Decentralised Binding of Self-Certifying Names to Real-World Identities for Assessment of Third-Party Messages in Fragmented Mobile Networks

Jan Seedorf, Dirk Kutscher, Fabian Schneider
seedorf/kutscher/schneider@neclab.eu; NEC Laboratories Europe, Heidelberg, Germany,

***Abstract*—Self-certifying names provide the property that any entity in a distributed system can verify the binding between a corresponding public key and the self-certifying name without relying on a trusted third party. However, self-certifying names lack a binding with a corresponding real-world identity. In this paper, we present a concrete mechanism for using a Web-of-Trust in conjunction with self-certifying names to provide this binding. We consider a decentralised scenario: fragmented (mobile) networks, where connectivity to centralized authentication entities and Web-of-Trust key-servers is not available. Our approach enables a particular functionality in this scenario: The assessment of messages from previously unknown third parties. To the best of our knowledge, there is no prior art for combining a Web-of-Trust approach with self-certifying names to enable such transitive third-party data origin authentication in decentralised networks. Our analytical evaluation shows that — depending on the overall size of the Web-of-Trust and the average friend-degree among its users — it is feasible to apply our approach fully decentralised at end user devices, or at least highly decentralised at access network nodes.**

I. Introduction

Self-certifying names provide the useful property that any entity in a distributed system can verify the binding between a corresponding public key and the self-certifying name without relying on a trusted third party [1]. This feature enables (e.g. in content-oriented architectures such as ICN) an entity to digitally sign data associated with a self-certifying name, and any receiving entity can verify the signature without relying on a trusted third party or a Public Key Infrastructure (PKI). Self-certifying names thus provide a decentralized form of data origin authentication. However, as noted in [2] and elsewhere, self-certifying names lack a binding with a corresponding real-world identity (RWI): the concept enables to verify that whoever signed some data was in possession of the private key associated with the self-certifying name, but it does not provide any means to verify what real-world identity corresponds to the public key, i.e. who actually signed the data [2].

Options to provide this binding between a public key and an RWI include a PKI, or a Web-of-Trust (WoT) [2]. However, there have not been concrete proposals for a WoT-based approach for binding a public key (or a self-certifying name) and an RWI in content-oriented architectures. Moreover, the number of trust relationships in a WoT scale exponentially with the number of WoT users. Thus, any solution would need to address the scalability problem for processing/analysing the trust relationships in a large Web-of-Trust.

We propose a concrete mechanism for using a WoT in conjunction with self-certifying names in a specific scenario: fragmented (mobile) networks, where connectivity to centralized entities and authentication servers is not available. Further, our approach targets to enable a particular functionality in this scenario: The assessment of messages from previously unknown third parties. To the best of our knowledge, there is no prior art for combining a Web-of-Trust approach with self-certifying names to enable such transitive third-party data origin authentication in decentralised networks. Furthermore, we propose an intermediate level of decentralization of the WoT database, thereby addressing the scalability problem of WoT files while at the same time enabling the fully decentralized use of the WoT database by end-devices in disconnected, fragmented networks.

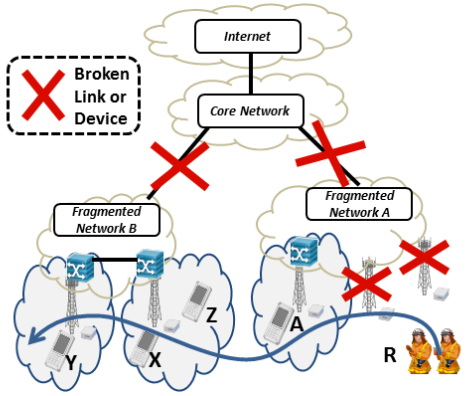

Figure 1. Mobile Network after Disaster with Fragmented Networks

A. Scenario and Example Use Case

To understanding the desired functionality of our proposed scheme, consider a natural or human-generated, large-scale disaster scenario like the enormous earthquake that hit Northeastern Japan in March 2011 [3]. In such a scenario, the (formerly connected) network is likely to be fragmented into several islands, due to failed links etc. One can assume that users can move among several of the fragmented islands over time (e.g. a user walking towards help/rescue with his mobile phone trying to connect to functional base stations along the way), and connect each time to any functional network equipment in each 'visited' fragmented network.

