Future Internet **2011**, *3*, 14-30; doi:10.3390/3010014

OPEN ACCESS

future internet

ISSN 1999-5903

www.mdpi.com/journal/futureinternet

Article

A Distributed Public Key Infrastructure Based on Threshold Cryptography for the HiiMap Next Generation Internet Architecture

Oliver Hanka [1,⋆], Michael Eichhorn [2], Martin Pfannenstein [2], Jörg Eberspächer [1] and Eckehard Steinbach [2]

[1] Institute for Communication Networks, Technische Universität München, 80333 Munich, Germany; E-Mail: joerg.eberspaecher@tum.de

[2] Institute for Media Technology, Technische Universität München, 80333 Munich, Germany; E-Mails: michael.eichhorn@tum.de (M.E.); martin.pfannenstein@tum.de (M.P.); eckehard.steinbach@tum.de (E.S.)

⋆ Author to whom correspondence should be addressed; E-Mail: oliver.hanka@tum.de; Tel.: +49-89-28923506; Fax: +49-89-28923523.

Received: 12 January 2011; in revised form: 29 January 2011 / Accepted: 30 January 2011 / Published: 1 February 2011

Abstract: In this article, a security extension for the HiiMap Next Generation Internet Architecture is presented. We regard a public key infrastructure which is integrated into the mapping infrastructure of the locator/identifier-split addressing scheme. The security approach is based on Threshold Cryptography which enables a sharing of keys among the mapping servers. Hence, a more trustworthy and fair approach for a Next Generation Internet Architecture as compared to the state of the art approach is fostered. Additionally, we give an evaluation based on IETF AAA recommendations for security-related systems.

Keywords: NGI; security; HiiMap; PKI; locator/identifier-split; Threshold Cryptography

---

1. Introduction

Today's Internet architecture faces some well-known limitations and many ongoing research activities exist to define the so-called *Next Generation Internet* (NGI). As security was not a major concern while

developing the current architecture, it is agreed that it needs to be an integral part of future concepts. This is because the Internet has transformed from a communication means where users can exchange files and messages to the basis of today's economy with billions of participants.

To compensate for the lack of security of the original design, many add-ons have been defined over time. Each newly identified security risk resulted in a new fix which was applied to one of the OSI layers. The outcome are many specific security solutions spread over all layers ranging from IPSec encryption at the network layer up to HTTPS in the application layer for example. One of the design goals for a Next Generation Internet architecture is to anchor the security functionality in the network layer to provide a clean and consistent approach for all participants without any need for further application layer security. Especially trust has always been an issue in today's communication infrastructure. Significant additional effort had to be put into establishing a trustworthy communication between two entities. Our approach for a Next Generation Internet is designed to support this from day one.

The *Hierarchical Internet Mapping Architecture* (HiiMap) [1] is a clean-slate approach based on the locator/identifier-split principle [2]. It is a highly scalable and reliable approach with built-in mobility support to also foster network-enabled mobile devices.

Our approach for a Next Generation Internet not only takes into account the emerging needs for a more flexible communication architecture but also a fair and equal mapping scheme distributed over all participating countries [1]. This guarantees the continuity of the Internet as a global means of communication devoid of political and social disputes.

In this article we outline our security approach for the HiiMap architecture and evaluate it according to the recommendations of the Internet Engineering Task Force (IETF) standard for Authentication, Authorization and Accounting (AAA) [3].

The remainder of this article is structured as follows: Section 2 discusses related work before we give a brief overview of the HiiMap architecture in Section 3. In Section 4 we detail the security approach and give an evaluation of it in Section 5. Section 6 concludes this article.

2. Related Work

In this section we discuss some cryptographic fundamentals and related work.

2.1. Security in Communication Systems

There exist general paradigms which have to be fulfilled to assure that a system can be regarded as secure. The Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T) [4] provides guidelines for security aspects in communication systems, which are available as the ITU-T X.805 recommendation. [5]

2.1.1. Dimensions

Within X.805, 8 security dimensions are defined which have to be kept in mind when designing a security-related system:

- Access Control

- Authentication
- Non-repudiation
- Data confidentiality
- Communication security
- Data integrity
- Availability
- Privacy

2.1.2. Threats

On the other hand, there are mainly 4 threats to which a communication system might be exposed:

- Destruction of information and/or other resources
- Corruption or modification of information
- Theft, removal or loss of information and/or other resources
- Disclosure of information.

2.2. Basic Principles of Cryptography

2.2.1. Public Key Infrastructure

Asymmetric crypto-systems use a public and a private key to provide secure communication between network entities. The public key is visible to anyone whereas the private key must not be disclosed to any other party than the key owner itself. A problem of asymmetric crypto-systems is the publishing of the public key.

Today, the exchange of public keys is done via a public key infrastructure (PKI), e.g., defined by the ITU-T standard X.509 [6,7]. All approaches have in common that a particular user or node publishes its public key on a key server of some sort from which it can be downloaded by other peers. After that, encrypted and signed messages can be exchanged. This, of course, requires that each participating node has to trust the key server. If a key pair ever gets lost, it can be revoked by including it in the so-called Certificate Revocation List (CRL), where all invalid keys and certificates are kept [7].

