Strengthening a Curated Web of Trust in a Geographically Distributed Project

Gunnar E. Wolf Iszaevich[*†] and Gina Gallegos-García[*]

[*]Sección de Estudios de Posgrado e Investigación
ESIME Culhuacan, Instituto Politécnico Nacional
Av. Santa Ana 1000, 04430, Ciudad de México, MEXICO
[†]Instituto de Investigaciónes Económicas, Universidad Nacional Autónoma de México
Cto. Mtro. Mario de la Cueva S/N
Ciudad Universitaria 04510, Ciudad de México, MEXICO

This is an Accepted Manuscript of an article published by *Taylor & Francis* in *Cryptologia* on December 27, 2016, available online: http://www.tandfonline.com/doi/full/10.1080/01611194.2016.1238421

Abstract

In this paper we present a trust management scheme, derived from the horizontal and almost anarchic *Web of Trust* model, but following a curatorship step that allows it to become a centerpiece for authentication in Debian, one of the largest and longest lived free software projects and producer of the eponymous GNU/Linux software distribution. This is done by analizing the experience gained through a large-scale key migration process that spanned five years and nearly 100% of the originally existing keys, carried out attempting to minimize loss of keyring connectivity and strength, while keeping up to date with the best current security practices.

Keywords Trust models, key length, PGP, Web of Trust

1 Introduction

The Debian project, creator of the Debian GNU/Linux distribution, is among the largest and longest lived free software projects. It is a completely volunteer-driven, community-run project, with a very big geographic dispersion. Due to the project's very nature, it requires the use of strong cryptographic mechanisms throughout its infrastructure.

The advent of public key cryptography in the late 1970s can be clearly seen as a gamechanger. Particularly, the breakdown of the different security properties that cryptography provides, and encryption, identity assessment and message signing abilities are separate parts of the cryptographic universe, and no longer just inferred by the knowledge of a given symmetric key.

Keeping public key cryptography useful for years after a key is created is feasible, however, the *key strength* is strongly affected by available computing power: As computers grow more powerful, solving problems deemed *too hard* only ten years ago is within reach; all of the desired properties of a cryptographic key grow weaker with time as its keylength (formally analogous to a *security parameter*, measured in bits) falls within reach of its potential attackers. Such is the case in public key cryptography: While fifteen years ago the standard recommended key length was 1024 bits, today forging a key of such length is within reach of large organizations or national governments, and the minimum recommended standard is 2048 bits (being 4096 preferred).

Large-scale migrations can be however very hard to achieve, as keys must be accompanied by identity verification. For some settings, this issue is solved by placing the burden of repeated identity assurance on the individual key holder (and not of the certification entity).

Debian's identity management scheme is based on the horizontal Web-of-Trust (WoT) model, instead of the hierarchic Certificate Authorities in widespread use in the commercial world, but it has what we define as a *curated* keyring. The need of a large-scale key migration, affecting at some point almost 100% of the existing keys, was detected back in 2009, but delayed due to its complexity and the degradation it would cause in the project keyring's strength. By late 2013 the situation remained bleak, with only close to 30% of the keys having migrated.

Through aggressive deadlines and social work, the migration was completed by early 2015; while over 200 keys were cut off the project, we argue this corresponds mostly to keys of already inactive developers and helps bring the project's security in line with reality.

This paper presents the situation at greater depth in Sections 2, 3 and 4, delineating the main points in the process, and presents as our results the closing numbers per year after the hard-migration was finalized. Finally, we present a proposed plan of action with prospective measures for future strengthening work.

2 Preliminaries

This Section illustrates several basic concepts to understand the problematic and development of the problem being studied.

2.1 Trust models in public key cryptography

Secure communications are possible in large-scale, open networks such as the Internet thanks to public key cryptography, a group of algorithms that allow two related numbers –the *private key*, that should be jealously guarded by each communicating party, and the *public key*, that can be publicly disclosed– to provide the assurance of integrity and confidentiality in communications.

There are many algorithms based on this idea; for many decades, the most commonly used one have been built on modular arithmetics — RSA (Rivest, Shamir, and Adleman 1978) and DSA (FIPS 1994–2013); there are several alterntive algorithms with interesting properties that could end up replacing them in the future, such as Elliptic Curves (Koblitz 1987; Miller 1986), but thanks to their relative simplicity and to the large install base of compatible implementations, the aforementioned algorithms will probably remain dominant for a long time.

However, while said algorithms solve the issue of communicating two endpoints that have already verified each other's identity, in order to be used in today's networks there needs a way for keys to be communicated, and for their identity to be trusted. Thanks to the introduction of *public key distribution systems* (Merkle 1978; Diffie and Hellman 1976), it is possible for two parties to securely exchange their keys in an open network. However, a determined attacker can intercept communications *prior to the key exchange* and impersonate either party while communicating to the other — an attack known as a *Man in the Middle* or *MitM*.