Given such a setting, decentralised authentication is challenging: In mobile networks, users are authenticated via central entities. In order to communicate in fragmented or disconnected parts of a mobile network, the challenge of decentralising such user authentication arises. Independently of the network being fixed or mobile, data origin authentication of content retrieved from the network is challenging when being 'offline' (e.g. disconnected from servers of a security infrastructure such as a PKI).

For instance, the following functionality is desirable for

978-1-4799-3088-3/14/$31.00 ©2014 IEEE

users, yet challenging to provide with state of the art authentication mechanisms: A user X has a personal relationship with user Y(e.g. they are married). After the disaster, X and Y are in different fragmented networks. X wants to let Y know that he is ok. However, X's mobile phone is broken. But X meets stranger Z, who let's X type a short text message in Z's mobile phone. Assuming that this message can somehow be delivered across fragmented networks via content-oriented, DTN-like[1] mechanisms, Y still needs to be able to verify that the message sent by Z is trustworthy. We propose a concrete scheme that enables the decentralised assessment of these kinds of *on-behalf-of* messages, even in settings where users cannot connect to any kind of centralized authentication server (or WoT keyserver) anymore.

Figure 1 shows the envisioned scenario and use case. Connectivity to the backbone or the Internet is broken, but certain parts of a mobile network infrastructure, e.g. base stations, are functional, forming small fragmented sub-networks. User Y is in a different fragment than user X. Rescue teams (R) (or user Z) may move across different network fragments, enabling to transport messages from one disconnected sub-network to another one in a DTN-like routing fashion.

B. Assumptions and Terminology

The proposed scheme assumes the following infrastructure and mechanisms:

- One or more Web-of-Trust keyserver(s) are present that each frequently publish a 'WoT file' that contains in a compressed, machine-readable format the verified certificate graph for the whole WoT[2]. Each public/private key pair in the system has a unique key-ID which is the cryptographic hash of the public key. Users can upload certificates which have been signed by other users to these servers. The semantic of a signature is that the signer considers the RWI in a signature as trustworthy. This means that users can express how trustworthy they regard other users to be by signing the certificates of these other users in the WoT[3]. Before the disaster, users promptly upload new certificates to the keyserver each time when having signed an RWI/self-certifying-name binding).
- All access network nodes of a certain type located close to end-users and not in the core of the network (e.g. base station in mobile networks or all nodes of type 'Broadband Remote Access Server' in fixed networks) frequently (e.g. once a day) download the WoT-file published by a WoT-keyserver. Note that such a WoT-file contains essentially a list of RWI-to-public-key bindings, where a) these bindings can be assumed to be correct (as the users trust the keyserver), and b) any RWI or public key can be searched for with low computational effort. We refer to the access network nodes that frequently download WoT files from central keyservers as 'D_{WoT}-nodes' (for decentralized WoT nodes).
- Some sort of a content-oriented architecture (e.g. using Information-Centric-Networking approaches or possibly a CDN on top of IP networks) is available even after the disaster, where users can publish and retrieve content by querying for (self-certifying) names under a given naming scheme (e.g. as outlined in [3]). This content-oriented architecture features some sort of publish/subscribe functionality, such that users can subscribe to a certain name and retrieve (either via frequent pull or via push) new content for this name. This means that the assumed architecture offers some sort of versioning, such that publishers can publish *new* content for a given name, and that users who subscribe to a given name can know which is the latest version of the content for the given name. DTN-like forwarding of messages across different disconnected sub-networks is possible. Message forwarding and the concrete routing of the content-oriented architecture are orthogonal to our proposed scheme: We solely consider the use of self-certifying names in a naming scheme and how to tie this information with Real-World-Identities in a decentralised scenario, where nodes are 'offline' from authentication servers or even WoT keyservers.

The D_{WoT}-nodes constitute a simplified version of a distributed WoT keyserver network, but with limited functionality: These nodes only possess a compressed version of the WoT certificate graph, but not the all the corresponding certificates themselves (as a regular keyserver would). The assumption is that this compressed WoT-file can be trusted, as it has been produced and obtained by a trustworthy central WoT keyserver. The D_{WoT}-nodes can hence only offer limited functionality using the WoT-file: a) Given an RWI, answer with the WoT key-ID, b) Given a WoT key-ID, answer with an RWI, c) Given two WoT key-IDs, answer with the certificate chain[4] between these key-IDs. This limited functionality is enough to enable our proposed scheme. Using compressed certificate graphs as in a WoT-file enables a lightweight, decentralised usage of the inherent trust relationships in a social network. In particular, using the WoT-file at the D_{WoT}-node level serves as a compromise between decentralisation (and thus working in fragmented, disconnected networks) and scalability of certificate graphs (i.e. with growing WoT size, these graphs may become too large to be stored and processed by end user devices such as mobile phones; we investigate this issue in Section III).