In [8] Ellison *et al.* argue that today's PKI based on *certificate authorities* (CA) imposes ten major risks. They describe, for example, the problematic trust background of the self-proclaimed authorities and discuss the weakest link issue of the CA structure. Furthermore they raise the question how the certificate holder identifies himself against the CA. They state that several procedures do exist and that there are no consistencies over all CAs.

2.2.2. Cryptographic Namespaces

Cryptographic namespaces take a different approach for key distribution. Instead of requiring a PKI, the public key is incorporated into the address of an end node. The Host Identity Protocol [9], for example, hashes the public key to a 128 bit value and uses this address as the identifier in a locator/identifier-split architecture. Other approaches like Accountable Internet Protocol [10] and Cryptographically Generated Addresses [11] incorporate a hash of the public key into an IPv6 address. In that way, a peer is able to instantly verify that the end node it is communicating with is in possession of the corresponding private key. There is no need to query a third party for the public key, hence no requirement for an additional trusted entity within the network.

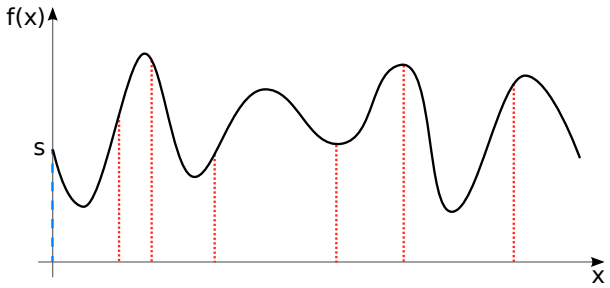
At a first glance, the cryptographic namespace seems to be a very promising idea, however, it has some major downsides. First of all, it adds inflexibility to key and address management. Whenever one of the items has to change (e.g., extended key length required), the other item has to change as well. Another problem is the missing link between the public key and a legal person. The cryptographic namespace only ensures that the end node with a certain address is also holding the valid private key for it. In practice, however, it is seldom relevant which end node we are talking to. For example, we want to talk to a specific person or our bank and not to a specific box of hardware down in a cellar. Another crucial point is the lack of key revocation. In case a private key gets lost or disclosed, the end node needs to generate a new $\langle public\ key, private\ key, address \rangle$-triple. Peers of that end node, however, might not immediately be aware of the address change and an attacker is able to use the old address combined with the private key to spoof his identity. Finally, in case the address space is quite full, it is easy for an attacker to generate a random key-address-triple which is already in use by another end node. In that way, an attacker is able to hide his identity.

2.2.3. Threshold Cryptography

A very promising field of cryptography within a flat hierarchy is Threshold Cryptography. This follows a (k, n)-threshold scheme. Here, n stands for the number of invoked parties and k for the minimum number which is necessary to decrypt a secret. To explain Threshold Decryption it is helpful to take a look to the principles of Shamir's Secret Sharing [12]. Shamir requires one dealer, which is trusted by all parties, as he knows every part of the key and he has to distribute them to the parties. Shamir's scheme relies on the construction of a polynomial following the pattern $f(x) = s + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1}$ where s represents the secret. Hence, all a_i values can be chosen unrestrictedly by the dealer. Afterwards, the dealer can generate n arbitrary value tuples $(x, f(x))$, where x can be disclosed, but as $f(x)$ is a share of the key, it must be kept as a secret. At least k value tuples are required to calculate a polynomial and consequently also the secret s.

In Figure 1, a polynomial is shown where the y-value is marked blue at $x = 0$. This is equivalent to the secret s in Shamir's principle. The red marked points are basic values which are distributed to every single party by the dealer. The x-value of the basic values are public, the corresponding $f(x)$-values are subject to non-disclosure and are known only by the respective party and by the dealer. If there are enough $(x, f(x))$-pairs available, it is possible to solve the polynomial and calculate the secret s accordingly.

Figure 1. Simplified geometric representation of Shamir's Secret Sharing.



2.3. Secure Address Mapping in Today's Internet

Today, the common method to determine endpoint nodes on the Internet is IPv4 which assigns a 32 bit address to an IP-enabled device. As these addresses are difficult to memorize for humans, the Domain Name System (DNS) was established. With that, domain names can be assigned to IP addresses, which makes them easy to remember and to type them into the web browser. When a request to one domain name is initiated, a name resolution via DNS is triggered and the IP address of the corresponding IP address holder, e.g., web server, is resolved. However, DNS has been exposed to various attacks like DNS-spoofing, cache poisoning or denial of service attacks against name server. Some approaches address these issues in order to provide secure routing and address respectively name resolution. One of the most popular ones has become Domain Name System Security Extensions (DNSSEC) [13–15] within the last years. DNSSEC is supposed to provide origin authentication of DNS data, data integrity, and authenticated denial of existence for DNS by means of a PKI. A DNS resolver, *i.e.*, client, can therefore verify the authenticity of the DNS data requested. However, DNSSEC does not provide confidentiality of the DNS data, which means that they are transmitted without encryption. Confidentiality for DNS queries is provided by DNSCurve [16], which encrypts the DNS data transmitted. Furthermore, DNSSEC leaves some security issues unanswered, e.g., it does not prevent distributed denial of service (DDOS) attacks on name server. In fact, it is even more prone to DDOS due to the higher resource consumption required to resolve one request. During the development process of DNSSEC, VeriSign, an operator of two DNS root servers, proposed [17] to share the master key among all root server operators using threshold cryptography as described in [18]. This would foster an equal voice of all operators. Currently, VeriSign is operating a prototype to evaluate Threshold Cryptography to activate private keys by multiple entities [19]. Anyhow, as 10 out of all 13 root server operators [20] are legally registered in the US, a fair balance cannot be achieved on a global scale.