This is where *Public Key Infrastructure* trust models come into play: As a given key can come from legitimate or fraudulent parties, a key can be *certified* with the signature of a trusted third party. There are two major trust models:

Certificate Authorities (CAs) A centralized, hierarchical model, based on *ultimately trusted* agents (Housley et al. 1999; Yee 2013). Users trust a set of *root certificate authorities*, and every certificate that is presented should have a *trust path* leading to one such source of ultimate trust.

This is the most widely deployed model, used for the SSL/TLS protocols, which includes all of the encrypted Web traffic (Durumeric et al. 2013). This model is mostly apt for client-server communications, where usually the only meaningful identity that has to be established is the server's, while the clients can connect anonymously, or authenticate via third methods, usually communicated over the established encrypted channel.

Web of trust (WoT) A much less used and studied model; this is a decentralized model, where there is no *ultimate trust* defined at any point. Trust paths are also used, but instead of trust *flowing down* from few authorities, it *steams up* from each of the system's users: Every acting entity in the system can validate the identity of some of its peers, and will trust a given key based on all of the other people that trust it as well (Network Associates, Inc. 1990–1999; Abdul-Rahman 1997).

This model is mainly used in the OpenPGP systems, used originally for e-mail privacy and certification; this model is horizontal, in the sense that every sender is also a receiver, and identity validation is usually carried out in both directions at different points in time.

PGP, the first OpenPGP implementation and still a reference, was meant for e-mail (Zimmerman 1991), but given that its encryption and/or signa-

ture covers *arbitrary static documents*, it has been adopted for many other uses, as it will be shown here.

Every user of said trust models needs to have a list of the *trust sources* in use. In the *WoT* model, the set of keys through which operations are carried out –both for trust assessment or for any other cryptographic operations– is known as a *keyring*.

2.2 Cryptographic key length

The basic operation for public-key cryptosystems was defined back in the mid to late 1970s on different algorithms, and their oldest implementations still in widespread use (Phil Zimmerman's *Pretty Good Privacy*, best known as *PGP*) dates from 1991 (Zimmerman 1991; Back 2001). The PGP implementation is very flexible; it allows its cryptographic material to be implemented through different protocols. The main algorithms that have been historically used with PGP are RSA and DSA; recent versions have added support for elliptic curve-based algorithms, but they are not yet in widespread use.

When said cryptosystems underwent a standarization process for industry and general public use, the National Institute for Standards and Technology (NIST) specified several algorithms acceptable for message signatures as part of the Digital Signature Standard (FIPS 1994–2013) — particularly, RSA (based on the *integer factorization* problem, Rivest, Shamir, and Adleman 1978) and DSA (based on the *discrete logarithm* problem and based on ElGamal 1985).

Both the *integer factorization* and the *discrete logarithm* varieties are based on *trapdoor permutations*, that is, functions that are much easier to perform in one direction than in the inverse without the knowledge of special information (the trapdoor).

In both cases, however, defining work as *much easier* to do is necessarily flexible: As more computing power is available, both the public and secret parts of said computation have to be much larger to resist factorization — this is known as the *security parameter*, and is usually expressed as the size (in bits) needed to represent the *keys* under different cryptosystems.

When originally presented, PGP worked with 384, 512 and 1024 bit keys. 500 bits for said keys achieve approximately the same entropy (thus security) than 48 bits for symmetric encryption (Smart 2012) — Which is nowadays, according to the same study, within reach for small organizations to break in a very short term.

Throughout the years, both RSA and DSA key length has grown, and new keys and certificates nowadays usually start at 2048 bits, with 4096 being the norm. However, while for many years cryptography was used by very few people, in the last fifteen years it became a fundamental part of the Internet. Keys and certificates are no longer something seldom used, but something that affects all users and businesses. Thus, old keys now have the need to be *migrated* — Or rather, key users must be migrated from using shorter to using longer, unrelated keys.

This issue posed only a minor inconvenence for the public hierarchical *Certificate Authorities* model, as said authorities have demanded the certificates submitted for signing to have a set expiry date, and it suffices for the authority to change its policies not to accept keys below a certain length or using the approved algorithms[1] for its whole user base to migrate to stronger keys after an expiry period; publicly trusted (this is, following the current versions of the CA/Browser Forum recommendations) certificate authorities should not issue certificates using said weaker algorithms after January 2016 (CA/Browser Forum 2015).

It must be noted, however, that organizations can use either *self-signed* certificates or internal-use certificate authorities; an interesting case is exemplified by the USA Department of Defense still issuing insecure certificates in January 2016 (Mutton 2016).

For the decentralized *Web of trust* model, however, it poses a much harder problem: With no central entity able to preemptively limit a key's validity, users of short, vulnerable keys might still be using them years after they have been determined as insecure. Unsuspecting third parties who trust users' identity will not be prompted into requiring stronger cryptographic material.

