Journal of Internet Services and Applications

RESEARCH Open Access

Insights on the large-scale deployment of a curated Web-of-Trust: the Debian project's cryptographic keyring

Gunnar Wolf[1]* and Víctor González Quiroga[2]

Abstract

The Debian project is one of the largest free software undertakings worldwide. It is geographically distributed, and participation in the project is done on a voluntary basis, without a single formal employee or directly funded person. As we will explain, due to the nature of the project, its authentication needs are very strict - User/password schemes are way surpassed, and centralized trust management schemes such as PKI are not compatible with its distributed and flat organization; fully decentralized schemes such as the OpenPGP Web of Trust are insufficient by themselves. The Debian project has solved this need by using what we termed a "curated Web of Trust".
We will explain some lessons learned from a massive key migration process that was triggered in 2014. We will present the social insight we have found from examining the relationships expressed as signatures in this curated Web of Trust, as well as a statistical study and forecast on aging, refreshment and survival of project participants stemming from an analysis on their key's activity within the keyring.

Keywords: Trust management, Cryptography, Keyring, Survival, Aging, Curated web of trust

1 Introduction

The Debian project is among the most veteran surviving free software projects; having been founded in August 1993 by Ian Murdock [1], it has grown to be one of the most popular Linux distributions by itself, as well as the technical base for literally hundreds of others. It is the only distribution that produces an integrated operating system capable of running on different operating system kernels - Although an overwhelming majority of Debian users use Linux, it has been *ported* to the FreeBSD and GNU HURD kernels as well [2–4].

But besides all of its technical characteristics, what makes Debian really stand out as a project is its social composition: it is, since its inception, a completely volunteer-driven, community-run project, with very big geographic dispersion [5–7]. Participants in the project have a shared set of emergent cultural values, some of which have been extensively documented [8].

Since Debian's early days, cryptographically strong identification was deemed necessary to guarantee the security guarantees Debian's users have; as the project grew, a viable trust management strategy had to be envised as well; we call it the *curated Web-of-Trust* model [9].

But cryptographic parameters that were deemed safe for long-term use in the mid nineties are now considered to be unsafe. By 2014, the Debian project underwent a large key migration to keep up with the security recommendations for the following years [10]. We described the full reasoning for this migration and an overview of the process and its numeric impact in the project in [9].

The aforementioned migration prompted a study of the direct metrics of the keyring's health, such as those detailed by [11], as well as a more transdisciplinary analysis of the keyring as a social network.

Throughout this work, we will present an overview of the *trust aging* that had started manifesting since around 2010, as well as its forceful re-convergence, and a statistical analysis on key survival expectations.

*Correspondence: gwolf@gwolf.org
[1]Instituto de Investigaciones Económicas, Universidad Nacional Autónoma de México, Mexico City, Mexico
Full list of author information is available at the end of the article

2 Background

In this section, we present background information needed for better understanding of what *prompted* our work, the process underwent by the Debian project that prompted for a fuller analysis to gain understanding of the keyring itself.

Throughout this section, a set of *Research Questions* (*RQs*) are presented, which guide the discussion that follows.

2.1 Trust models in public key cryptography

Besides encryption and signing, public key cryptography provides several models for identity assessment, called *trust models*. The most widespread model is the *Public Key Infrastructure* (PKI) model, a hierarchical model based on predetermined *roots of trust* and strictly vertical relationships (certificates) from *Certification Authorities* (CAs) to individuals. This model is mostly known for being the basis for the `ssl` and `tls` protocols, providing among others secure communication between Web browsers and servers using the `https` protocol.

As we have presented [9], the Debian project, being geographically distributed and with no organizational hierarchy, bases its trust management upon the *Web of Trust* (WoT) model, with an extra step we have termed *curatorship*. The WoT model has been an integral part of OpenPGP since its inception [12]. For this model, there is no formal distinction between nodes in the trust network: all nodes can both receive and generate certificates (or, as they are rather called in the WoT model, *signatures*) to and from any other node, and trust is established between any two nodes that need to assert it by following a *trust path* that hopefully links them in the desired direction and within the defined tolerable distance [11]. This leads to the first research question this work attempts to answer:

*RQ*1 Being Debian such a long-lived project, how does its trust model endure time? Does aging qualitatively challenge it?

Beside the aforementioned work, several other works have studied the information that can be gathered from the total keyring in the SKS keyserver network[1] [13]. The work we will present in this paper is restricted to a small subset thereof - As of December 2016, the SKS network holds over 4 million keys, while the active Debian keyrings hold only around 1500.

2.2 Cryptographic strength

Public key cryptography works by finding related values (typically, very large prime numbers). The relation between said numbers, thanks to mathematical problems that are *hard enough* to solve to be unfeasible to be attacked by brute force, translates to the strength of the value pair.

Over the years since the public invention and publication[2] of public key cryptography [14], several algorithms for finding and relating said numbers have been incorporated into the Digital Signature Standard [15]; currently, the most widely used are RSA (based on the *integer factorization problem*; [16]) and DSA (based on the *discrete logarithm* problem; [17]).

Said schemes' strength is directly related to the size of the numbers they build on. Back in the 1990s, when Internet connectivity boomed and they first became widely used [12], key sizes of 384 through 1024 bits were deemed enough; using longer keys demanded computing resources beyond what was practical at the time.

Of course, computers become more powerful constantly; cryptographic problems that were practically unsolvable 10 or 20 years ago are now within the reach of even small organizations ([10], p. 11). Cryptographic keys used for RSA and DSA algorithms should now be at least 2048 bits, with 4096 becoming the norm.

By 2009 (when the need to migrate to stronger keys was first widely discussed within the Debian project) the amount of 1024-bit keys was close to 90% of the total keyring; the upcoming need of migration was repeatedly discussed, and due to the threat of an attack becoming feasible for a medium-sized organization ([10], pp. 30,32), by July 2014 a hard cutoff line for expiring keys shorter than 2048 bits was set for January 2015, setting a six month period for key migration. We published a analysis on that migration process [9], which prompted the present work.

2.2.1 *Cryptographic certificates in the Debian project*

Not many free software projects started in the 1990s are still active today, but those that are tend to be very large and important. One such case is Debian; as mentioned in Section 1, the project was founded in 1993. Although the vast majority of its developers did not join until many years later, as we will explain in Section 5, many developers have been active for over a decade.

Being Debian a globally distributed project, where any project member is trusted to perform unsupervised uploads that will ultimately be installed and executed in millions of computers worldwide, the needed level of trust in a member's identity clearly surpasses what the traditional username-password pair offers; Debian Developers have used the cryptographic signature as their means of authentication to project services since its early days ([18], pp. 18–20).

Even more, *key signing parties* (KSPs, sessions where each participant verifies the other participants' identity, to later produce a cryptographic certificate or *signature* of the identity, thus strengthening the WoT, further studied in Section 3) ([18], p. 11) have been a long-standing tradition and are acknowledged as a social ties building event at developer conferences and gatherings.