For simplicity, we denote names in the self-certifying naming scheme generically as $[cp : scp]$, where ':' is the concatenation operator, cp is a *context* part, and scp is a *self-certifying* part. Note that this notation is very similar to the often used $[L : P]$ notation of names, where 'P' is a cryptographic hash of the principal's public key and 'L' is the label [2]. In our scheme the cp part provides some sort of context through pre-defined labels, and the scp part is a cryptographic hash of the principal's public key.

[1]Delay Tolerant Networks

[2]Note that such a file format exists and is in the range of $1.5MB$ for 40.000 PGP users [4]. However, with a larger number of users this file may grow much larger; we investigate this scalability issue in Section III.

[3]Note that the semantic of signatures would be different when a centralized authority would sign RWI-to-public-key bindings: the trust scheme would not be decentralized anymore in this case as not users would certify name-to-public-key bindings (and the WoT-keyserver merely verify such binding and publish the list), but a central authority would make decisions on trustworthiness of users. The semantic of the signed binding would therefore be that a central entity has checked a given policy, and not that users consider each other trustworthy with respect to a given behavior.

[4]A Certificate Chain between two key-IDs k_x, k_y in a Web-of-Trust is a chain of signed certificates between users in the Web-of-Trust such that k_x asserts k_1 asserts k_2 ... asserts k_y, where assertion refers to signing the binding of the name with the corresponding public key.

II. A Scheme for Assessment of On-Behalf-Of Messages by Untrusted Third Parties

A. Scheme in Detail

The proposed scheme works as follows (taking the notation and use case example outlined in the previous section):

1) User Y wants to retrieve information about X, so Y *frequently* tries to resolve two names that contain a context part, cp, and a self-certifying part, scp. cp is a reserved string; scp is the cryptographic hash of a public key (e.g. scp_X is the SHA-1 hash of the public key of user X). Note that X and Y have a direct relationship in the Web-of-Trust, so it can be assumed that Y has the self-certifying name under which X publishes information, scp_X (e.g. stored in her mobile phone). The names that Y tries to resolve are:
   a) $[cp_{imp} : scp_X]$: This is the default name for X to publish important information about himself. Since X does not have a working device, X does not publish under this name, and thus Y will not retrieve any result for the name resolution query.
   b) $[cp_{obotrans} : scp_X]$: This is the default, reserved transitive 'on-behalf-of' name for X. X is not able to sign messages as X does not have a working device. However, Z can publish data under $[cp_{obotrans} : scp_X]$. The reserved context $cp_{obotrans}$ signals to the content distribution architecture that this is a transitive 'on-behalf-of' publication of content, so the self-certifying property will not be checked (as Z cannot directly sign with X's private key that would correspond to scp_X). The authenticity of the content published under $[cp_{obotrans} : scp_X]$ can therefore also not be checked directly; this needs to happen transitively as outlined below.
2) Since Z does not know scp_X (and X would likely not know scp_X without his phone), Z sends a query to any functional $D_{WoT}-node$ in his currently visited fragmented network with RWI_X, i.e. the RWI of X (which presumably X can tell to Z). The D_{WoT}-node will return ID_X, the key-ID for X's public key in the WoT, using its latest (retrieved prior to the disaster) WoT-file.
3) Every time Z reaches a new fragmented network, he publishes as content for $[cp_{obotrans} : scp_X]$ his own non-transitive 'on-behalf-of' name $[cp_{obo} : scp_Z]$ along with his public key, pk_Z. The reserved context cp_{obo} signals that this is a non-transitive 'on-behalf-of' name.
4) Further, Z publishes under $[cp_{obo} : scp_Z : scp_X]$ as content the message M from X, but signed by Z.
5) Y will retrieve as result for querying for $[cp_{obotrans} : scp_X]$ the name $[cp_{obo} : scp_Z]$ and pk_Z. The reserved context cp_{obo} signals to Y that this is a non-transitive 'on-behalf-of' name for scp_Z.
6) Y now knows that some user Z published data on behalf of X. She can resolve the name $[cp_{obo} : scp_Z : scp_X]$ (which she formed by appending the self-certifying name of X to the obtained data item $[cp_{obo} : scp_Z]$ from the previous step) to obtain the message M from X, published and signed by Z. Y can verify the signature over M by Z (using pk_Z), so Y can be sure that M could only have been signed by Z. Further, as the hash of the public key is the key-ID in the WoT, Y can be sure that indeed the RWI of Z, RWI_Z, is part of the WoT.
7) Y can now query any functional D_{WoT}-node in her currently visited fragmented network for the certificate chain from its own key-ID, ID_Y, to the key ID of Z, ID_Z, in the WoT. Using a double-side Breadth Search First (dBFS) algorithm on the WoT-file, the D_{WoT}-node can determine this certificate chain[5].
8) Depending on properties of the certificate chain (e.g. its length), Y can decide whether to trust Z, and therefore whether to regard the original message M from X as authentic.