2.3 Measuring keys and keyrings

There are many measures by which any given key, or a keyring as a whole, can be measured; the following are the most basic and simple reachability and distance-based measurements (Penning 2006–2016):

Shortest distance Individual measure of a key pair. Taking any two keys in the keyring, the length of the shortest path between them, irrespective of whether they are part of the strong set; can be infinity in case no such path exists. This measure is *directonal*: Trust paths (and thus their lengths) from A to B can be different than from B to A.

Strong set Group measure within a keyring, the largest subset of keys between which there is at least one trust path from each to all of the others; it can be relevant either in absolute terms or as a percentage of the whole keyring.

Reachable set The union of the strong set and all of the keys that are reachable from it, although do not link back.

Mean shortest distance (MSD) Individual measure of any given key belonging to a set (usually, only the strong set is relevant for this measure). It is the average of the shortest distances from all of the other keys in said set.

[1] A problem analog to the one covered by this Section affects the widely used message *SHA-1* message digest algorithm. While said algorithmic vulnerability falls outside the scope of the present work, the mitigation and migration processes are equivalent.

Average mean shortest distance (AMSD) Global measure for any key set (usually the strong set) of any given keyring. It is the average of the MSDs of all of its keys.

3 The Debian project and its curated keyring

We now proceed to analyze the specifics for the keyring migration of the Debian keyring: In first place, the specific context of the project at hand, and secondly the experience and learnt lessons from said process.

3.1 The Debian project

The Debian project is one of the largest *free software* undertakings to date (Debian Project 1997–2015), and has held this position for well over ten years (González-Barahona et al. 2001; Amor et al. 2005). Founded in 1993, it is also one of the longest lived *free software* projects still under active development (Fernández-Sanguino et al. 2015). It has seen participation by over 5,500 contributors (Zini 2013–2016).

Debian is a very interesting social case study, as it exists merely as a work group with no direct backing or financing company; all participation in Debian is voluntary (although different companies do pay their employees full-time to work on Debian) (Monga 2004). The mere social implications of this project goal and self-definition have been studied at length from very different angles (González de Requena Redondo 2010; Siobhán O'Mahony 2012; Coleman 2013).

The project is based upon a series of documents describing its goals and work ethos, such as the Social Contract, Constitution and Policy (Debian Project 1997–2004; Debian Project 1998–2015; Debian Project 1996–2015); this has allowed for the close to 1500 curent project maintainers and developers to be globally distributed, as Figure 1 shows, although undoubtedly following a distribution closely related to the industrialized countries (Debian Project 2015).

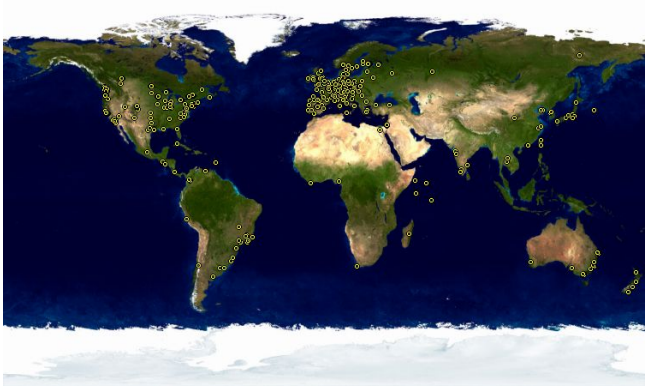

Figure 1: Geographical distribution of Debian Developers (Debian Project 2015)

Debian produces the largest Linux-based distribution, measured both in size (with over 30,000 *source packages*, independent programs integrated for it) and in adoption, particularly when its *derivative distributions* are also counted in — Debian is the *upstream distribution* for Ubuntu, Mint, Kali, and literally hundreds of other organized collections of free software (Byfield 2011; Lundqvist and Rodic 2006–2012).

The project's global dispersion, size and wide adoption are the main factors for the importance of this study: Given the high responsability associated with upload rights to Debian, any particular developer could potentially upload malicious code, which would reach easily millions of computers, and would be run with administrative privileges.

Such an abuse has not happened so far, and care is taken not to open the window for any fraudulent activity of this type. This requires strong identity assurance.

3.2 Strong identity assurance: A *curated* Web of Trust

The most widespread identification method worldwide is the old username-password pair. This scheme, however secure it can be made, is just too weak for today's standards; even security-minded users tend to pick trivial passwords (Burnett 2011) or repeat their use in unrelated servers, and even in the best case, an alphanumeric keyboard-typable 10 character long password will not yield enough entropy to be considered secure.

Cryptographic key certificates, as described in Section 2.1, allow for *extremely improbable to be forgeable* documents, validated by third parties (be it in the CAs model or in the WoT model). The Debian project adds one further step to the WoT model: A *curatorship*.