Exchanging key signatures can be a challenging event for newcomers to a community, as can be seen following the exploration and proposal by [19]. Even within a community as tech-savvy as Debian is, we feel it important to understand how useful and how effective KSPs are. Thus the following research question:

*RQ*2 What is the actual effectivity of KSPs for Debian? Are they worth fostering and keeping, or should an alternative trust-building model be sought?

A long-time, socially active developer's key can often be signed by hundreds of people, and the more signing activity a given key has, the more *central* it becomes to the WoT (it becomes a *trust hub*).

While key migration pace did see a strong increase past July 2014, full project participation was effectively cut for 252 developers - that is, about a fourth of the project. Two and a half years later, there were still 167 keys marked as removed that have not been acted upon. We analyze this process at [9]; for the present work, suffice it to say that analyzing this migration process was instrumental in the analysis to be presented.

Our hypothesis is that, even considering the global dispersion of the project, the removed keys mostly belong to people who had already *drifted away* from their project engagement and were inactive; the upcoming Section 4 discusses how this can be understood (and even predicted) from the WoT, even analyzing it years before the migration took place; social practice in Debian makes it hard to determine when a developer is no longer active; although there is a formal process for following up seemingly-inactive developers, [20, 21] given the high amount of human work it requires, it has so far not reached enough coverage.

A process enhancement, automating a good part of the needed follow-up and providing a simple interface for inactive developers to signal they are effectively inactive, has been recently enabled [22]; this change is too recent to be accurately reported, but during the first month after its implementation, it has led to 20 developers to acknowledge they are no longer active in the project. Sixteen of them had 1024-bit keys, which means they had been inactive in most substantive project activities[3] for at least two and a half years already.

This process brings up yet another question: Given that both due to challenges brought up by advances regarding cryptographic strength, and by shifts in priorities or time availability in the lives of the members of the project will most likely continue to create fluctuations in each person's interactions with the project, can anything be learnt from past behaviour to help it cope with future fluctuations? Hence,

*RQ*3 From the data gathered, processed and presented as part of this work, what insights on future behaviour of the keyrings be found via statistical means?

2.3 Threats to validity

This article is based exclusively on the Debian Curated Web of Trust, it does not relate to or cover any other project's keyring. This is mainly because, to the best of our knowledge, *there just is no other project* which implements a CWoT in a similar fashion. As we explain in Section 2.2.1, the practice of exchanging key signatures is strongest in the Debian project, it does exist in other free software communities, but not with the same strength exhibited in Debian. Even just by sheer size, the footprint of `@debian.org` mail addresses in the SKS network is larger than most countries [13].

As for other groups that could be comparable, we could find the image of a community in several free software projects (such as Tor, Fedora, OpenBSD). However, said communities do not use a keyring as an integral part of their infrastructure. That is, there is no curation process to them, and access is not granted based on whether an individual presents a key that belongs to a given keyring.

It should be noted, the authors have started talking with a well recognized free software development project, which will possibly adopt curation and privilege-granting processes similar to Debian's. We do not want to commit them, so we have chosen not to name them.

We have been approached with questions regarding the analysis of the *keyring blobs* described in Section 4. Graphically interpreting a graph such as the ones prompting this study (Fig. 3) might not be meaningful; the shape of the *blob* itself could be an artifact of specific nodes ordering. In order to address this question, we tried reversing and randomizing the nodes in the *graphviz* source files, and found our observations to be sustained. We also switched the rendering engine to the JavaScript-based `visjs`, and found it to be stable. However, the analysis is still ocular; we have not performed any numerical analysis that can confirm our hypothesis.

As for Fig. 4, a similar question arises when considering overplotting: Are the colors we see a faithful representation, or is there hidden information underneath? Even more, is the color choice correct? As we mention in Section 4, some colors are more visible than others. For this question, we also compared the resulting plots to plots done with the edges presented in different order and with different colors; the results are coherent with what we present.

3 Measuring key signing parties (KSPs)

Given we are already using the developers' keyring to measure social engagement, connectedness and activity of individuals, it makes sense to study the KSPs. We will thus proceed to compare the size, progression and reach of the KSPs held at the Debian yearly developer's conference, *DebConf* (DC).

Before starting with this session's analysis, we must point out this section was written not only based on properties obtained from the data set, but with personal knowledge of one of the authors having been a participant and organizer of DebConf for most of its editions; we explain some observed trends based on insights of the Debian project community that cannot be supported by formal reference material.

Of course, KSPs are not only held at DC; a long-standing tradition is for developers to announce in the *debian-private* mailing list their travel plans, and it is customary for them to explicitly mention meeting their peers to exchange key signatures. There are also formal KSPs at different free software meetings. But for the purposes of our study, we decided to focus on the largest and more representative events.

We have analyzable data for the past twelve DC editions. Attendance to DC (and participation in its KSPs) varies by several factors, mainly the geographical location and the world (as well as each country's) economic conditions, but we will present some hypotheses as to some observed trends. The number of participants per each DC edition, as well as the proportion of its participants who were part of the KSP is shown in Fig. 1.

Part of the attendee and KSP participation data are quite natural, and are well known and discussed within the Debian community: throughout the years, the conferences with highest attendance are those held in locations closest to large concentrations of Debian developers - Table 1 shows the location where each of the DC conferences has been held since 2006, as well as the absolute numbers used for Fig. 1, sorted by its number of attendees.

We say this distribution is natural because the most attended DCs (15, 17 and 7) were held close to important developer density population centers, and the lowest

Table 1 DebConf edition, location, number of attendees, number of KSP participants, and number DDs in the KSP for each DebConf since 2006, sorted by its number of attendees

DC	Location	Attendees	KSP	DDs
15	Heidelberg, Germany	581	282	145
17	Montreal, Canada	405	155	104
7	Edinburgh, Scotland	394	179	63
11	Banja Luka, Bosnia	335	139	91
14	Portland, United States	314	123	86
10	New York, United States	301	161	105
13	Vaumarcus, Switzerland	290	139	72
16	Cape Town, South Africa	282	130	85
6	Oaxtepec, Mexico	259	154	69
9	Cáceres, Spain	246	179	62
8	Mar del Plata, Argentina	223	76	29
12	Managua, Nicaragua	161	65	39

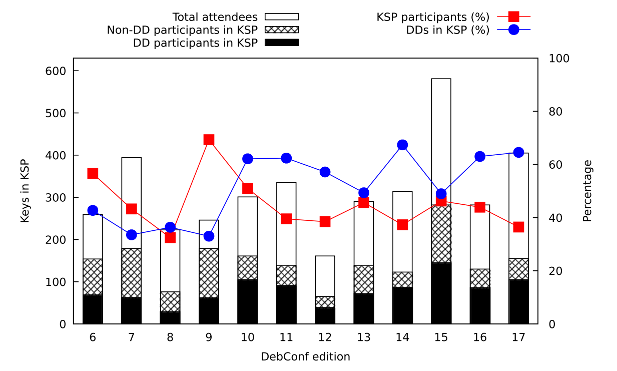

Fig. 1 Number of participants per each DebConf, showing the proportion (in black, as well as in the label) of those who were part of the key signing party

attendances (12, 8) were held in countries further away. Given the developer density in the United States, attendance for DC 10 and 14 rate seems low; this might be caused by several developers not willing travel there due to their ideological positions, or unable to do so, due to their national origin. We found it surprising to find only 246 people attended DC 9, held in Western Spain, in a region that between 2002 and 2013 pioneered free software development [23]; its number, however, is close enough to the following most attended, 6 and 16, to require further information to explain their relative sizes - Issues such as economic conditions on the relevant year ([24], p. 26) or even perceived ease to travel to the destination.