3. Our Next Generation Internet Architecture

In this section, a brief overview of the HiiMap architecture is given. We only discuss the aspects which are relevant for the security extension introduced and evaluated in this article. For a more detailed introduction, please refer to [1].

In today's Internet architecture, the IP address has two semantic meanings. First, it serves as the address stating *who* I want to communicate with and, second, *where* I can find that person or node.

Contrary to the IP addressing scheme, any protocol following the locator/identifier-split principle, provides two separate addresses—namely an identifier and a locator.

One advantage of the locator/identifier-split idea is the built-in support for mobility. Applications communicate based on identifiers which are not changing even if a node is roaming. By changing its connection point towards the network only the locator reflects the new topological location of the node. An intelligent entity within the network is responsible to map the identifier to the current valid locator. The mapping system stores the current valid $\langle identifier, locator \rangle$-tuple. Whenever a node roams and its locator changes, it needs to inform the mapping system of the updated locator. Storing multiple locators for a single identifier is also possible. In addition to the locator, a timestamp is stored with each mapping entry. This timestamp indicates the last update of the locator.

The HiiMap architecture is based on the locator/identifier-split and provides a two-tier hierarchical mapping system. The 128 bit identifier in HiiMap is a flat randomized address and once assigned it never changes other than the user requests a new identifier for his node. In HiiMap, the identifier can also be used to address persons or content, but for simplification we only consider end nodes from here on. The mapping system in HiiMap is split into administrative domains, so called regions. Each region holds the mapping for the nodes it is responsible for. Even if a node roams (temporarily residing within another region) the mapping for that node is still resolved at its responsible region. To identify the responsible region of a node, an 8 bit regional prefix is provided for each identifier. Contrary to the identifier, this prefix is allowed to change whenever a node relocates to another region—this means, switching to another provider which is associated with another region for example. Figure 2 illustrates the identifier—regional prefix set. The regional prefix for any identifier can be learned in various ways. It can be provided along with the identifier from either a link on a web page or a domain name system entry. It also can be queried at the global authority which holds the identifier to responsible region mapping for each id. To lessen the burden of the global authority, the regional prefix can be cached as it is expected to change quite rarely.

Figure 2. Identifier with regional prefix (RP).

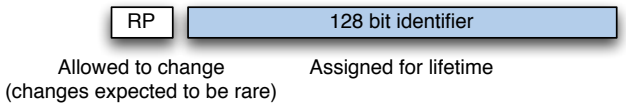
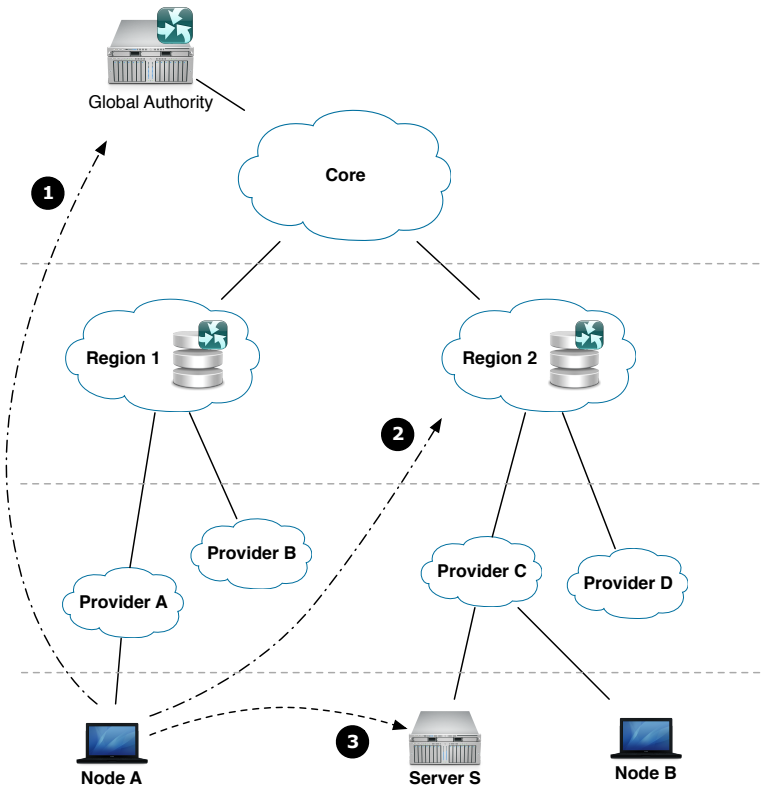


Figure 3 shows the topology of the HiiMap mapping system and an example of the lookup process. In this case, node A wants to establish a connection to a server S. In case node A did not learn the identifier from a domain name system or link—and, therefore, did not learn the regional prefix—it first needs to contact the global authority (1). The global authority returns the regional prefix corresponding to the server's identifier. In a next step—and all succeeding connections—user A directly contacts the region of the server S to resolve the mapping for the locator (2). The mapping system of the region returns the current valid locator and node A is able to send packets addressed to the locator of server S (3). In case node A wants to communicate with another mobile node, the connection setup process remains the same. If the second node, however, roams during the connection, it informs node A directly about its new locator.