Besides asserting the identity of each project member, the Debian *keyring-maint* team maintains a list of keys that define a *set of keyrings*, each of which has an associated level of privileges in the project. Every Debian Developer, both uploading and non-uploading (DD and DD-NU), and Debian Maintainer (DM) should[2] have a valid key in the corresponding keyring. *keyring-maint* is also in charge of ensuring keyrings are kept up to date and ahead of any weaknesses that can appear in the used public key cryptosystems.

The team also pushes to keep a high *strong set* ratio (see Section 2.3); although exceptions to this rule are granted as required by the *Debian Account Managers*, mainly on account creation (as Section 3.3 explains), the mere existence of a policy requiring two active DD signatures for every requested key transition (Debian Project 1997–2014) has resulted in a more richly integrated social aspect of the project, with keysigning being an important activity pursued worldwide when Debian-related people meet.

Do note that the *keyring-maint* team's work is hosted at a version-controlled Git repository. While the master, working copy is not publicly available, a public

[2]Ideally, there should be a 1:1 relation between keys and people, but as we will soon describe, for some time it has not necessarily been the case.

copy of all of the committed work (which includes the full historic data used as the source of information for this work) is available and can be *cloned* from `https://anonscm.debian.org/gitweb/?p=keyring/keyring.git`

From what was presented so far, the main elements relevant to the present work are:

- Debian is a geographically very disperse project, with no formal organization behind it.
- Strong cryptographic identity validation is a must for the project's nature.
- The standard cryptographic protocol over 20 years ago is no longer considerable secure enough.
- Key migrations must be done preserving a strong identity trust: The *keyring-maint* team requires two signatures from active DD keys to perform a key transition, unless requested to act otherwise by the *Debian Account Managers*.

As Debian has grown, so has the number of developers. While back in 2000 the strong set spanned just about a hundred nodes (as can be seen in the Developers keyring's strong set depicted in Figure 2), it nowadays covers close to 750.

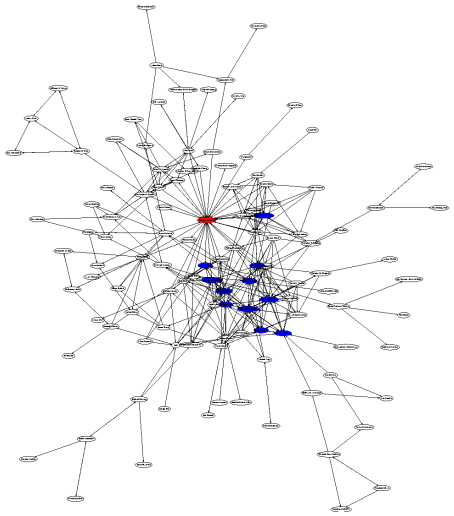

Figure 2: Strong set in the Developers keyring of Debian in 2000, highlighting the keys with highest centrality (Darxus 2002)

The Debian keyrings are mainly used by the project's infrastructure; the scripts that immediately determine if a package upload or a vote cast for a General Resolution (GR) come from an authorized person do so by checking

the keyrings; changing the password for logging in to the project servers is also done via a signed message.

Keyrings are also available for system users to download, via the *debian-keyring* package.

3.2.1 Defined keyrings

The team manages the following four *active* keyrings:[3]

Keyring Currently active keys corresponding to *Debian Developers (uploading)* (DD). Before 2009, this was the only existing keyring, and with over 800 currently active keys, it is by far still the largest one.

Nonupload Currently active keys corresponding to *Debian Developers (non-uploading)* (DD-NU). It is the smallest and youngest active keyring: It was defined in late 2010, and as of this writing holds only 17 keys.

Maintainers Currently active keys corresponding to *Debian Maintainers* (DM). It was defined in mid 2009, and currently holds 234 keys.

Role-keys Keys belonging to Debian teams, not individuals, such as the *Security team* or the *Debian CD*.

From those four, only *Keyring* and *Nonupload* are considered as *Debian project members* and, as such, *sources of trust* — This means that only the signatures from keys belonging to keys in those keyrings are considered for identity trust paths.

The following *inactive* keyrings are defined as well. Those keyrings hold keys no longer active, but which are part of the project's historic memory:

Emeritus Inactive keyring for former Debian Developers that have formally retired from the project.

Removed-1024 Keys that were removed from the project as part of the process presented in Section 4.2 and not yet replaced with a suitable longer key.

Removed Other keys that have been removed for various reasons during the keyring's history.

3.3 Curatorship and keyrings

Section 3.2 presents the need for the curatorship process in the Debian project. This process is delineated by a set of well-defined actions resulting in a key entering, leaving or changing status in one of the managed keyrings.

We characterize the work done with the Debian keyrings as a *curatorship* as an analogy with the work done for art collections and exhibits. Quoting Nauman and Kraynak (2005) on the *role of the curator*:

[3] This work does not attempt to describe the differences and relations between the different Debian contributor types; further details can be found at (Debian Project 1997–2016).