Not every attendee to the conference takes part in the KSP. Looking at the proportion of attendees who signed up to participate shows also an interesting picture. Only three times KSP participation has had over half of the attendees (DC 9, with 72.76%, 6, with 59.46%, and 10, with 53.49%). Other than those three occurrences, participation has remained in a fairly narrow 15% band, between 34 and 49%.

Not all participants in DC, nor in the KSP, are formal members of the Debian project (Debian Developers). Figure 1 captures this as well. We find some interesting patterns when looking at both relations; the match is not perfect: we counted each key as belonging to a DD if it had an identity with an *@debian.org* mail address. Some (few) DDs are known not to add this identity to their key material.

DC 9 was quite outstanding as it is by far the conference with least non-KSP participants, although it keeps a high proportion of non-DD participants. This can relate to the economic recession mentioned in ([24], p. 26): while fewer Debian Developers attended the conference than usual, it was held in a region with high free software involvement; looking at the list of participants, it contains a high number of local people affiliated with different Free Software projects.

DC 7, 8 and 9 have the lowest percentage of DDs in the KSP. This can be caused by the aftereffects of the *Transnational Republic experiment* carried out in DC 6: a DD, well known and well connected to the WoT, presented a seemingly official ID document generated by a fictional non-government, the Transnational Republic experiment [25, 26]. This incident led to an important discussion within the project and a change in the way KSPs were held after said conference. As this clearly showed, this had the result of socially diluting the responsibility of verifying each other (that is, if everybody has checked somebody's identity, each participant does only a very cursory look on the documents. Given KSPs were already over 150 participants long, expecting thoroughness was clearly unrealistic.

Now, how significant are KSPs to the connectedness of their participants? Not everybody in a keyring cross-signs during a KSP. In fact, until the aforementioned incident in DC 6, the KSP *protocol* in Debian used to be for every participant to form two lines, having ID documents ready, and cross-check every other participant's identity.

From DC7 onwards, keysigning sessions are held in a *continuous KSP* fashion: after an initial session where a document stating the key for the full document with the keys of every participant is verified to be correct and the same for every KSP participant, attendees are encouraged to meet and have a relaxed talk and introduction with each other. Of course, the amount of cross-signatures every person will get is much smaller than with the two-line model, but they are also of greater significance.

After DC 9, the keyring maintenance team started raising awareness of the need to migrate to stronger keys, as explained in Section 2.2.1 and detailed at length in [9]; we infer that raised the need to reconnect newer, stronger keys into the keyring, as the high DD participation in KSPs starting at DC 10 shows.

Considering this, how effective are KSPs to build trust in keyrings? To this effect, we measured the ratio between total signatures and keys on each KSP's keyring (that is, the average percentage of the keyring each key is directly connected to), at the date of the KSP session itself, and weekly for the following 15 weeks; this can be seen in Fig. 2.

KSP effects are not immediate; participants are encouraged to print out a copy of the base document at home, from a computer system they ultimately trust; the first thing shown by Fig. 2 is how long does it usually take to KSP participants to sign, send, receive and upload the keys: while there is a big variation in the initial three weeks, changes quickly converge and change past said period is very minor.

As for the total connectedness for each of the KSP keyrings, the inner increment is quite sensible for all cases during the observed period; DC 7 shows the least

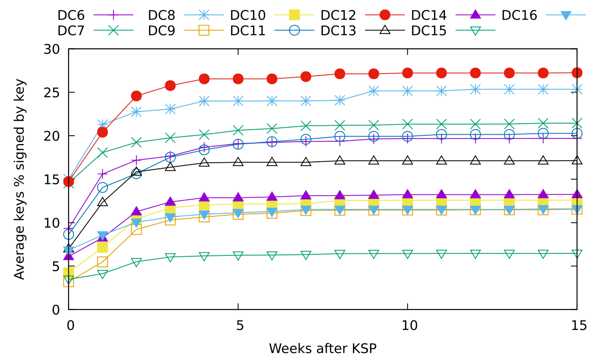

Fig. 2 Average percentage of KSP directly connected to each key by KSP at the KSP date, and weekly for the following 15 weeks

improvement, increasing the average percentage of keys signed directly from each key from 14.55 to 21.45 (a 47% increase). A hypothesis towards such a small increase could be that DC 7 was held in Scotland. The Debian-UK community, predominantly centered in Cambridge, is very large and tightly knit socially from the very early days of the project. The KSP started off with a very well connected keyring, and although this has been one of the largest conferences in the project history and the total participants in the KSP is the second highest, being this the first KSP after the *Transnational Republic experiment* mentioned above, it follows that many people would be reluctant to sign keys from people they weren't already familiar with.

Two years later, the KSP had the biggest increase: DC 9's keyring went from 3.20 to 11.55, a 261% increase. Do note that, although it was the smallest conference in Europe, Table 1 shows the absolute numbers for the KSP were coincidentally equal to DC 7's, and the proportion of DDs was almost equal. We think this might be explained because many of the DDs who attended DC 9 were *not* strongly connected to the project, in contrast with the mentioned situation of Debian-UK.

Table 2 compares the starting and ending points shown in Fig. 2. We find it striking that DC editions that would suggest very different settings are so close together - DC 7 and 8 are close to each others' antithesis; second largest and second smallest KSPs, DC7 in one of the DD-densest countries and DC8 in a very DD-sparse region; so are DC12 and DC15.

As mentioned at the beginning of the present Section, most of the explanations of phenomena depending on social interaction are just hypotheses; we have to acknowledge several data are within the range for uncertainty to play a heavy hand.