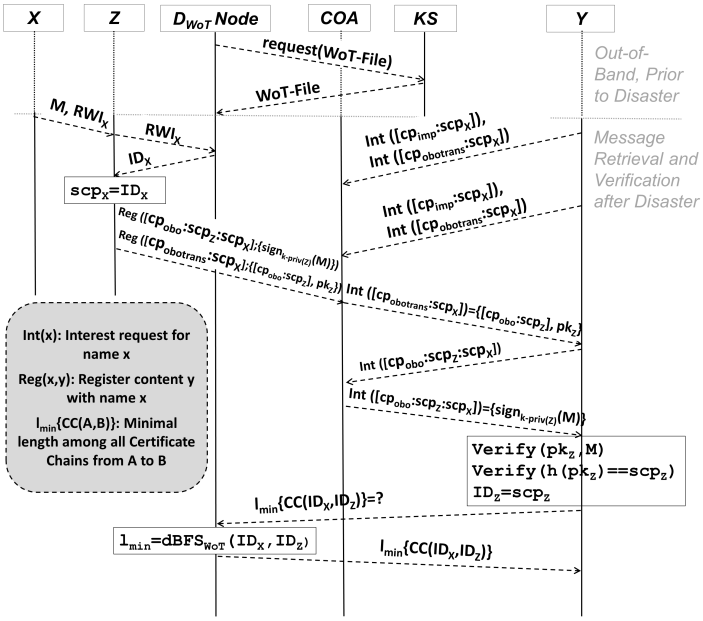

Figure 2. Detailed Message Flow of Proposed Scheme

An important fact in our scheme is that $h(pk_X) = ID_X = scp_X$, where $h()$ is a predefined system-wide hash-function. In other words, the WoT key-ID is equivalent to the self-certifying name part used in the naming scheme. This ties the self-certifying name with the ID of the corresponding public key in the WoT.

Figure 2 shows the message flow of our scheme among the participating entities, where COA is not a single node, but virtually represents a Content-Oriented Architecture, i.e. a given infrastructure for the publishing and retrieving of content, and KS a WoT keyserver. Note that the actual Content-Oriented Architecture (COA) is orthogonal to our scheme and only serves as a means for publishing and retrieving content for a given name. Note further that the users will never see nor need to know any self-certifying name or key-ID. All users have to know is their own RWI (e.g. an email address), from which the corresponding self-certifying name (which is equivalent to the WoT keyID in our scheme) can be derived — presumably automatically and not visible directly to users — via the WoT-file.

[5]Note that the execution of such a dBFS algorithm is computationally very feasible on today's hardware for common WoT files; we investigate the performance details of this algorithm further in Section III.

B. Decentralisation Options and Tradeoffs

To allow for scalability with respect to WoT-file size and certificate graph search performance, so far we have presented the scheme with access network nodes (such as base-stations) acting as D_{WoT}-nodes. However, it is also possible that end-user devices (e.g. smartphone, tablets) act as D_{WoT}-nodes themselves. In this case, end-user devices frequently (e.g. once a day) download the current WoT-file from a trusted WoT key-server, and when offline (i.e. after a disaster has struck and networks are fragmented) use this file to search for RWIs, key-IDs, and to compute certificate chains itself. We analytically study the scalability and performance of this option in III. Such a complete decentralisation of the scheme would allow for third-party message assessment while only having DTN-like connectivity among end-user devices instead of requiring some functional D_{WoT}-nodes as part of the fragmented network infrastructure.