Figure 3. Example HiiMap topology.



As mentioned earlier, we partition the mapping system into multiple regions. We base the partitioning on countries, whereby each country forms its own region. This concept has two important advantages. Firstly, most countries show a relatively stable state. It rarely occurs that a country institutes or vanishes. This means there is a very seldom need to adjust the regional prefixes. With a prefix length of 8 bits, there are still enough spare prefix addresses for new countries. The second benefit of a partitioning based on countries is the common legal system. Each country has its own laws and ways of law enforcement. Therefore, a region based on more than one country has to deal with different political and legal systems. In this way it is easier to build trust relationships between providers and handle infringements of contracts by the local law enforcement. The regional mapping authorities—operated under the same law as the participating providers—can easily be subject to contracts and business relations. Today's DNS registries, responsible for the Top Level Domains of countries (e.g., DENiC for .de), can be seen as a role model for the regional authorities. The difference, however, is the direct participation of providers in terms of financing the authorities. We also advice that a government authority (or another non-profit organization) retains control over the regional authorities.

Further, we put a focus on reliability and scalability. To this end, we suggest to use distributed hash tables (DHT) within the regions as a database solution for the mapping system. They are able to scale [21] with the demands and require only low maintenance efforts due to self-organization. In case one node within the DHT fails, the other nodes automatically take over. To keep the lookup delay to a bare minimum, we suggest to use so called one-hop DHTs like the D1HT protocol [22].

4. Security Approach

In this section, we describe the integration of the public key infrastructure into the HiiMap mapping system.

4.1. PKI and the Mapping

In today's Internet, the PKI is separated from all network services. This means that additional resources for the PKI must be provided despite all the network elements already in place for other functionalities (e.g., DNS server). Contrary to this, we propose to integrate the PKI into the mapping system for the HiiMap architecture. This has the benefit that resources can be shared between functionalities and maintenance can be kept significantly lower compared to operating separate services.

Each mapping entry consists of the identifier as the primary key and a set of locators by which the node currently can be reached (see Section 3). Further, a timestamp of the last update and a flag, indicating whether the location update was cryptographically signed by the node or not, is stored. Normally any location update must be signed to prevent identifier hijacking. Some devices, however, might not have enough computational power to handle long asymmetric cryptographic operations. To integrate those devices, the owner can request at the regional authority to send unsigned location updates. An unsigned location update is signaled by a set flag and peers can decide whether they want to start a communication with such a node or not.

To combine the PKI with the mapping system, only the public key of each node must be additionally stored for each mapping entry. This means that no additional protocol or infrastructure has to be provided for querying and storing the public keys. Because the public key is a very static value and not expected to change frequently, the additional burden for the mapping system is limited and the public key databases can be optimized for frequent read requests—contrary to frequent read and write requests for the locators.

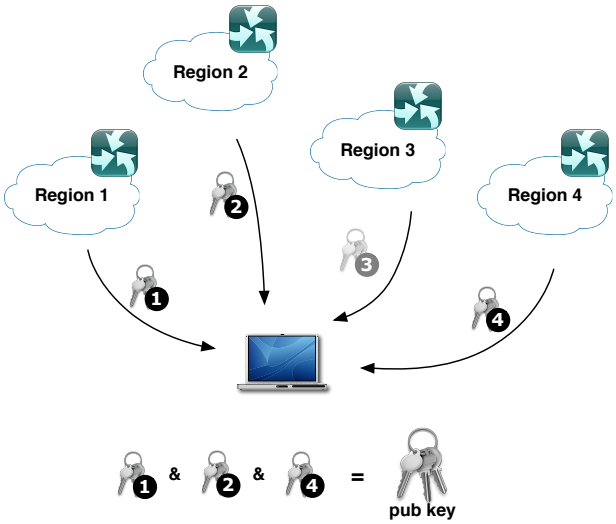
In comparison to today's PKI, it is also not necessary to provide additional lists for key revocation. This is implicitly realized by the loose identifier to public key binding in the mapping system. Whenever a public key for a certain identifier changes, the old public key implicitly becomes invalid.

4.2. Trusting the Mapping

By storing the public key at only one administrative domain (region) in the mapping system, however, the user heavily depends on the trustworthiness of that particular region. In case the mapping service provider collaborates with an attacker, it could send a wrong or manipulated public key to the client. Therefore, any security functionality based on the public/private key principal would be rendered useless. Even worse, the client considers the connection to be secure while in fact talking directly to the attacker.

