  -- Compatible to PrivateKeyUsagePeriod taken from RFC3280
  UsagePeriod ::= SEQUENCE {
    notBefore     [0]     GeneralizedTime OPTIONAL,
    notAfter      [1]     GeneralizedTime OPTIONAL
 }

 -- adapted from RFC3280
  SymmetricAlgorithmIdentifier ::= SEQUENCE {
    algorithm              SymmetricAlgorithm,
    parameter              AlgorithmParameters OPTIONAL
  }

  AsymmetricAlgorithmIdentifier ::= SEQUENCE {
    algorithm              AsymmetricAlgorithm,
    parameter              AlgorithmParameters OPTIONAL
  }

  MacAlgorithmIdentifier ::= SEQUENCE {
    algorithm              MacAlgorithm,
    parameter              AlgorithmParameters
  }


  SymmetricAlgorithm ::= ENUMERATED {
```

```
    aes128      (1000),
    aes192      (1001),  -- optional support
    aes256      (1002),
    camellia128 (1100),
    camellia192 (1101),  -- optional support
    camellia256 (1102)
  }

  AsymmetricAlgorithm ::= ENUMERATED {
    rsa         (2000),
    dsa         (2100),
    secp384r1   (2500),
    sect409k1   (2501),
    secp521r1   (2502)
  }

  MacAlgorithm ::= ENUMERATED {
    sha384   (3000),
    sha512   (3001),
    -- FIXME check AEAD
    tiger192 (3100)
  }

  ECCurveType ::= ENUMERATED{
    secp192r1,
    sect163k1,
    sect163r2,
    secp224r1,
    sect233k1,
    sect233r1,
    secp256r1,
    sect283k1,
    sect283r1,
    secp384r1,
    sect409k1,
    sect409r1,
    secp521r1,
    sect571k1,
    sect571r1,
    ...
  }

  AlgorithmParameters ::= SEQUENCE {
    keySize   [10000] INTEGER (0..65535) OPTIONAL,
    curveType [10001] ECCurveType OPTIONAL,
    ...
  }

  -- Symmetric key
  SymmetricKey ::= SEQUENCE {
    keyType SymmetricAlgorithmIdentifier,
    key     OCTET STRING
  }

  -- Asymmetric Key
  AsymmetricKey ::= SEQUENCE {
    keyType       AsymmetricAlgorithmIdentifier,
    publicKey     [1] OCTET STRING,
    privateKey    [2] OCTET STRING OPTIONAL
  }