C. Notable Properties

The presented scheme has the following notable properties. First, the trustworthiness of messages can be assessed in a decentralized fashion and does not rely on a PKI or access to any centralised entity. While this holds for any self-certifying name, the presented scheme extends this property for real-world identities and further for only transitively known identities in a social network (i.e. the WoT) by proposing a dedicated naming scheme. Second, using the concept of the downloaded WoT-file, any end user node in the system can potentially compute certificate chains and verify transitive trust chains, either itself (e.g. smartphones acting as D_{WoT} nodes themselves) or by using services provided by decentralized access network nodes (e.g. base stations acting as D_{WoT} nodes).

Moreover, a key idea is to use the hash of the public key as self-certifying component in the content-oriented architecture's naming scheme and at the same time as the key-ID in the WoT. This cryptographically ties self-certifying names (and the corresponding signed data items) with real-world identities in the WoT, and enables to use WoT in a decentralized fashion. Finally, as a kind of privacy-preserving feature, Z is only publishing as content $[cp_{obo} : scp_Z]$ under the specific name $[cp_{obotrans} : scp_X]$. It is thus not easy to infer for outsiders for what identities Z has performed 'on-behalf-of' publishings. In particular, there is no name which resolves to all the identities for which Z has performed 'on-behalf-of' publishing.

D. Potential Limitations

The presented scheme also has some limitations. First, the use of certificate chains for deciding on trustworthiness of real-world identities can be attacked. For instance, malicious nodes can try to trick users into signing their RWI-to-public-key binding in the WoT. In this sense, the scheme offers *better-than-nothing security* in situations where connectivity to an (otherwise potentially more secure) authentication infrastructure has been lost. Such attacks on certificate chains can be mitigated by applying more sophisticated schemes for deciding on the trustworthiness of certificate chains, e.g. using the number of independent paths between two key-IDs in the WoT, which would force an attacker to infiltrate multiple paths to be considered trustworthy.

Second, the WoT files used by nodes will always be slightly outdated. This is not a major drawback since the changes in a WoT certificate graph are not expected to be rapid in nature. Instead, a slightly outdated WoT file will contain useful information for the targeted scenarios. Similarly — as in any scheme based on asymmetric cryptography — key revocation may impose some challenges with the presented scheme: Users can revoke keys and certificates at any time with a keyserver, but once users are 'offline' from a keyserver and use WoT-files, such revocation is not easily possible with the proposed scheme. Again, this is not expected to be a problem since also revocation changes in a WoT certificate graph are not expected to be rapid in nature. It is noteworthy though, that key and signature revocation may happen more frequently with the proposed scheme than in a traditional PKI setting, as the signatures have an additional semantic of assessing the trustworthiness of identities in a social network which potentially may change more frequently than the need to revoke a key in a PKI scenario.

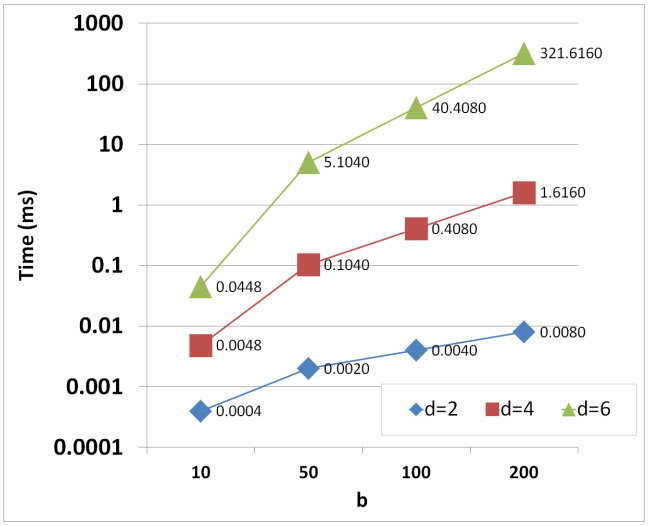

Figure 3. Time needed for certificate graph search on a device with 0.05 GTEPS graph processing capability, depending on d and b

Furthermore, our scheme assumes that all users share a common web-of-trust among each other. This may be tricky to achieve in reality, where users use various social networks and different mobile network operators exist in most countries. For instance, in a mobile network operated by a network operator, this network operator could in principle take the role of offering a keyserver that frequently publishes WoT-files. But then only users of this operator are likely to benefit from the scheme. In case of a multi-operator scenario, the keyservers that maintain the web-of-trust would need to be operated by an outside entity, or by multiple operators together. One possibility to make our approach work on a large scale is a as follows: Users would form a general, multi-operator Web-of-Trust (where keyservers are maintained by an outside entity or jointly by the operators together), and in case of a disaster mobile networks would allow roaming users to post and retrieve urgent messages similar to allowing emergency calls in today's network.