> «(...) The "somebody else" who is usually primarily responsible for how and where a work of art is shown is the curator. The curator selects a work for exhibition and makes decisions about the context within which it will be displayed. This requires sensitivity to the interests and intentions of the artist. The curator also needs to ensure that the work is displayed in such a way that it is accessible and meaningful to the public. Furthermore, curators working within a museum environment, have an added responsibility to their institution. It is their role, along with conservators and art technicians, to delineate a comprehensive and accurate record of the artwork for the future.»

And quoting Eeds and Peterson (1991) on the role of a curator, the article very adequately starts with the following paragraph:

> «A curator knows art, collects it, cares for it, and delights in sharing it with others, helping them see it in ways they may not have discovered if left on their own. We choose this term for teachers who do these things with literature, opening this way of knowing the world to their students.»

Although not concerned with works of art or the relationship between content creator and public, we present the *curator* as the person o team making the difference between having a *set of items* and a *collection of items*: A collection implies basic quality criteria to which all of its items conform, and a wholeness or unity between them all.

The Debian *keyring-maint* team cares for the quality and meaningfulness of the defined keyrings. This does not just mean trying to keep the keyrings' strong sets tight and AMSDs low (as defined in Section 2.3), but continuously evaluating the changing conditions and challenges in the domain where the keyring exists (the cryptographic algorithms and their implementation), centrally ensure the trust conditions are not at risk of losing quality.

All in all, having a curated keyring in this context means maintaining and applying a set of guidelines to keep a set of items (in this case, of PGP keys) as a *valuable* collection.

Of course, adding this measure of control does bring the WoT model closer to the centralized CA model: There is a single entity controlling the acceptance, permanence and retirement of nodes. We hold the curatorship to be qualitatively different because it effectively separates the validation of a key's identity (stemming completely from the WoT) from the conformation of the accepted key set (overseen by the *keyring-maint* team).

4 Case study: The migration away from 1024-bit keys in Debian

The different keyrings have been managed under a version control system since mid 2008, so the baseline of our work can only track back to said date; we can appreciate in the evolution of the percentage of the strong set respective to the full developer keyring, shown in Figure 3, that connectedness has remained almost constant thorughout the last seven years: Even though there is an undeniable reduction and jitter at the end of this period due precisely to the migration we are describing, the strong set has consistently remained between 82 and 89% of the full keyring.

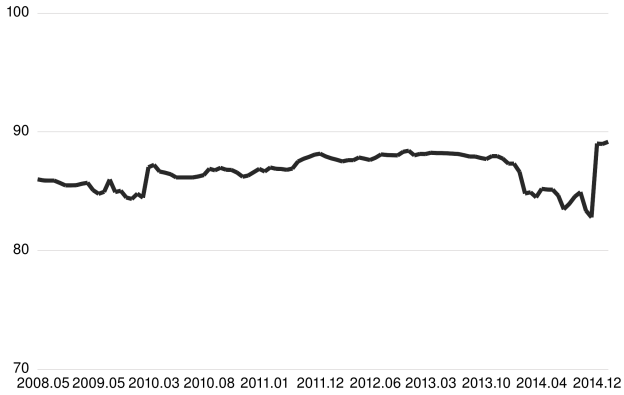

Figure 3: Strong set (percentage of total) for the Developers keyring, 2008-2015: 82 to 89% of the total keyring

4.1 Earlier migrations: The PGPv3 keys removal

By 2014, it was clear that 1024-bit keys did not provide the security level required by the project and that the problem was big enough it would not be solved by just calling attention to it, so a hard migration strategy with mandatory cutoff periods had to be sketched. This was not unprecedented: By 2010 and after a migration period slightly over a year (McDowell 2009), *keyring-maint* managed to get all (yet) older PGPv3 keys to be retired due to weaknesses in the MD5 message digest algorithm they used, as well as their short keylength. As Figure 4 shows, when the drive away from PGPv3 keys started in 2008, close to 240 developers had such keys, and it was until August 2010 that the number was finally driven to zero.