4 Trust aging and reestablishment

The work done for the described keyring migration, as well as the migration process itself, presented a great

Table 2 Increase of the average number of signatures 16 weeks after a KSP

DC	Partic.	DDs (%)	Begin	End	Increase	%
7	179	102 (57)	14.55	21.45	6.90	47
8	76	47 (62)	14.99	25.34	10.35	69
16	130	86 (66)	6.82	11.58	4.76	69
12	65	39 (60)	14.72	27.25	12.53	85
15	282	145 (51)	3.48	6.46	2.98	85
6	154	76 (49)	9.34	19.69	10.35	110
14	123	87 (71)	6.12	13.25	7.13	116
11	139	91 (65)	8.64	20.27	11.63	134
13	139	72 (52)	7.00	17.13	10.13	144
10	161	105 (65)	4.22	12.59	8.37	198
9	179	112 (63)	3.20	11.55	8.35	260

Table shows DebConf edition, absolute number of KSP participants, the portion of them that were DDs (as well as the percentage they are of the total), beginning and ending averages of directly connected keys in KSPs, with the proportion of increase seen in that KSP (absolute and percentage). The table is presented sorted by the increase percentage

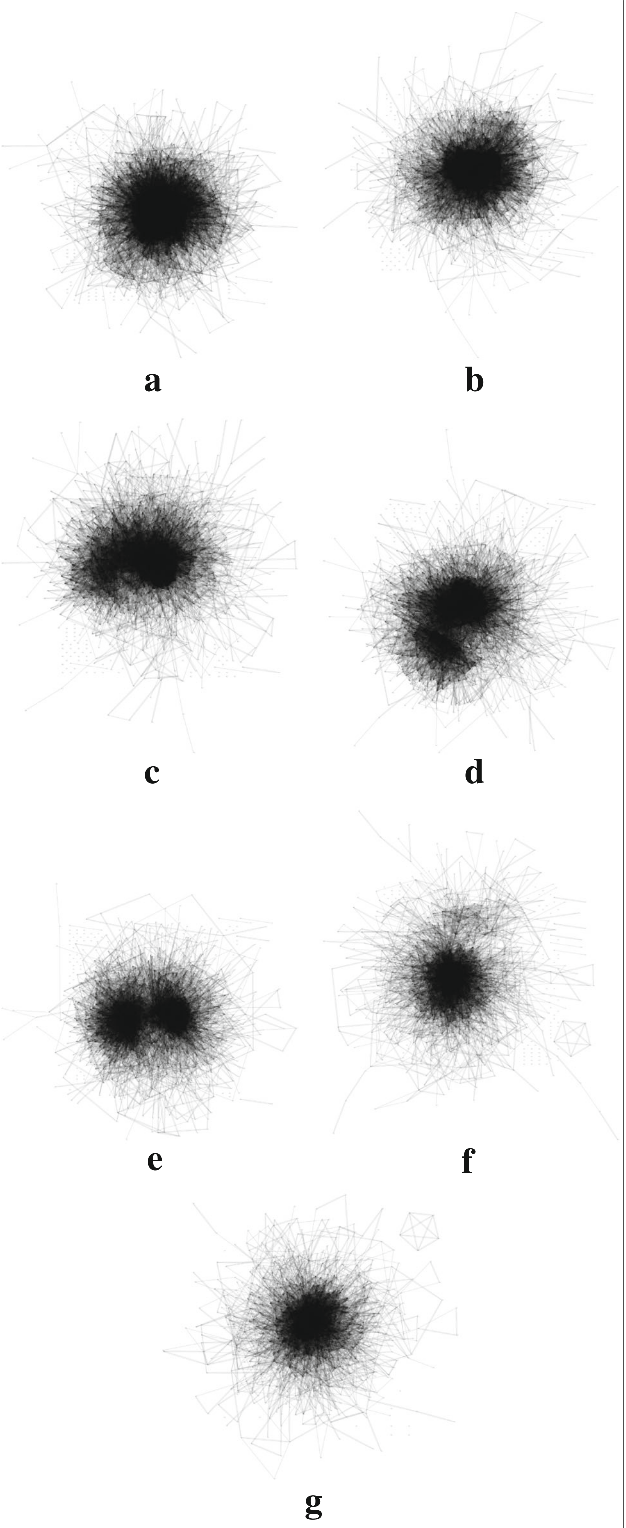

Fig. 3 Snapshots of the Debian keyring evolution at different points in time. **a** January 2009 **b** January 2010 **c** January 2011 **d** January 2012 **e** January 2014 **f** December 2014 **g** January 2015

opportunity to understand the key migration as a social phenomenon as well, using the keyring as a way to measure social cohesion and vitality.

We prepared graphic representations of the keyring at its various points in time, in the hope to learn from it patterns about its growth and evolution that can warn about future issues. For the trust-mapping graphs, we use directed graphs, where each key is represented by a node and each signature by an edge from the signer to the signee. For starters, we were interested in asserting whether the characteristics observed on the whole OpenPGP WoT [11] repeated in the subset of it represented by the Debian keyrings. Of course, said work was done as a static analysis on the keyring back in 2011; back then, the whole OpenPGP keyring stood at 2.7 million keys; at the time of this writing there are 4.5 million keys, growing by 100 to 400 keys every day [27].

Figure 3 presents seven snapshots of the main developers keyring, processed by *Graphviz* using the *neato* layout program, which implements the *spring* minimal energy model [28]. Of course, at the scale they are presented, each individual edge or node becomes irrelevant; there is too much density at the center, and the outlying nodes and edges appear as just noise. However, the *shape* of the strong set[4] does lend itself to analysis.

Figure 3a, b and g present a regular shape, approximately following Ulrich's observations, that the strong set of the WoT exhibits scale-freeness. Quoting ([11], Section 4.3),

> Connectivity-wise, scale-free graphs are said to be robust against random removal of nodes, and vulnerable against the targeted removal of hubs (which

leads to partitioning). This is usually explained by the hubs being the nodes that are primarily responsible for maintaining overall connectivity.

Ulrich notes that the WoT graph is *similar to a scale-free one and exhibits a hub structure, but is not scale-free in the strict sense.*

Something happened, however, in the course of 2010 that led to the WoT acquiring the shape shown in Fig. 3c by the end of the year - Instead of a seemingly uniform *blob*, there is a distinct protuberance. This horn grew throughout the following years, and by 2014, the keyring consisted of two roughly equivalent *blobs* somewhat weakly linked together, as Fig. 3d and e show.

We find this protuberance to be the portrait of a social migration: The project is often portrayed as unique among free software projects due to the close personal ties among developers; its yearly developers' conference, *DebConf*, has a very high repeating attendance rate. However, given the project has lived for over 20 years, it is understandable many of the original members have grown inactive and moved on to other life endeavors; formal retirement is requested from developers, but many people reduce their engagement gradually, and just never formally retire.

While the geographical dispersion makes it quite hard for some developers to meet others and obtain new certificates, as we already mentioned there is a tradition in Debian to announce travels in a (private, developers-only) mailing list, and active developers often will gladly meet people traveling to their region just for a key signature exchange.

Although the number of developers that by late 2010 had migrated to a stronger key was still quite small, the call for key migration was initially answered by those with most active key activity -hence, probably more conscious about the importance of this migration. Of course, although it was not a targeted removal, it was a socially self-selected one: trust hubs were among the first to migrate to stronger keys. And even though they attempted to re-build their WoT relationships and cross-sign with other developers at gatherings such as DebConf, the group of developers that -as explained in Section 4- had drifted away from project activity didn't reconnect with them.