III. Evaluation

In this section, we estimate the scalability and performance of our proposed approach. In particular, we are interested in studying the applicability of our scheme in a

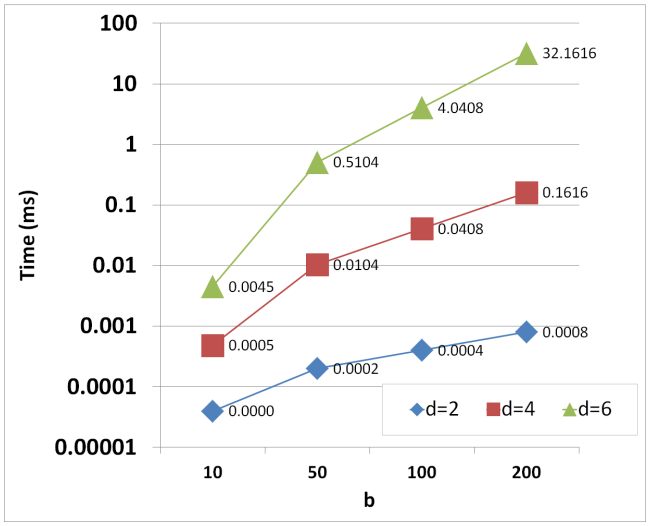

Figure 4. Time needed for certificate graph search on a device with 0.5 GTEPS graph processing capability, depending on d and b

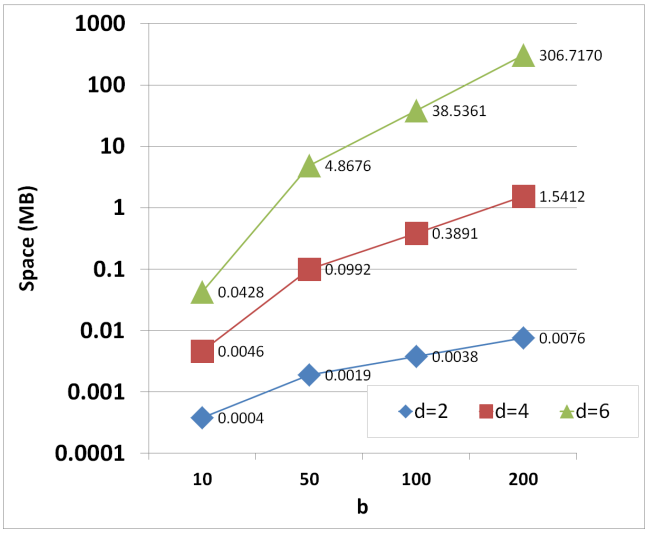

Figure 5. Memory needed for certificate graph search given a WoT key-ID size of 160 bit, depending on d and b

completely decentralised fashion, i.e. D_{WoT}-nodes running on end user devices. Thus, we analytically estimate how the $dBFS$ algorithm scales with increasing WoT sizes, and how it would perform on common smartphones.

A. Certificate Graph Search Performance

The time-complexity and space-complexity of a double-sided BFS algorithm is both $O(b^{\frac{d}{2}})$, where b is the branching factor and d is the depth of the search [5]. In our case, d is the certificate path length and b is the average number of identities a member of the WoT has signed. The average time t() and space s() needed for a dBFS algorithm can be derived as follows (assuming even values of d, i.e. the depth of the search[6]):

$$t(dBFS) = 2 \times \left(b^{\frac{d}{2}} + b^{\frac{d}{2}-1} + \ldots + b\right) \quad (1)$$

$$s(dBFS) = m_{req}(node) \times 2 \times \left(b^{\frac{d}{2}} + b^{\frac{d}{2}-1} + \ldots + b\right) \quad (2)$$

where $m_{req}(node)$ refers to the space needed for storing a single node (i.e. as graph vertex) in memory. Note that Equ. 1 and 2 are conservative estimations (i.e. likely over-estimations) of the average time and space needed, because traversal of all nodes at depth d is assumed, while a given search might terminate earlier.

A node in the certificate graph can be represented by its WoT key-ID. Assuming SHA-1 as the cryptographic hash function $h()$, the space for storing a key-ID, $m_{req}(node)$, is 20 bytes. Today's smartphones can achieve a *Traversed Edges Per Second* (TEPS) rate in graph processing of up to 0.5 GTEPS (Billion TEPS) [6]. In other words, modern end-user devices can process up to 5×10^8 links per second in graph search.