There were core differences between that experience and the newer one. Firstly, the amount of keys in need to be migrated was close to 20% of the project's membership.[4] Secondly, a large percentage of old PGPv3 keys were

[4] As PGPv4-handling software was not yet mature at some stage, developers were allowed

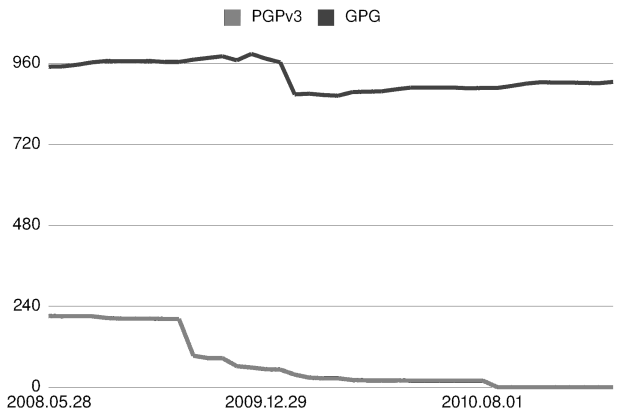

Figure 4: Migration away from the PGPv3 key format in the Developers keyring, 2008–2010

of effectively inactive developers. When a response request (*WAT rounds*) was sent out by the Debian Account Managers, keys belonging to developers that failed at all to answer to the mails were marked as disabled; the last "dip" of the PGPv3 line, in August 2010, marks the necessary forceful removal of the only 17 remaining active PGPv3 keys.

4.2 The 1024-bit keys removal

As Figure 5 shows, in early 2014, when the migration efforts started, the amount of keys in need of migration was slightly over half of the active keyring. There was already a downwards trend in it favoring stronger keys since attention was first drawn to the matter in mid 2009, when it was close to the total amount of keys in the keyring, but the pace was too slow for it to naturally converge.

At the annual Debian conference (DebConf 14) in Portland, Oregon, the migration plan was announced: The full, final migration would be carried out over the following six months, and a strong invitation would be sent to every affected developer to upgrade their keys *as soon as possible*. A cutoff date was set to January 1st, 2015, by which point all remaining keys would be disabled and handled as not trusted. The *keyring-maint* team asked the approximately 200 developers attending the session to support the plan (which was received with no opposition), and set on to implement it.

The response was overwhelmingly good; over the next six months, 266 key replacement requests were successfully handled. Nevertheless, 252 keys had to be removed, each of which means a developer effective losing the ability to perform their Debian-related duties. One year later, the number of such keys that have not yet been replaced remains at 227.

to have both a PGPv3 and PGPv4 keys.

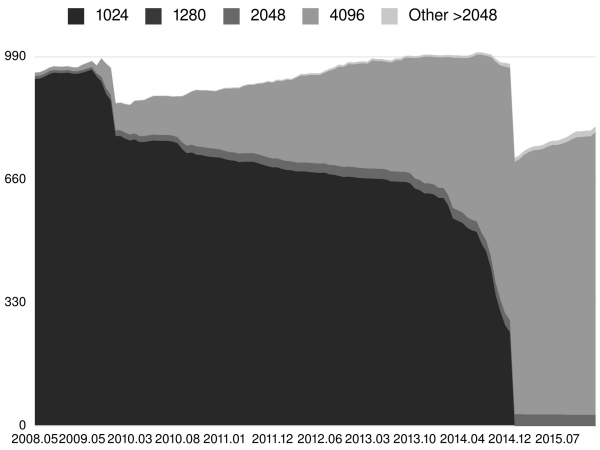

Figure 5: Historical evolution of the Developers keyring, 2008-2016

5 Conclusions

Throughout this article, we have presented the *curated Web of Trust* model, which, although is directly derived from the almost anarchic, transitive *Web of Trust*, allows for the existance of a group of *curators* (in the case of Debian, the *keyring-maint* team) that oversee and determine policies that the participating keys must comply. This, of course, without falling into the opposite model, the rigid and hierarchical *Certificate Authorities* model, that are based on the vulnerable and not fully auditable *ultimately trusted agents*: In the *curated Web of Trust* model, the curators' keys get no different treatment than any other member's key; while *keyring-maint* members technically could alter their own keys, given that the keyring is generated from a publicly available Git tree (see Section 3.2), all changes are publicly logged and refered to the relevant request tracker ticket, keeping full transparency and accountability.

The described model has long been used at the Debian project, proving a natural fit due to the project's geographically distributed, voluntary participation nature. Many of the project members will never have the opportunity to mutually meet, but maintianing a large keyring (in the vicinity of 1000 independent keys) with a strong set spanning 82% to 89% of the set for eight years shows that the model enables a stable path of trust to exist between most project participants.

Finally, throughout the experience presented as the migration away from weaker keys (shorter than 2048 bits), we sustain that, by setting a few clear replacement requirements, this model is able to undergo large-scale transitions without significantly losing inner connectivity. The keyring did suffer, as almost one fourth of the keys had to be retired without their owners' being yet able to comply with the requirement, but the rate at which they are submitting their

keys for inclusion one year after the migration was done (as shown in Figure 5) is clearly higher than the replacement race before the aggressive migration started.

5.1 Future work

The present work has prompted us to look into several health aspects of the Debian keyrings. While up to now the main curatorship criteria have been strong enough cryptography and ensuring conectedness on new key additions, further studying the keyring can bring up insights on new policies to adopt.