While the migration to keys longer than 1024 bits took much longer than originally expected, the initial push was not bad: during 2010, it reached from practically zero to close to 10% of the keys - But many of those keys were *hubs*, people long involved in the project, with many social bonds, and thus very central keys. When those people migrated to newer keys, the signatures linking their long-known fellow developers to the project were usually not updated, and several old keys could have even *become islands*, gradually losing connectivity to the strong set.

Given Debian's longstanding practices, rather than isolated, many such keys started *drifting apart* as a block, growing separated from the center of mass. This explains why the migration started as a *lump* to later become two large, still somewhat strongly connected bodies, mostly stable over the years. Of course, as more developers migrated to strong keys, by late 2014 the remaining group started losing cohesion, and by December 2014 (before it was completely removed), it is barely noticeable - All of its real *hubs* had migrated to the new center of mass, with many previously connected keys becoming isolated, as Fig. 3f shows.

In order to prove this hypothesis, we generated again the same graphs, but factoring in the *trust aging*: if individual signatures are colored by their age, it is possible to visually identify if a significant portion of the group's trust is aging - That is, if social bonds as reflected by intra-key signatures are over a given edge. The seven subfigures of Fig. 4 correspond with those of Fig. 3, but with color-coded edges (according to the image caption)[5].

Surprisingly, even Fig. 4b shows a clear grouping of keys by signature age - But this grouping does not appear a year earlier, in Fig. 4a. This can, again, be indicative that the first people to migrate to stronger keys, even before it altered the overall shape of the WoT, migrated during 2009; by early 2010, they might constitute the tight, new (blue) group still in the periphery, that eventually became the core of the newer *blob*.

5 Statistical insights on the keyring history

Following from the same data set, we started a further statistical analysis; this section presents the preliminary results we gathered from applying survival analysis techniques.

The general focus of survival analysis is on the modeling the time it takes until a specific event occurs, in social sciences one often speaks of *event history* [29]. We have found interesting findings from studying how many people keeps participating in the Debian project throughout the time, that is, to model the time until departure from the Debian project. The main motivation comes from the need to understand keyring population along time and from the the implications of survival as reliability of subjects (it is more likely for someone to be trusted if they've been long enough in a community), thus arising a rough measure for trust. Our sampled data is defined by the keys that make up the curated WoT from the Debian Developers keyring [9].

The analyzed data is treated as a longitudinal study. We point out that intervals are not of the same length in time: each data point is a *tag* in the keyring's Git history,[6] and the period of analysis spans between July 2008 and July 2016. During said period, 124 tags were recorded, averaging to 23.96 days each, with a standard deviation

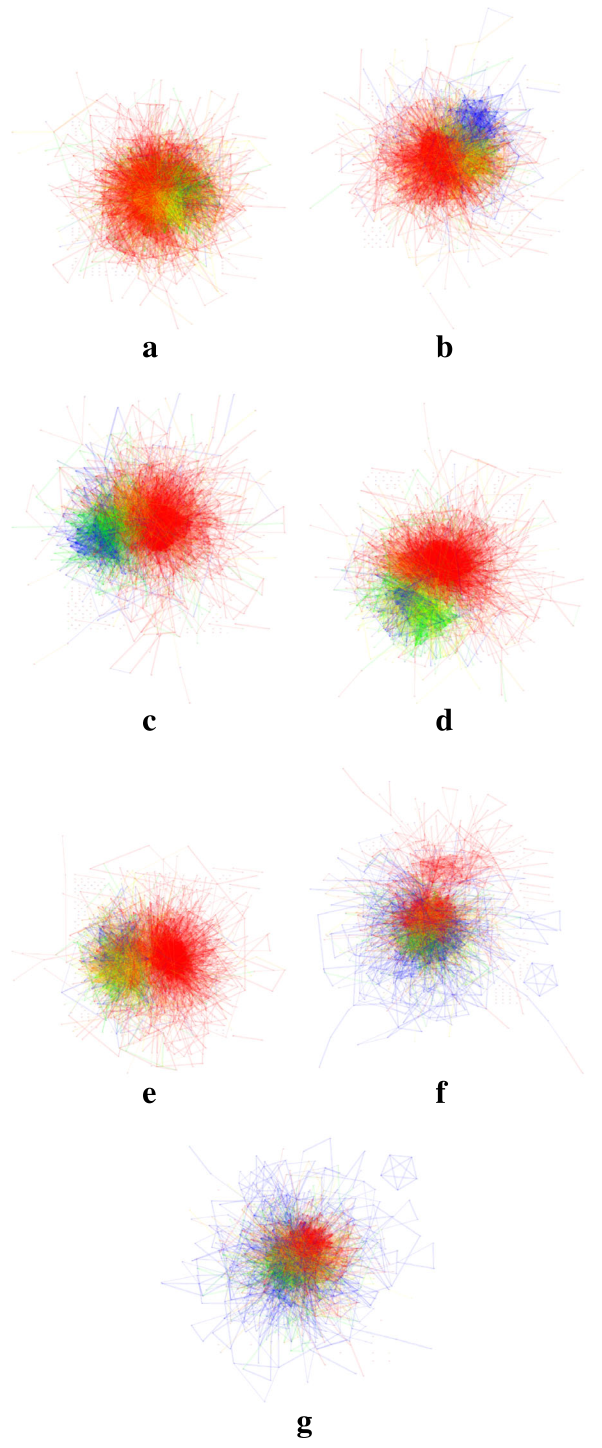

Fig. 4 Snapshots of the Debian keyring evolution at different points in time, showing signature age: blue, ≤ 1 year; green, between 1 and 2 years; yellow, between 2 and 3 years; orange, between 3 and 4 years; red, ≥ 4 years. Signature coloring is relative to each of the snapshots: blue edges in **a** represent signatures made throughout 2008. **a** January 2009 **b** January 2010 **c** January 2011 **d** January 2012 **e** January 2014 **f** December 2014 **g** January 2015

of 27.95, with a maximum of 69 days and a minimum of one day.

Given the way the keyring is structured, we used the *long key ID* (the lowest 64 bits of its fingerprint, in hexadecimal representation) as a unique identifier for each key. For each tag and key we counted the number of signatures made to that key by counting the number of non-zero entries in the corresponding key column of an adjacency matrix at a specified tag.

We identified people's participation in Debian using their key activity record (has the key stopped getting signatures?) and keyring membership (has the key stopped being part of the keyring of interest?) which codes our data as right-censored because no further information about keyring membership is known afterwards; right censoring scheme constitutes data where all that is known is that the individuals are still *alive* (the keys are still active) at a given time, [30].

For this analysis, in order to make a key-to-person correspondence, we considered key ownership using the following equivalence relation: keys are equivalent (i.e. they refer to the same person) if they have the same metadata. We considered that using the name as metadata was naturally appropiate to make a distinction among people.

We used the R programming language mainly leveraging routines from `survival` and `flexsurv` packages; unless otherwise stated, significance level is assumed to be 5%.