To address this issue, we suggest to divide the public key into n pieces and distribute those to various independent regions. To calculate the public key only k out of n pieces are required, using the threshold cryptography, as explained in Section 2.2. Figure 4 illustrates an example with $n = 4$ and $k = 3$. The client is able to reconstruct the public key from a subset of the key pieces. This means that not all key pieces must be obtained. Untrustworthy or topology wise far away regions can be omitted and the client can choose which regions to query for the key pieces. In that way, the key is not stored at only a single region and at least k regions must be queried to reconstruct the key.

Figure 4. Reconstructing the shared public key.



In comparison to distributing full key copies to several regions, this approach has an important benefit. In case a region stores the complete key and cooperates with an attacker, it could swap the key with another one. This is obviously a problem in case the requester relies only on a single region. Even if he queries multiple regions, the result will be a set of non–matching keys. This means that he needs to base his trust on the majority principle by selecting one of the keys. Some requesters might choose to cancel the connection setup because of the irregularities of the obtained key set.

In a non-shared key system, malicious regions could generate a new public key that corresponds to their private key and use this combination to interfere the whole system. Contrary, by only holding a piece of the key, a malicious region can only toggle some bits in the corresponding part of the key. This would be detected during a request as the data queried would be invalid. Once the requester has obtained all key pieces, he only needs a subset of all pieces to reconstruct the original key. Even if he received a modified key piece, he can use various combinations of key pieces to reconstruct the public key. In case he accidentally included the modified piece, the reconstruction results in a non–valid public key format. In that way, the requester knows that at least one of the pieces was modified and he needs another variation of key pieces.

4.3. Determining the Storage Location

Having multiple pieces of the public key distributed over several regions in the mapping system, one question remains: In which way does the client learn about the storage location of the key pieces?

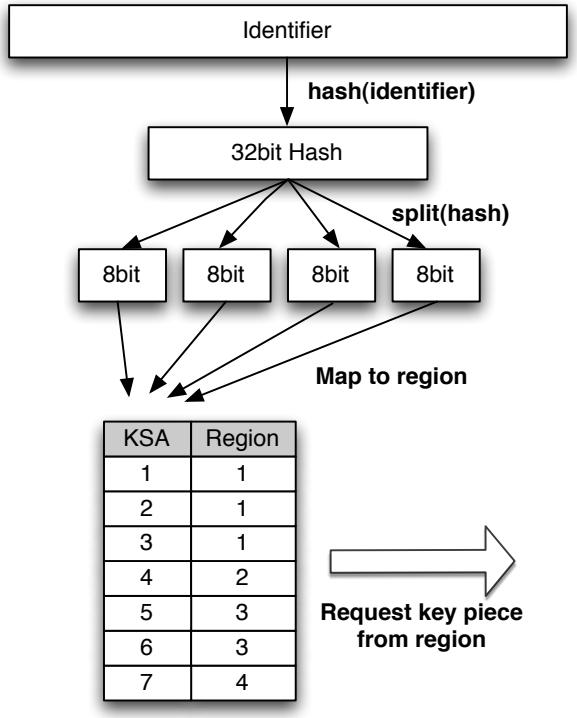
Storing the list of the regions at the responsible region imposes another risk. In case the responsible region is the attacker, it can simply manipulate this list and distribute the malicious key pieces to collaborating regions. Therefore, the client must learn the information about the storage locations in another way.

For the following proposal, we assume that less than 256 regions exist and for the sake of simplicity we assume $n = 4$ in this example. Note that any other number up to 256 will work for n. After having received the locator and key piece from the responsible region, the client hashes the identifier to an

32 bit value. The hereby used hash function must satisfy the requirement that even small changes to the input must result in a complete different hash value. This is to ensure that the key pieces of subsequent identifiers are stored at different regions. We suggest to use a cryptographic hash function like SHA-1 for that matter.

In a next step, the 32 bit outcome of the hash function is split into quarters (8 bit each). Each 8 bit value represents the storage location of one of the key pieces. We will call it key storage address space (KSA) from now on. Since there are viewer then 256 countries, the 8 bit address space for the regions is not completely full and a mapping directive is required. This mapping directive can be downloaded from the global authority. It is sufficient to do this very seldom, as the directive is expected to change very rarely. For each value in the KSA, the mapping directive specifies a region, where the key piece is stored. This means that a single region can be responsible for several KSA values. In that way, the load can be fairly distributed over all regions depending on their size. Figure 5 illustrates the process.

Figure 5. Key Piece Storage Location.



Should the hashing and mapping to regions result in two key pieces being stored at the same region, the first piece is stored at this region and the second piece at the region with the next higher region number.

4.4. Detecting Malicious Regions

With the security approach presented, clients have to query multiple key holders in order to resolve one routing request. Depending on the setup of the Threshold Cryptography algorithm, the minimum number of regions that have to be queried k, varies. As soon as the client has at least k resolves conducted, it can decrypt the secret s. However, s can only be retrieved if all resolves are correct. If one

of them has been modified, e.g., to disturb the whole mapping system, the secret will be corrupted. In this case, a client can alter the regions which are taken into account for one query. In a further step, the client could also rotate these regions to determine which region has provided an invalid response. This can then be reported to a central entity, e.g., the global authority (GA). We propose that random probes are also conducted by the GA periodically to monitor all regions and the operation of the overall system. This again should be indicated to clients in order to give a hint on each region's credibility.