Given these assumptions, Figure 3 shows the time needed to find a certificate chain for different values of b and d, assuming a moderately powered end user device, i.e. with 0.05 GTEPS (Figure 4 shows these results for a powerful smartphone, i.e. with 0.5 GTEPS, respectively). It can be observed that even for large values of b (the average node degree) and values of d (the maximum certificate chain length considered) of up to 6, a certificate chain can be found in less than 350ms even on devices with low graph processing power (or less than 35ms for powerful devices, respectively). A depth of $d = 6$ would mean that certificate chains of up to length 6 would be considered to imply trustworthiness regarding the WoT key-ID. Longer certificate chains are most likely not useful in assessing key-IDs, as the risk of (un-)intentionally wrong assessments of other users' behaviour grows with each certificate in the chain. Thus, even though time scales exponentially as d increases, this is not a problem for our use case as only small certificate chain lengths are meaningful to use for assessment of messages. Note that this also implies that the overall WoT size is not relevant for the algorithm's performance as only a search for a certificate chain up to depth d is performed. Note that in the Facebook social graph, the median for the node degree is 99 [7]. Hence we can consider values of $b = 100, 200$ as very large estimates for b in a WoT.

Figure 5 shows the total amount of space needed to execute a $dBFS$ algorithm, for different values of d and b. The results show that for $b = 200$ and $d = 6$, i.e. a very large WoT where long certificate chains are still considered to imply trustworthiness, slightly more than 300MB of memory are needed to execute the $dBFS$ algorithm. In all other cases, less than 100MB of memory are needed.

B. Size of Compressed Certificate Graph

A format for storing WoT certificate graphs in a compressed way has been proposed in [4]. In this 'WoT-file' format, first real-world identities are listed sequentially, followed by a list of the corresponding key-IDs in the same order. Then, a third sequential list lists for each key-ID an array of other key-IDs it has signed. Hence, the total size of such a file, f_{wot} is given by

$$size(f_{wot}) = n \times (m_{req}(name))_{av} + n \times m_{req}(node) + n \times b \times m_{req}(node)\ [byte] \quad (3)$$

where n is the total number of nodes (i.e. real-world identities) in the WoT and $(m_{req}(name))_{av}$ refers to the

[6]For uneven values of d, only one BFS would need to expand to depth $\frac{d}{2}$ while the other BFS would only need to expand to depth $\frac{d}{2} - 1$.

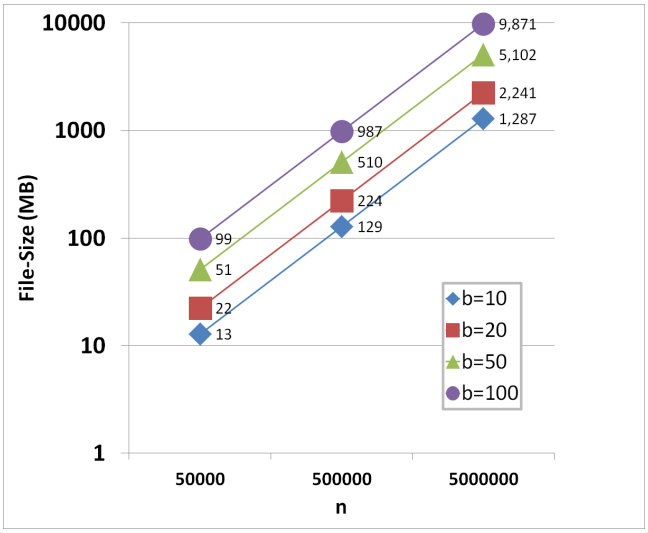

Figure 6. Space requirements for storing compressed WoT certificate graph, depending on n and b

average space needed for storing a real-world identity. Note that Equ. 3 expresses the *non-transient* space needed to *permanently* store a WoT file, e.g. on flash memory. Equ. 2 expresses the *temporary* space requirement during the execution of the $dBFS$ algorithm, i.e. the *transient* memory requirements (e.g. amount of RAM).