Our line of approach begans showing the proportion of remaining people in the keyring along time through the survival function, that is, keyring permanency. Then, using the accumulated hazard function, we get the expected exits per person that remains in the keyring until the end time (in perpetuity). And lastly, using the hazard rate function, we get the departure rate from the keyring.

For the non-parametric or observed curves we used the Kaplan-Meier product limit estimator for the survival function, [31], the Nelson-Aalen moments estimator for the accumulated hazard function [32], and the kernel density estimator for the hazard rate function, [33].

A parametric estimation to see the *mortality law* fitting our data was made using a Generalised Gamma distribution through maximum-likelihood estimation [34]. The motivation for using the Gen. Gamma model is due to the closeness and confidence band coverage it has to the observed hazard rate function obtained non-parametrically. Proper justification for said model comes from the fact that it minimizes Akaike's Information Criterion when compared to the other models, making it a better model in terms of information loss, [35], while also rejecting other models using a log-likelihood test of -1422.395 at 3 degrees of freedom, [36]. The estimated parameters found for our model were $\mu = 2.399$, $\sigma =$

0.2722, $Q = 2.5594$, with standard errors of 0.0719, 0.1432, 1.3519 respectively.

Finally to make inference about the effect of received signatures as a predictor for survival, we used a semi-parametric model using the Cox Proportional Hazards model [37] by taking the average number of signatures received by a person as a covariate. The estimated regressor for the avg. num. of signatures received was $\beta = -0.00839$, with a standard error of 0.0024 and a p-value of 0.0046. Proper verification for the proportional hazard assumption for the avg. num. of signatures received was done testing the Scaled Schoenfeld Residuals [38] against a Schoenfeld Individual Test which yielded a p-value of 0.1617. This means the average number of signatures received by someone has a statistically significant risk contribution at any given time [39].

In the non-parametric plot of Fig. 5, we observe downward steps when at least one people stops getting signatures. The crosses represent the followup time for censored observations (for which no further information is known and thus the proportion of keys remains). This plot does reflect the fact that many keys were dropped during the 1024-bits key removal (circa January 2015). Observed proportion of keys being above the theoretical model from year 3.5 to 6.5 years suggests that after three years the keys wouldn't be much likely to leave; at least not until after 6.5 years, where the probability of remaining afterwards is almost as the 50% chance of heads in a coin flip. It is remarkably that keyring permanency doesn't really go below 50%, showcasing good health in the keyring.

As we mentioned, due to the 1024-bit key migration, there is a clear skew that introduces a sharp drop around 6.5 years.

Figure 6 shows the people exits given one key in perpetual risk, that is, if it is to remain in the keyring for all its time span. The increasing steps from the non-parametric exits is natural being the accumulated sum per tag of the exits over remaining people ratios. The similarity from previous plot is expected since cumulated hazard is a logarithmic transformation from survival function. We see again that the observed plot lies below the theoretical model starting from year 3.5 through year 6.5 (about 3 years), quickly increasing afterwards more than expected. It is not until near year 3.5 that a someone is expected to be half-way to go, which is better when compared to the expected life of a subject being 5 years.

Figure 7 shows the departure rate is analogous to a *mortality* rate. The observed behaviour suggest that coming of age there's a sudden increase on the risk i. e. keys "wear out" to their age around year 5.5, certainly close to the expected life. Yet the parametric departure rate being under the non-parametric rate at the final years shows the dramatic effect from the 1024 removal. Another remarkably finding was that departure rate in general gives empirical evidence to say that 7 out of 100 keys will leave "any time now" (from the fact that hazard rate is the instantaneous probability of failure at a specified time; *failure* means, the probability of a key will completely cease activity after a given time of life) in a 8 year lapse.

From Fig. 8 we see that the average number of signatures received trend has a negative risk contribution. It is noteworthy that coming of age, the risk contribution starts to be positive by a bit, effectively showing the effect of the hazard rate at later ages. The resulting effect was that for each signature received by someone it reduces the baseline risk by 0.83% on average.

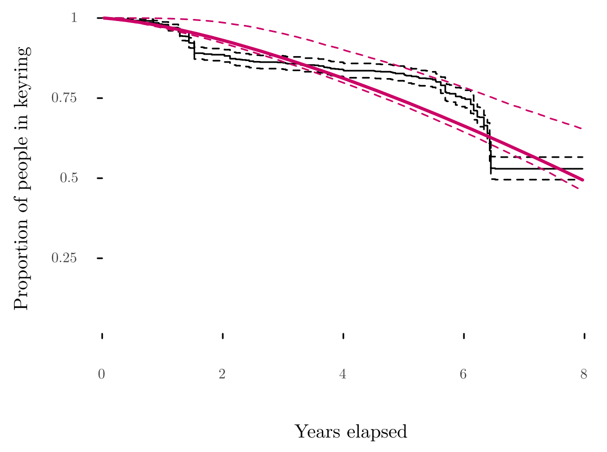

Fig. 5 Probability of people permanency. The black line follows observed (non-parametric) data from the keyring, with crosses representing the followup time for right-censored observations; the red line is the parametric estimation; dotted lines represent confidence bands

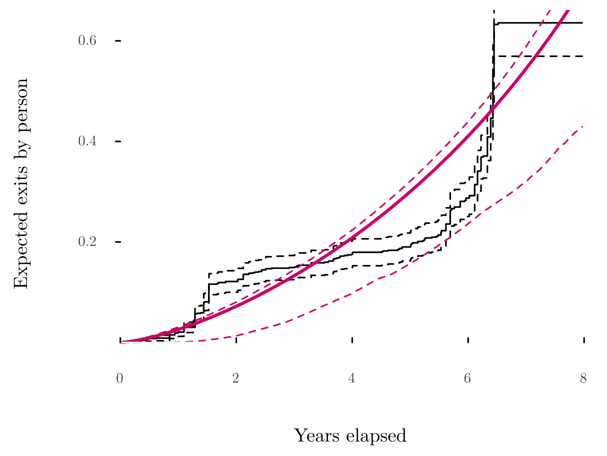

Fig. 6 Cumulated hazard of people exits. The black line follows observed (non-parametric) data from the keyring, with crosses representing the followup time for right-censored observations; the red line is the parametric estimation; dotted lines represent confidence bands

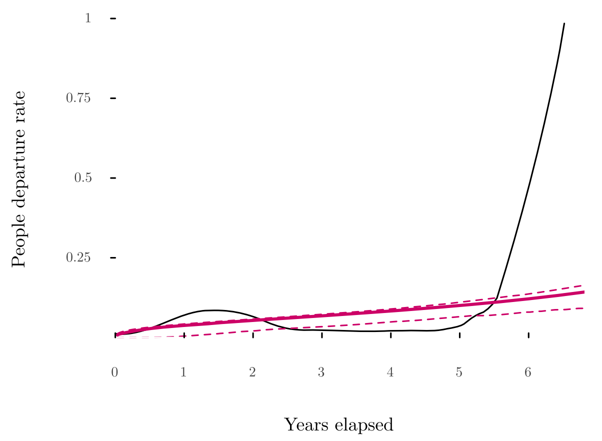

Fig. 7 Hazard rate of people exits. The black line follows observed (non-parametric) data from the keyring, the red line is the parametric estimation; dotted lines represent confidence bands

So how many signatures does someone needs to be almost failure-proof? We found a minimal average number of 358 signatures are required to make up for a failure risk statistically close enough to 0 (provided that the subject holds consistent and not by-chance interaction with the people from said 358 signatures).