4.5. Initial Key Upload

Whenever a new node connects to the network for the first time, its public key must be stored in the mapping system in a secure way. For security and usability reasons, we propose to use cryptographic smart cards to store the public key, private key and identifier of any node at the users side [23]. The smart card is issued by the global authority—or one of its delegates—and initially generates a public–private key pair on chip. One of the major advantages of smart cards is the property that the private key never leaves the smart card. Before shipping the smart card to the user, the public key is derived from it and stored in the mapping system. In this way, the GA is never in possession of the private key and the public one is copied to the mapping server within a secure environment. For further details of this initial bootstrap see [23].

4.6. Attack Scenarios

There are several attack scenarios that can be derived from the description given above. All of them intend to either reroute a connection establishment, to disturb communication at all or to subvert the overall trust model. The most obvious approach would be to inject a counterfeited key in one of the regions. This, of course, would be feasible if one of the regions itself has a vicious intention to disturb the whole system. In that case, a client would get a faked public key from that region, but as it needs multiple regions to get the endpoint reference (see Section 4.2), the faked key would occur to the user. In fact, keeping in mind the threshold cryptography principle depicted in Figure 1, that counterfeit key would end up in modified $(x, f(x))$ -tuples which makes it impossible to determine the secret s, respectively the correct endpoint reference.

A more sophisticated attack scenario would be a cooperation of several malicious regions which all deploy vicious public keys. This issue can be addressed by relying on the (k, n)-threshold scheme. That means that the number of regions k needed to determine s has to be sufficiently high. Apparently, this is a trade-off between complexity, delay and security. However, we do not see the motivation of several malicious regions cooperating in order to jeopardize the whole architecture. These attacks correspond to DNS Poisoning in today's Internet.

A common way to disturb communication have become Distributed Denial of Service (DDoS) attacks recently. Here, a significant amount of clients send requests to one target. The latter one then collapses from the sheer amount of (unreasonable) requests. Of course, this attack method is also conceivable in the presented system. However, the vital information to keep the system up are shared between several nodes within one region. Furthermore, multiple regions hold the same information. Therefore, the effort to jam the system seems excessively high.

The system relies on certificates and therefore on unique identities which ensure a trustworthy relationship between nodes. Hence, a possible attack could also address the certificate with the private key of one client. An attacker could then spoof his identity. To secure the certificate, the system relies on cryptographic smart cards as described in Section 4.5. If an attacker gets possession of the smart card, the original owner can report the loss to his corresponding regional administration which then deletes the public key pieces from their mapping entries and replaces them with the new key pieces. The smart card, respectively the certificate are then invalid and the user is issued a new smart card with a new certificate on it. Consequently, the trustworthiness of identities can be ensured by the described approach.

5. Qualitative Evaluation

In order to show how our approach performs, *i.e.*, if it can be regarded as secure and if it fulfills the performance requirements of a future internet, we present a qualitative metric of evaluation, which matches our system against the requirements from the guidelines mentioned in [3].

We consider an AAA framework to stand out as a suitable and convenient solution, due to three main reasons. First, Authentication, Authorization and Accounting principles infer an interaction of mechanisms to cover the total number of security dimensions in order to provide access control, non-reputation and privacy support. That means, countermeasures against the enumerated security threats in Section 2.1 are considered as well. Second, an AAA framework does not establish specific security protocols, but outlines a collection of conditions to be followed in order to progressively customize the security level of one particular network. The integration of multiple technologies brings some flexibility and continuous scalability dynamic to the security infrastructure. Last but not least, AAA infrastructures are subject of evaluation by multiple standards from the IETF [3], which states the chance to review the efficiency of the adopted mechanisms for a future internet.

In the following we evaluate our solution against the requirements of the AAA framework. We take 4 levels of coverage as a metric to describe how our system performs: The qualification levels are “T” for total coverage, “P” for partial coverage, “F” for failed coverage and “OS” for out of scope of this work.

5.1. General Requirements

Table 1 lists all general requirements from the IETF AAA recommendation. The first both issues, scalability and fail-over, are covered by our DHT-based approach, due to the nature of the employed DHT protocol [21]. A mutual authentication is done between the client and the mapping system based on the used asymmetric cryptography principle. Transmission level security is generally supported if it is required by one party, however, it has to be noted that our system resides on layers below the transmission layer. Therefore, it is transparent to an eventual Transmission Layer Security. The data object confidentiality is only partially covered because we secure the transmission of data, but the mapping system is open to everyone. This is because any authenticated client is able to initiate lookup requests. As we use signatures to sign mapping responses within our architecture, integrity of the data objects can be regarded as fully covered. The transport of certificates is not required because the system is not relying on certificates at all. We, therefore, regard this aspect as out of scope. As all security-related data is transmitted in a trustworthy manner—encrypted and signed—the system fulfills

all AAA reliability requirements. IPv4 and IPv6 is completely transparent for our architecture, in fact, our approach is not limited to IP routing respectively addressing. We therefore support both IP versions as well as possible future network layer protocols. Correspondingly, also proxies are higher layer issues and are therefore fully supported. The requests and replies from and to the mapping system are not supposed to be altered on their way. This is ensured by the integrity checks. We, therefore, consider the audability aspect to be out of scope. Our system does not require a shared secret in terms of a common token—note the difference between shared credentials and Shamir's shared secret. We solely rely on asymmetric cryptography. Similarly to the support of future protocols (network layer and higher), service specific attributes can seamlessly be included into the communication pattern as this does not affect our proposed architecture.