Figure 6 shows how the WoT-file size scales with the total number of users, n, depending on b, assuming an average real-world identity size of 50 byte (e.g. a combination of actual user name and email address) and a key-ID size of 20 byte (i.e. $160bit$ as with SHA-1). It can be observed that file size scales linearly with the total number of users, n, and also with the average degree in the social graph, b. However, the results reveal that actual file sizes grow quickly with increasing n or b. Only for WoTs with less than 50.000 users a file size of less than 100MB can be achieved. Even for small values of $b = 10, 20$ (which may be realistic, considering that only a subset of all contacts may be considered trustworthy by users), file sizes are larger than 1GB for a WoT with 5 Million users.

C. Discussion of Evaluation Results

In summary, our results show that the $dBFS$ algorithm could be *executed* solely on even moderately powerful end-user devices for quite large social networks among users. However, the *storage size* of compressed WoT-file grows quite large even for a moderately sized social network. This implies that our scheme allows for message assessment in a completely decentralised fashion (i.e. running solely on end-user devices) only in cases where the overall WoT size is moderate (e.g. in the order of up to 100.000 users).

At the same time, the results show that even for large WoTs and social degrees, b, the scheme can be applied at decentralised D_{WoT} nodes such as base stations. Such nodes can be expected to have at least the computational algorithm execution power as modern smartphones, and moreover are surely capable of storing WoT-files in the order of several Gigabytes.

Note that there are several ways to increase the overall scalability of our scheme in order to allow the processing of larger WoT sizes on end-user devices: 1) Instead of sending complete WoT files each time, only the delta to the last version has to be transmitted. While this does not help in saving storage space at devices, it would significantly decrease the amount of data that would need to be transmitted frequently; 2) Experience with actual PGP WoT-files shows that such files can be compressed by on average 60% using regular compression techniques such as gzip [4]; 3) Hash extension techniques[7] [8] could help to make the key-ID size smaller. This would decrease $m_{req}(node)$ (e.g. by 50% if a hash of only 80-bit could be considered secure using such techniques, which is not unrealistic [8]), and thus significantly decrease overall file sizes.

IV. Conclusion

In this paper, we have presented a concrete mechanism for using a Web-of-Trust in conjunction with self-certifying names in a specific scenario: fragmented (mobile) networks, where connectivity to centralised entities and authentication servers is not available. In such a scenario, our scheme allows users to assess the trustworthiness of messages they receive from untrusted, third-party devices. Our analytical evaluation has shown that the time and space requirements for certificate chain search are acceptable for smartphones common on today's market. However, our evaluation has also revealed that while the storage requirements for the certificate graph scale linearly, these requirements are significant for any but moderately sized social networks, rendering actual file sizes potentially too large for end user devices.

In summary, our results reveal that our scheme can be applied solely on end-user devices for moderate Web-of-Trust sizes of up to 100,000 users, assuming a social friend degree of less than 100 and storage capabilities of 200MB on users' devices. For larger Web-of-Trusts, our scheme is still applicable in a decentralised fashion through functioning access network nodes, such as base stations, that aid end-user devices in WoT graph search. As future work, we intend to evaluate our scheme in-depth via prototypical implementation on Android devices and to investigate ways to increase the scalability of our scheme even further.

References

[1] T. Aura, "Cryptographically generated addresses (cga)," in *Proc. of Information Security, 6th International Conference, ISC 2003*, ser. LNCS, Springer, Ed., no. 2851, October 2003, pp. 29–43.

[2] A. Ghodsi, T. Koponen *et al.*, "Naming in content-oriented architectures," in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, 2011, pp. 1–6.

[3] M. Arumaithurai, J. Seedorf *et al.*, "Using icn in disaster scenarios," Internet Engineering Task Force, Internet-Draft draft-seedorf-icn-disaster-01, Oct. 2013, work in progress. [Online]. Available: http://tools.ietf.org/html/draft-seedorf-icn-disaster-01

[4] J. Cederlof, "The Web of Trust .wot file format," Nov. 2004. [Online]. Available: http://www.lysator.liu.se/ jc/wotsap/wotfileformat.txt

[5] S. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach*, 3rd ed. Prentice Hall, 2003.

[6] "Graph crest," website. [Online]. Available: http://www.graphcrest.jp/eng/news2013-11-b.html

[7] J. Ugander, B. Karrer *et al.*, "The anatomy of the facebook social graph," Techreport, 2012, http://arxiv.org/pdf/1111.4503.pdf.

[8] T. Aura and M. Roe, "Strengthening short hash values," Techreport, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.145.7681.

[7] Essentially, a second hash-value is used, where the m leftmost bits must be 0. Using this technique, the effective hash length can be incremented by m bits.