As a comparison, the max recorded average of signatures received by a person was of 168.45 yielding an absolute failure risk reduction of about 75.67% from the baseline, and as expected, said person is indeed active; the mean average of signatures received was of 11.82 yielding an absolute failure risk reduction of about 9.45% from the baseline.

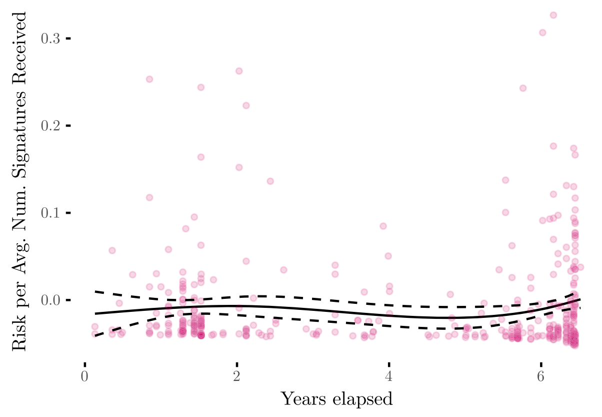

Fig. 8 Scaled Schoenfeld Residuals for the Average Number of Signatures Received. The black line follows the fitted trend for the risk contribution from the avg. signatures received, the red dots are observed risk contributions from the avg. signatures received at a specific time; dotted lines represent confidence bands; colored areas represent higher risk concentrations

In summary receiving 12 signatures (which is roughly the expected average number of signatures received) reduces the odds of exiting at any given time by 10% from the baseline risk. In general people in the project will constantly remain active for about 4.5 years, as long as they went through 1.5 years. It is just after six years where people effectively have an uncertain end unless they interacted meaningfully with other people: holding consistent relationship with at least 30% of people in the keyring can almost guarantee a lifetime membership.

6 Conclusions and future work

The Debian keyring is a very peculiar subset of the whole OpenPGP Web of Trust analyzed in [11]. The work we present here provides data empirically supporting the theoretical observations, particularly regarding the robustness of what he defines as the LSCC (Largest Strongly Connected Component). The migration away from 1024-bit keys provided an opportunity to follow the progression of the connectivity in our WoT after several of its hubs were removed.

The preliminary results for this work have been shared with a group of Debian developers. Historically, the usual practice for key signing has been to generate non-expiring signatures; people that have already cross-signed their keys don't have an incentive to refresh their trust. There is an ongoing discussion as to whether this practice should change towards time-limited signatures, better modeling ongoing social relationships, or to stick to current practice.

The resulting survival analysis can be used to generate an objective measure for trust; this study was done only on the *Debian Developers* keyring, it would be interesting to compare with the more loosely connected *Debian Maintainers* keyring. We also want to further explain the keyrings by stratification. The survival analysis showcases good health of the *Debian Developers* keyring (which makes up the mass of Debian's WoT).

Finally, the methodology followed for this study could be applied to other free software projects, aiming to correlate with events and trends spanning a wider population than Debian's; the applicability of our work to other projects, however, depends on having a proper data set to base the work off. As mentioned in Section 2.3, we are aware of only one large project interested in formally structuring a Curated Web-of-Trust keyring, but other data sets could be taken as inputs - several authors have performed studies based on the pattern of discussions in mailing lists [40]; most of the analysis we presented here is based on the observation of seven years of history, so it will take a long time before this can be applied over other data sets.

Endnotes

[1] For a WoT model to be able to scale beyond a small number of participants, *key servers* (systems that store and allow for retrieval of public key material) are needed. The *Synchronizing Key Server* (SKS) network is the largest network of OpenPGP key servers.

[2] While it is now known that public key cryptography was invented by the UK Government Communications Headquarters (GCHQ) in the early 1970s, it was kept as classified for over 25 years, which resulted in [14] being widely credited for its invention and publication.

[3] An active key is required for most regular activities a developer performs in Debian - Most notably, for uploading packages and for voting in general resolutions.

[4] The strong set is defined as the largest set of keys such that for any two keys in the set, there is a path from one to the other [41].

[5] Some care should be taken interpreting the presented graphs. Particularly, chosen colors are not equally strong and visible against white background; mid-range (orange, yellow) signatures appear weaker than red or blue ones. Also, the drawing algorithm overlays lines, and in high density areas, only the top ones prevail. Still, we believe our observations to hold despite these caveats.

[6] Version control systems handle the concept of *tags* in a repository: specific points of a project's development that are in some way relevant or significant; many projects use *tags* to mark their releases. This is the case of the Debian keyring maintainers' repository: tags mark each keyring version that was put in production. The team attempts to put a new version in production roughly once a month.

Acknowledgements
Jonathan McDowell and Daniel Kahn Gillmor, members of the Debian keyring maintenance team, for their input and explanations on the OpenPGP key format, specific semantics on signatures, and expert input on the direction of this research. GPLHost, for donating the computing resources needed for this work.

Funding
Not applicable.

Availability of data and materials
This study is based on data publicly available at the Debian keyring maintenance team's Git repository, available at https://anonscm.debian.org/git/keyring/keyring.git.
The scripts used to analyze the keyring are available at https://krstat.debian.net/acad/scripts.
Section 3 presents information from the *keysigning parties*, downloaded from their respective coordination pages, archived at https://people.debian.org/~anibal/.

Authors' contributions
VGQ is responsible for the analysis and subsequent explanation of Section 5. GW is responsible for the rest of the article. All authors read and approved the final manuscript.

Competing interests
Gunnar Wolf is a Debian Developer and part of the Debian keyring maintenance team.

Publisher's Note

Author details
[1]Instituto de Investigaciones Económicas, Universidad Nacional Autónoma de México, Mexico City, Mexico. [2]Facultad de Ciencias, Universidad Nacional Autónoma de México, Mexico City, Mexico.