A natural match can be found between a deeper keyring analysis following social network methodologies, with keys as nodes and trust signatures as directed edges. Work is currently underway, and first results have presented (Wolf 2016), on performing an aging and vitality analysis on the set of signatures; with said work, the effect of not counting signatures over a given age threshold will be better understood, and might allow to shape policies to ensure the connections depicted by the strong set are actually socially meaningful, and not just decade-old random signatures with no true current real meaning.

Of course, similar analysis could be run on different projects. While Debian is the biggest identifiable project using a PGP Web of Trust (Cederlöf 2004), many other large projects in the Free Software environment can be analyzed.

References

Abdul-Rahman, Alfarez (1997). "The pgp trust model". In: *EDI-Forum: The Journal of Electronic Commerce* 10.3, pp. 27–31. URL: http://system.khl.mydns.jp/college/WOT/The_PGP_Trust_Model.pdf (visited on 01/16/2015) (cit. on p. 3).

Amor, Juan José et al. (2005). "From pigs to stripes: A travel through Debian". In: *Proceedings of the DebConf5 (Debian Annual Developers Meeting)*. Citeseer. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.133.797&rep=rep1&type=pdf (visited on 01/13/2016) (cit. on p. 6).

Back, Adam (2001). *PGP Timeline*. URL: http://www.cypherspace.org/adam/timeline/ (cit. on p. 4).

Burnett, Mark (2011). *10,000 Top Passwords*. URL: https://xato.net/passwords/more-top-worst-passwords/ (cit. on p. 7).

Byfield, Bruce (2011). *Linux Leaders: Debian and Ubuntu Derivative Distros*. Datamation. URL: http://www.datamation.com/osrc/article.php/3926941/Linux-Leaders-Debian-and-Ubuntu-Derivative-Distros.htm (visited on 01/13/2016) (cit. on p. 7).

CA/Browser Forum (2015). *Baseline Requirements Certificate Policy for the Issuance and Management of Publicly-Trusted Certificates*. URL: https://cabforum.org/wp-content/uploads/Baseline_Requirements_V1_3_1.pdf (visited on 01/12/2016) (cit. on p. 5).

Cederlöf, Jörgen (2004). *Dissecting the leaf of trust*. URL: http://www.lysator.liu.se/~jc/wotsap/leafoftrust.html (cit. on p. 14).

Coleman, E Gabriella (2013). *Coding freedom: The ethics and aesthetics of hacking*. Princeton University Press. URL: http://gabriellacoleman.org/Coleman-Coding-Freedom.pdf (visited on 01/13/2016) (cit. on p. 6).

Darxus (2002). *Graphing the Debian Keyring Web of Trust*. URL: http://www.chaosreigns.com/code/sig2dot/debian.html (cit. on p. 8).

Debian Project (1996–2015). *Debian Policy Manual*. URL: https://www.debian.org/doc/debian-policy/ (visited on 01/13/2016) (cit. on p. 6).

— (1997–2004). *Debian Social Contract*. URL: https://www.debian.org/social_contract (visited on 01/13/2016) (cit. on p. 6).

— (1997–2014). *Rules for key replacement in the Debian keyring*. URL: http://keyring.debian.org/replacing_keys.html (visited on 01/13/2016) (cit. on p. 7).

— (1997–2015). *About Debian*. URL: https://www.debian.org/intro/about (visited on 01/13/2016) (cit. on p. 6).

— (1997–2016). *Debian New Members Corner*. URL: https://www.debian.org/devel/join/newmaint (visited on 02/22/2016) (cit. on p. 9).

— (1998–2015). *Constitution for the Debian Project (v1.5)*. URL: https://www.debian.org/devel/constitution (visited on 01/13/2016) (cit. on p. 6).

— (2015). *Developer locations*. URL: https://www.debian.org/devel/developers.loc (visited on 11/09/2015) (cit. on p. 6).

Diffie, Whitfield and Martin E Hellman (1976). "New directions in cryptography". In: *Information Theory, IEEE Transactions on* 22.6, pp. 644–654 (cit. on p. 3).

Durumeric, Zakir et al. (2013). "Analysis of the HTTPS certificate ecosystem". In: *Proceedings of the 2013 conference on Internet measurement conference*. ACM, pp. 291–304 (cit. on p. 3).

Eeds, Maryann and Ralph Peterson (1991). "Teacher as Curator: Learning to Talk about Literature". In: *The Reading Teacher* 45.2, pp. 118–126. ISSN: 00340561. URL: http://www.jstor.org/stable/20200822 (cit. on p. 10).

ElGamal, Taher (1985). "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". In: *Adv. in Cryptology, Springer-Verlag*. URL: http://link.springer.com/chapter/10.1007/3-540-39568-7_2 (visited on 02/02/2016) (cit. on p. 4).

Fernández-Sanguino, Javier et al. (2015). *A brief history of Debian*. URL: https://www.debian.org/doc/manuals/project-history/ (visited on 01/13/2016) (cit. on p. 6).