Table 1. AAA general requirements qualifications.

Type	Requirement	Rank
General	Scalability	T
	Fail-over	T
	Mutual Authentication	T
	Transmission Level Security	T*
	Data Object Confidentiality	P
	Data Object Integrity	T
	Certificate Transport	OS
	Reliable AAA Transport Mechanism	T
	Run over IPv4	T**
	Run over IPv6	T**
	Support Proxy and Routing Brokers	T
	Auditability	OS
	Shared Secret not Required	T
	Ability to Carry Service Specific Attributes	T

*If required **Not limited to

5.2. Authentication Requirements

As can be seen in Table 2, the HiiMap architecture with its identifier/locator–split principal supports NAI right away. An authentication is not required to be authorized to resolve endpoints from the mapping system. We therefore regard this issue as fully covered. Reauthentication is out of scope for our concept, as connections to the mapping system only consist of a single request and response message scheme. Higher layers, however, may implement this mechanism. All other requirements concerning the authentication are transparently supported as they reside on higher layers.

Table 2. AAA authentication requirements qualifications.

Type	Requirement	Qualification
Authentication	Network Access Identifier (NAI) Support	T
	Authorization without Authentication	T
	Reauthentication on Demand	OS
	Challenge-Handshake Authentication Protocol (CHAP) Support	T*
	Extensible Authentication Protocol (EAP) Support	T*
	Password Authentication Protocol (PAP)/Clear-text passwords	T*

*If required

5.3. Authorization Requirements

A summary of all authorization requirements is given in Table 3. Support for static and dynamic IP address assignment is possible if required. This as well as the support for Remote Authentication Dial-In User Service (RADIUS) is covered correspondingly to the IPv4/v6 support outlined in the general requirements Section 5.1. The system covers the capability to reject single participants, e.g., if they behave in a suspicious way, due to the integration of signatures. Hence, also access rules could be applied. As this is not meaningful in the given scenario (everybody is allowed to retrieve an address mapping), we regard this criteria as out of scope. Layer 2 Tunneling cannot be prohibited. However, the mapping system cannot be bypassed and, therefore, Layer 2 Tunneling has no benefit to an attacker. Reauthorization on Demand is not required here and therefore neglected. Queries from clients do not have a significant influence on the system's state, nor do they change them (as far as it is not a malicious DDoS attack). Furthermore, the distributed self-organized mapping mechanism guarantees a steady operable state. Unsolicited Disconnects, *i.e.*, a disconnect initiated by the mapping system, is regarded as failed here. Our approach handles requests on an event–basis. Therefore, there is no way for a server to disconnect an ongoing transmission.

Table 3. AAA authorization requirements qualifications.

Type	Requirement	Qualification
Authorization	Static and Dynamic IP Address Assignment	T*
	RADIUS Gateway Capability	T*
	Reject Capability	T
	Preclude Layer 2 Tunneling	OS
	Reauthorization on Demand	OS
	Support for Access Rules and Filters	OS
	State Reconciliation	T
	Unsolicited Disconnect	F

*If required

5.4. Accounting Requirements

Accounting, in general, is not within the scope of this article as an evidence of who has requested which address at a given time is not practical nor applicable to a global mapping system. The system puts a focus on integrity of the requests/responses in terms of integrity and validity rather than on monitoring each of them. However, basic requirements within the accounting section (Table 4) can be regarded as fulfilled, which is a guaranteed delivery and accounting timestamps (for location updates as described in Section 3). Dynamic accounting, which is regarded as out of scope, denotes multiple recordings of timestamps for a single session respectively transmission. Our system does not integrate sessioning, therefore, this has to be included in higher layer services. Other requirements as real-time accounting, an obligatory compact encoding, the possibility to extend the accounting records as well as a batch accounting are not incorporated here.

Table 4. AAA accounting requirements qualifications.

Type	Requirement	Qualification
Accounting	Real-Time Accounting	OS
	Mandatory Compact Encoding	OS
	Accounting Record Extensibility	OS
	Batch Accounting	OS
	Guaranteed Delivery	T*
	Accounting Timestamps	T
	Dynamic Accounting	OS

*If required

Evaluating the overall results including general requirements, authentication, authorization, and accounting aspects, our system fulfills 92% of the requirements, partially covers 4% and misses 4%. These results are obtained neglecting out of scope aspects.

6. Conclusions

In this article we introduced and evaluated a security approach for the HiiMap Next Generation Internet architecture. The HiiMap architecture is a clean-slate approach which takes scalability, reliability, mobility and political trust issues into account. A fair and trustworthy architecture for the Next Generation Internet can therefore be achieved. Based on this foundation we presented how to integrate a PKI into the mapping system. By using Shamir's Threshold Cryptography principle, we accomplished to distribute the public keys over several administrative domains (regions). In that way, a requester is not dependent on the trustworthiness of a single region. Furthermore we showed how the global authority can verify the integrity of each domain.