Received: 23 October 2017 Accepted: 25 March 2018
Published online: 16 May 2018

References

1. Fernández-Sanguinoa J, et al. A Brief History of Debian1997–2015. https://www.debian.org/doc/manuals/project-history/. Accessed 22 Dec 2016.
2. Monga M. From Bazaar to Kibbutz: How freedom deals with coherence in the Debian project. In: Feller J, et al, editors. Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engeneering. Edinburgh: Association for Computing Machinery (ACM); 2004. p. 71–5. https://doi.org/10.1049/ic:20040268. http://homes.di.unimi.it/monga/lib/oss-icse04.pdf.
3. SPI, et al. Debian GNU/kFreeBSD1997–2016. https://www.debian.org/ports/kfreebsd-gnu/. Accessed 22 Dec 2016.
4. SPI, et al. Debian GNU/HURD1997–2016. https://www.debian.org/ports/kfreebsdgnu/. Accessed 22 Dec 2016.
5. Robles G, Gonzalez-Barahona JM, Michlmayr M. Evolution of volunteer participation in libre software projects: evidence from Debian. In: Proceedings of the 1st, International Conference on Open Source Systems. 2005. p. 100–7.
6. Robles G, Dueñas S, Gonzalez-Barahona JM, et al. In: Feller J, editor. Corporate Involvement of Libre Software: Study of Presence in Debian Code over Time. Boston: Springer US; 2007, pp. 121–32. https://doi.org/10.1007/978-0-387-72486-7_10. http://dx.doi.org/10.1007/978-0-387-72486-7_10.
7. Mateos-Garcia J, Steinmueller WE. The institutions of open source software: Examining the Debian community. In: Information Economics and Policy 20.4. *Empirical Issues in Open Source Software*; 2008. p. 333–44. https://doi.org/10.1016/j.infoecopol.2008.06.001. http://www.sciencedirect.com/science/article/pii/S0167624508000346.
8. Coleman EG. Coding freedom: The ethics and aesthetics of hacking. 2013. http://gabriellacoleman.org/Coleman-Coding-Freedom.pdf. (Visited on 01/13/2016).
9. Wolf G, Gallegos-García G. Strengthening a CuratedWeb of Trust in a Geographically Distributed Project. 2017. http://www.tandfonline.com/doi/full/10.1080/01611194.2016.1238421.
10. Smart N. ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012). *Tech. rep. 7th Framework Programme*. European Commission; 2012. http://www.ecrypt.eu.org/ecrypt2/documents/D.SPA.20.pdf. Accessed 14 Jan 2016.
11. Ulrich A, et al. Investigating the openPGP Web of Trust. In: Proceedings of the 16th European Conference on Research in Computer Security. ESORICS'11. Leuven: Springer- Verlag; 2011. p. 489–507. ISBN: 78-3-642-23821-5. http://dl.acm.org/citation.cfm?id=2041225.2041260.
12. Zimmerman PR. Why I Wrote PGP. 1991. https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html.
13. Cederlöf J. Dissecting the leaf of trust. 2004. http://www.lysator.liu.se/jc/wotsap/leafoftrust.html.
14. Diffie WE, Hellman ME. New directions in cryptography. Inf Theory IEEE Trans. 1976;22(6):644–54.
15. PUB FIPS. 186. digital signature standard (DSS). In: National Institute of Standards and Technology (NIST) *(1994–2013)*. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf. Accessed 02 Feb 2016.
16. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM. 1978;21(2):120–6.
17. ElGamal T. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. 1985. http://link.springer.com/chapter/10.1007/3-540-39568-7_2. Accessed 02 Feb 2016.
18. Ferraro F, O'Mahony S. Managing the Boundaries of an "Open" Project. In: Padgett JF, Powell WW, editors. The Emergence of Organizations and

Markets. *Chap. 18*. Princeton University Press; 2012. p. 545–65. http://www.umass.edu/digitalcenter/events/pdfs/OMahony_open_project.pdf. Accessed 13 Jan 2016.

19. Bichsel P, et al. Security and trust through electronic social network-based interactions. In: Computational Science and Engineering, 2009. CSE'09. International Conference on. *Vol. 4*. IEEE; 2009. p. 1002–7.
20. Debian Project. Teams/MIA2007–2014. https://wiki.debian.org/Teams/MIA. Accessed 07 Mar 2016.
21. Huber M, Debian MIA. Raiders of the "Lost" Maintainer. 2007. http://www.linux-magazine.com/Online/News/Debian-MIA-Raiders-of-the-Lost-Maintainer.
22. Zini E. Bits from the New Member Process. 2017. https://lists.debian.org/debian-devel-announce/2017/08/msg00009.html. Accessed 12 Sept 2017.
23. Aparicio FF. TIC y sociedad: salvando la brecha digital. El caso de Extremadura: los nuevos centros del conocimiento y el software libre. In: Revista Latinoamericana de, Tecnología Educativa-RELATEC 3.1. 2007. p. 29–44. http://relatec.unex.es/article/view/21.
24. DebConf organization. DebConf9 final report. 2009. https://media.debconf.org/dc9/report/.
25. Srivastava M. Please revoke your signatures from Martin Krafft's keys. 2006. http://lists.debconf.org/lurker/message/20060525.073637.78ce0660.en.html.
26. Krafft M. On the point of keysigning. 2008. http://madduck.net/blog/2008.01.28:on-the-point-of-keysigning/.
27. Synchronizing Key Servers. SKS OpenPGP Keyserver statistics; 2016. http://pool.sks-keyservers.net:11371/pks/lookup?op=stats. Accessed 31 Dec 2016.
28. Kamada T, Kawai S. An algorithm for drawing general undirected graphs. Inf Process Lett. 1989;31(1):7–15.
29. Tutz G, Schmid M. Modeling discrete time-to-event data. In: Springer Series Stat; 2016. https://doi.org/10.1007/978-3-319-28158--2.
30. Klein JP, Moeschberger ML. Survival analysis: statistical methods for censored and truncated data. New York: Springer-Verlag; 2003. https://doi.org/10.1007/b97377.
31. Kaplan EL, Meier P. Nonparametric estimation from incomplete observations. J Am Stat Assoc. 1958;53(282):457–81.
32. Aalen O. Nonparametric inference for a family of counting processes. Ann Stat. 1978;6(4):701–26.
33. Muller H-G, Wang J-L. Hazard rate estimation under random censoring with varying kernels and bandwidths. In: Biometrics. 1994. p. 61–76. https://doi.org/10.2307/2533197.
34. Prentice RL. A log gamma model and its maximum likelihood estimation. Biometrika. 1974;61(3):539–44.
35. Akaike H. A new look at the statistical model identification. IEEE Trans Automatic Control. 1974;19(6):716–23. https://doi.org/10.1109/TAC.1974.1100705.
36. Prentice RL. Discrimination among some parametric models. Biometrika. 1975;62(3):607–14.
37. Cox DR. Regression models and life-tables. J R Stat Soc Ser B Methodol. 1972;187–220.
38. Schoenfeld DA. The asymptotic properties of nonparametric tests for comparing survival distributions. Biometrika. 1981;68:316–9.
39. Grambsch PM, Therneau TM. Proportional hazard tests and diagnostics based on weighted residuals. Biometrika. 1994;81(3):515–26.
40. Poo-Camaño G, et al. Herding cats in a FOSS ecosystem: a tale of communication and coordination for release management. J Internet Serv Appl. 2017. https://doi.org/10.1186/s13174-017-0063-2.
41. Henk P. Penning. analysis of the strong set in the PGP web of trust. 2015. http://pgp.cs.uu.nl/plot/.