FIPS, PUB (1994–2013). "186. digital signature standard (DSS)". In: *National Institute of Standards and Technology (NIST)*. URL: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf (visited on 02/02/2016) (cit. on pp. 3–4).

González de Requena Redondo, Fernando (2010). "Los espacios de la comunidad: un estudio etnográfico preliminar sobre el Proyecto Debian". PhD thesis. Departamento de Antropología Social y Cultural, UNED. URL: http://

e-spacio.uned.es/fez/eserv.php?pid=bibliuned:500405&dsID=espacioscomunidad.pdf (visited on 01/13/2016) (cit. on p. 6).

González-Barahona, Jesús M et al. (2001). "Counting potatoes: the size of Debian 2.2". In: *Upgrade Magazine* 2.6, pp. 60–66. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.4.8320&rep=rep1&type=pdf (visited on 01/13/2016) (cit. on p. 6).

Housley, Russell et al. (1999). *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. Tech. rep. RFC 2459. Internet Engineering Task Force. URL: https://tools.ietf.org/html/rfc2459 (cit. on p. 3).

Koblitz, Neal (1987). "Elliptic curve cryptosystems". In: *Mathematics of computation* 48.177, pp. 203–209 (cit. on p. 3).

Lundqvist, Andreas and D. Rodic (2006–2012). *GNU/Linux Distribution Timeline*. URL: http://futurist.se/gldt/wp-content/uploads/12.10/gldt1210.png (visited on 01/13/2016) (cit. on p. 7).

McDowell, Jonathan (2009). *Notes from keyring-maint; end of the world not predicted*. URL: https://lists.debian.org/debian-devel-announce/2009/05/msg00005.html (visited on 01/13/2016) (cit. on p. 11).

Merkle, Ralph C (1978). "Secure communications over insecure channels". In: *Communications of the ACM* 21.4, pp. 294–299 (cit. on p. 3).

Miller, Victor (1986). "Use of elliptic curves in cryptography". In: *Advances in Cryptology—CRYPTO'85 Proceedings*. Springer, pp. 417–426 (cit. on p. 3).

Monga, Mattia (2004). "From Bazaar to Kibbutz: How freedom deals with coherence in the Debian project". In: *Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engeneering*, pp. 71–75. URL: http://homes.di.unimi.it/~monga/lib/oss-icse04.pdf (visited on 01/13/2016) (cit. on p. 6).

Mutton, Paul (2016). *US military still SHAckled to outdated DoD PKI infrastructure*. Netcraft. URL: http://news.netcraft.com/archives/2016/01/08/us-military-still-shackled-to-outdated-dod-pki-infrastructure.html (cit. on p. 5).

Nauman, Bruce and Janet Kraynak (2005). *Please pay attention please: Bruce Nauman's words: writings and interviews*. Mit Press (cit. on p. 9).

Network Associates, Inc. (1990–1999). "Validity and trust". In: *An introduction to cryptography*. Chap. 1.1, pp. 29–33. URL: ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf (cit. on p. 3).

Penning, Henk P. (2006–2016). *Analysis of the strong set in the PGP web of trust*. URL: http://pgp.cs.uu.nl/plot/ (visited on 01/13/2016) (cit. on p. 5).

Rivest, Ronald L, Adi Shamir, and Len Adleman (1978). "A method for obtaining digital signatures and public-key cryptosystems". In: *Communications of the ACM* 21.2, pp. 120–126 (cit. on pp. 3–4).

Siobhán O'Mahony, Fabrizio Ferraro an (2012). "Managing the Boundaries of an "Open" Project". In: *The Emergence of Organizations and Markets*. Ed. by John F. Padgett and Walter W. Powell. Princeton University Press. Chap. 18, pp. 545–565. URL: http://www.umass.edu/digitalcenter/

events/pdfs/OMahony_open_project.pdf (visited on 01/13/2016) (cit. on p. 6).

Smart, Nigel (2012). *ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012)*. Tech. rep. 7th Framework Programme, European Commission. URL: http://www.ecrypt.eu.org/ecrypt2/documents/D.SPA.20.pdf (visited on 01/14/2016) (cit. on p. 4).

Wolf, Gunnar (2016). *Learning from our keyring: What do our PGP keys say about the project?* Debian Project. URL: https://debconf16.debconf.org/talks/22/ (cit. on p. 14).

Yee, Peter E. (2013). *Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. Tech. rep. RFC 6818. Internet Engineering Task Force. URL: https://tools.ietf.org/html/rfc6818 (cit. on p. 3).

Zimmerman, Philip R. (1991). *Why I Wrote PGP*. URL: https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html (cit. on pp. 3–4).

Zini, Enrico (2013–2016). *Debian Contriibutors list*. URL: https://contributors.debian.org/contributors/flat (visited on 01/13/2016) (cit. on p. 6).