Contrary to having security functionality spread over all OSI layers, we bundled it into the network layer. As a result a solid and consistent security functionality is provided to all upper layers. The qualitative evaluation which has been performed according to [3] shows that we meet almost all recommendations which are not out of scope for this security approach.

Acknowledgement

This work has been performed within the G-Lab project [24] and was funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (Project ID 01BK0807). It also has been supported, in part, by the BMBF funded research project SEIS (Security in Embedded IP-based Systems) [25]. The authors alone are responsible for the content of the paper.

References

1. Hanka, O.; Kunzmann, G.; Spleiß, C.; Eberspächer, J.; Bauer, A. HiiMap: Hierarchical Internet Mapping Architecture. In *Proceedings of First International Conference on Future Information Networks*, Beijing, China, 14–17 October 2009.
2. Meyer, D.; Zhang, L.; Fall, K. Report from the IAB Workshop on Routing and Addressing. *IETF* **2007**, RFC 4984.
3. Calhoun, P.; Hiller, T.; McCann, P. Criteria for Evaluating AAA protocols for Network Access. *IETF* **2000**, RFC 3127.
4. Union, I.T. Telecommunication Standardization Sector (ITU-T). Available online: http://www.itu.int/ITU-T/ (accessed on 29 January 2011).
5. ITU-T. ITU-T Recommendation X.805—Security architecture for systems providing end-to-end communications. 2003. Available online: http://www.itu.int/itudoc/itu-t/aap/sg17aap/history/x805/index.html/ (accessed on 29 January 2011).
6. Housley, R.; Ford, W.; Polk, W.; Solo, D. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. *IETF* **1999**, RFC 2459.
7. Housley, R.; Polk, W.; Ford, W.; Solo, D. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. *IETF* **2002**, RFC 3280.
8. Ellsion, C.; Schneider, B. Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. *Comput. Secur. J.* **2000**, *16*, 1–7.
9. Moskowitz, R.; Nikander, P. Host Identity Protocol. *IETF* **2006**, RFC 4423.
10. Andersen, D.G.; Balakrishnan, H.; Feamster, N.; Koponen, T.; Moon, D.; Shenker, S. Accountable internet protocol (AIP). In *Proceedings of SIGCOMM '08: The ACM SIGCOMM 2008 Conference on Data Communication*, Seattle, WA, USA, 17–22 August 2008.
11. Aura, T. Cryptographically Generated Addresses (CGA). In *Information Security, Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 29–43.
12. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613.
13. Arends, R.; Austein, R.; Larson, M.; Massey, D.; Rose, S. DNS security introduction and requirements. *IETF* **2005**, RFC 4033.
14. Arends, R.; Austein, R.; Larson, M.; Massey, D.; Rose, S. Resource records for the DNS security extensions. *IETF* **2005**, RFC 4034.
15. Arends, R.; Austein, R.; Larson, M.; Massey, D.; Rose, S. Protocol modifications for the DNS security extensions. *IETF* **2005**, RFC 4035.
16. Bernstein, D. DNSCurve. Available online: http://www.dnscurve.org/ (accessed on 29 January 2011).

17. VeriSign. VeriSign Root Zone Signing Proposal. Available online: http://www.ntia.doc.gov/DNS/VeriSignDNSSECProposal.pdf/ (accessed on 29 January 2011).
18. Cachin, C.; Samar, A. Secure Distributed DNS. In *Proceedings of DSN '04: The 2004 International Conference on Dependable Systems and Networks*, Florence, Italy, 28 June–1 July 2004.
19. Okubo, T.; Ljunggren, F.; Lamb, R.; Schlyter, J. DNSSEC Practice Statement for the Root Zone ZSK Operator. Available online: https://www.verisign.com/repository/dnssec-practice-statement-root-zone-zsk-operator.pdf/ (accessed on 29 January 2011).
20. Root Server Technical Operations Association. Available online: http://www.root-servers.org/ (accessed on 29 January 2011).
21. Kong, J.S.; Bridgewater, J.S.A.; Roychowdhury, V.P. A General Framework for Scalability and Performance Analysis of DHT Routing Systems. In *Proceedings of International Conference on Dependable Systems and Networks*, Philadelphia, PA, USA, 25–28 June 2006.
22. Monnerat, L.R.; Amorim, C.L. D1HT: A Distributed One Hop Hash Table. In *Proceedings of 20th IEEE International Parallel and Distributed Processing Symposium*, Rhodes Island, Greece, 25–29 April 2006.
23. Fritz, W.; Hanka, O. Smart Card Based Security in Locator/Identifier-Split Architectures. In *Proceedings of International Conference on Networks*, Les Menuires, France, 11–16 April 2010.
24. G-Lab. Available online: http://www.german-lab.de/ (accessed on 29 January 2011).
25. EENOVA. SEIS (Security in Embedded IP-based Systems). Available online: http://www.strategiekreis-elektromobilitaet.de/projekte/seis/ (accessed on 29 January 2011).