  pkcs-1    OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 1 }

END
```

Glossary

adverser FIXME

Agent FIXME

EWS FIXME

IMAP IMAP (currently IMAPv4) is a typical protocol to be used between a Client MRA and a Remote MDA. It has been specified in its current version in [3]. The protocol is capable of fully maintaining a server based message store. This includes the capability of adding, modifying and deleting messages and folders of a mailstore. It does not include however sening mails to other destinations outside the server based store.

Item of Interest (IoI) FIXME

LMTP FIXME

Local Mail Store A Local Mail Store offers a persistent store on a local non volatile memory in which messages are beeing stored. A store may be flat or structured (eg. supports folders). A Local Mail Store may be an authoritative store for mails or a “Cache Only” copy. It is typically not a queue.

mail server admin FIXME

MDA An MDA provides an uniform access to a Local Mail Store.

Remote MDA A Remote MDA is typically supporting a specific access protocol to access the data stored within a Local Mail Store .

Local MDA A Local MDA is typically giving local applications access to a server store. This may be done thru an API, a named socket or similar mechanisms.

MRA A Mail receiving Agent. This agent receives mails from a agent. Depending on the used protocol two subtypes of MRAs are available.

Client MRA A client MRA picks up mails in the server mail storage from a remote MDA. Client MRAs usually connect thru a standard protocol which was designed for client access. Examples for such protocols are POP or IMAP

Server MRA Unlike a Client MRA a server MRA listens passively for incomming connections and forwardes received Messages to a MTA for delivery and routing. A typical protocol supported by an Server MRA is SMTP

MS-OXMAPIHTTP FIXME

MSA A Mail Sending Agent. This agent sends mails to a Server MRA.

MTA A Mail Transfer Agent. This transfer agent routes mails between other components. Typically an MTA receives mails from an MRA and forwardes them to a MDA or MSA. The main task of a MTA is to provide reliable queues and solid track of all mails as long as they are not forwarded to another MTA or local storage.

MTS A Mail Transfer Service. This is a set of agents which provide the functionallity tor send and receive Messages and forward them to a local or remote store.

MSS A Mail Storage Service. This is a set of agents providing a reliable store for local mail accounts. It also provides Interfacing which enables clients to access the users mail.

MUA A Mail User Agent. This user agent reads mails from a local storage and allows a user to read existing mails, create and modify mails.

Privacy From the Oxford English Dictionary: “

1. The state or condition of beeing withdrawn from the society of others, or from the public intrest; seclusion. The state or condition of beeing alone, undisturbed, or free from public attention, as a matter of choice or right; freedom from interference or intrusion.
2. Private or retired place; private apartments; places of retreat.
3. Absence or avoidance of publicity or display; a condition approaching to secrecy or concealment. Keeping of a secret.
4. A private matter, a secret; private or personal matters or relations; The private parts.
5. Intimacy, confidential relations.
6. The state of being privy to some act.

”[11, FIXME]
In this work privacy is related to definition two. Mails should be able to be handled as a virtual private place where no one knows who is talking to whom and about what or how frequent (except for directly involved people).

POP POP (currently in version 3) is a typical protocol to be used between a Client MRA and a Remote MDA. Unlike IMAP it is not able to maintain a mail store. Its sole purpose is to fetch and delete mails in a server based store. Modifying Mails or even handling a complex folder structure is not doable with POP

Service FIXME

SMTP SMTP is the most commonly used protocol for sending mails across the internet. In its current version it has been specified in [6].

Storage A store to keep data. It is assumed to be temporary or persistent in its nature.

user FIXME

UBE FIXME

Bibliography

[1] Luis von Ahn, Andrew Bortz, and Nicholas J. Hopper. “k-Anonymous Message Transmission”. In: *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*. Ed. by Vijay Atluri and Peng Liu. ACM Press, Oct. 2003, pp. 122–130. DOI: 10.1145/948109.948128. URL: http://www.abortz.com/papers/k-anon.pdf (cit. on p. 13).

[2] David Chaum. “Untraceable electronic mail, return addresses, and digital pseudonyms”. In: *Communications of the ACM* 24.2 (Feb. 1981), pp. 65–75. URL: http://users.cis.fiu.edu/~carbunar/teaching/cis5372.2013.online/reading/chaum_untraceable.pdf (cit. on p. 14).

[3] M. Crispin. *RFC3501 INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1*. IETF, 2003. URL: http://tools.ietf.org/pdf/rfc3501.pdf (cit. on p. 39).

[4] M. Elkins. *RFC2015 MIME Security with Pretty Good Privacy (PGP)*. IETF, 1996. URL: http://tools.ietf.org/pdf/rfc2015.pdf (cit. on p. 1).

[5] N. Freed and N. Borenstein. *RFC2045 Multipurpose Internet Mail Extensions; (MIME) Part One: Format of Internet Message Bodies*. IETF, 1996. URL: http://tools.ietf.org/pdf/rfc2045.pdf (cit. on p. 1).

[6] J. Klensin. *RFC5321 Simple Mail Transfer Protocol*. IETF, 2008. URL: http://tools.ietf.org/pdf/rfc5321.pdf (cit. on p. 39).

[7] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: *2007 IEEE 23rd International Conference on Data Engineering*. IEEE. 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856. URL: http://www.utdallas.edu/~mxk055100/courses/privacy08f_files/tcloseness.pdf (cit. on p. 13).

[8] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), p. 3. URL: http://www.cs.cornell.edu/~vmuthu/research/ldiversity.pdf (cit. on p. 13).

[9] Andreas Pfitzmann and Marit Hansen. *A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management*. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf. v0.34. Aug. 2010. URL: http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf (cit. on pp. 13, 14).

[10] Tom Postmes, Russell Spears, Khaled Sakhel, and Daphne De Groot. “Social influence in computer-mediated communication: The effects of anonymity on group behavior”. In: *Personality and Social Psychology Bulletin* 27.10 (2001), pp. 1243–1254. DOI: 10.1177/01461672012710001. URL: http://psp.sagepub.com/content/27/10/1243.short (cit. on p. 31).

[11] A. Stevenson. *Oxford Dictionary of English*. Oxford reference online premium. OUP Oxford, 2010. ISBN: 9780199571123. URL: http://www.oed.com (cit. on p. 39).

[12] UNHR. *International Covenant on Civil and Political Rights*. 1966. URL: http://www.ohchr.org/en/professionalinterest/pages/ccpr.aspx (cit. on p. 9).

Index