Progression and Forecast of a Curated Web-of-Trust: A Study on the Debian Project's Cryptographic Keyring

Gunnar Wolf[1(✉)] and Víctor González Quiroga[2]

[1] Instituto de Investigaciones Económicas,
Universidad Nacional Autónoma de México, Mexico City, Mexico
gwolf@gwolf.org
[2] Facultad de Ciencias, Universidad Nacional Autónoma de México,
Mexico City, Mexico

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 PGP Web of Trust are insuficient 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, some recommendations on personal key-signing policies, and a statistical study and forecast on aging, refreshment and survival of project participants stemming from an analysis on their key-handling.

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 [5], 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 hs been *ported* to the FreeBSD and GNU HURD kernels as well [15,16].

F. Balaguer et al. (Eds.): OSS 2017, IFIP AICT 496, pp. 117–127, 2017.
DOI: 10.1007/978-3-319-57735-7_12

But besides all of its the 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 [13].

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 [20].

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 [14]. We described the full reasoning for this migration and an oveview of the process and its numeric impact in the project in [20].

The aforementioned migration prompted a study of the direct metrics of the keyring's health, such as those detailed by [19], 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.

2 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 secure communication between Web browsers and servers using the `https` protocol.

As we have presented [20], 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 [21]. 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 [19].

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] [3]. The work we will present in this paper is restricted to a small subset

[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.

thereof—As of December 2016, the SKS network holds over 4 million keys, while the active Debian keyrings hold only around 1500.

2.1 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.

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 [21], key sizes of 384 through 1024 bits were deemed enough; using longer keys demanded computing resources beyond what was practical at the time.

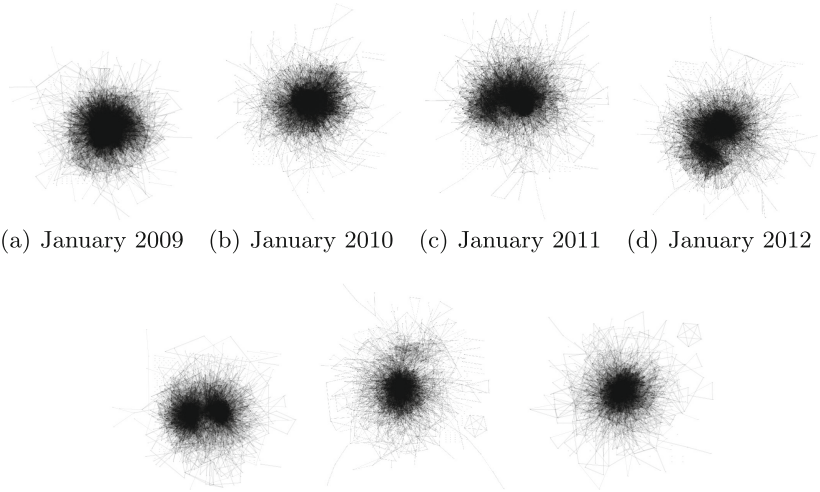
Of course, computers become more powerful constantly; cryptographic problems that were practically unsolvable 10 or 20 years ago are now within the reach of even small organizations [14, p. 11]. Cryptographic keys used for RSA and DSA algorithms should now be at least 2048 bits, with 4096 becoming the 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 [14, 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 [20], which prompted the present work.

3 Trust Aging and Reestablishment

The work done for the described keyring migration, as well as the migration process itself, presented a great 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 characterstics observed on the whole OpenPGP WoT [19] 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 [17].



(a) January 2009 (b) January 2010 (c) January 2011 (d) January 2012

(e) January 2014 (f) December 2014 (g) January 2015

Fig. 1. Snapshots of the Debian keyring evolution at different points in time

Figure 1 presents seven snapshots of the main developers keyring, processed by *Graphviz* using the *neato* layout program, which implements the *spring* minimal energy model [6]. 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[2] does lend itself to analysis.

Figures 1(a), (b) and (g) present a regular shape, approximately following Ulrich's observations, that the strong set of the WoT exhibits scale-freen. Quoting [19, Sect. 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. 1(c) 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 Figs. 1(d) 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

[2] 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 [10].

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, there is a tradition in Debian to announce travels in a (private, developers-only) mailing list, and active developers often will gladly meet people travelling to their region just for a key signature exchange.

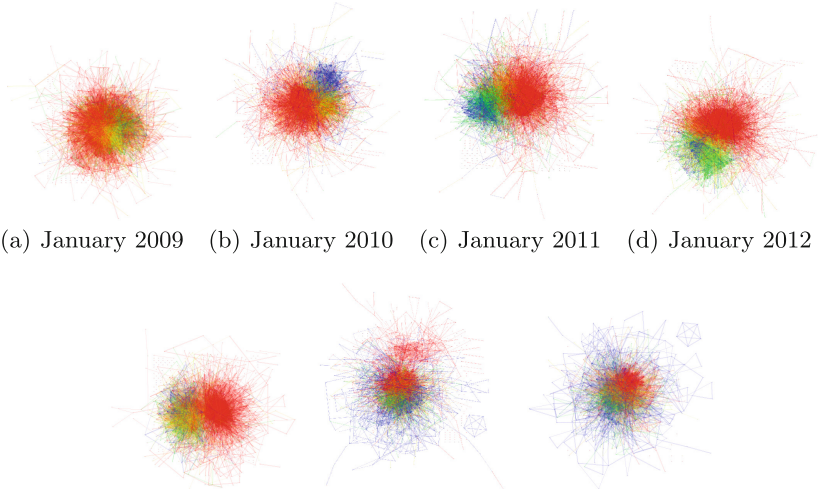
Although the number of developers that by late 2010 had migrated to a stronger key was still quite small, the call for key migration was initially answered by those with most active key activity –hence, probably more conscious about the importance of this migration. Of course, although it was not a targetted 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 Sect. 3– 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 practicaly 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. 1(f) 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. 2 correspond with those of Fig. 1, but with color-coded edges (according to the image caption).[3]

[3] 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.



(a) January 2009 (b) January 2010 (c) January 2011 (d) January 2012

(e) January 2014 (f) December 2014 (g) January 2015

Fig. 2. 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 graph (a) represent signatures made throughout 2008. (Color figure online)

Surprisingly, even Fig. 2(b) shows a clear grouping of keys by signature age? But this grouping does not appear a year earlier, in Fig. 2(a). 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 te tight, new (blue) group still in the peripheria, that eventually became the core of the newer *blob*.

4 Expectations on Key Survival

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* [18]. 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. Our sampled data is defined by the PGP keys that make up the curated WoT from the Debian Developers keyring [20].

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

Given the way the keyring is structured, we used key 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 PGP key activity record (has the key stopped getting signatures?) and keyring membership (has the key stopped being part of the keyring of interest?) which makes our data is 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, [8].

For this analysis, we must highlight a drawback from our failure definition: in real life there's no 1-to-1 correspondence of a key-person pair; key migrations – regarding our current study logic– mean one key dies (leaves the keyring) and another key enters as a completely independent individual. This will be refined in further analysis.

We used the R programming language with its `survival`, `flexsurv` packges; unless otherwise stated, significance level is assumed to be 5%.

Our first line of approach was first to show through the survival function the proportion of remaining keys in the keyring along time, that is keyring permanency. Then using the cumulated hazard function we get the expected exits per key that remains in the keyring until the endtime (in perpetuity). Finally, for 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, [7], the Nelson-Aalen moments estimator for the cumulated hazard function [1], and the kernel density estimator for the hazard rate function, [9].

A parametric estimation to see the *mortality law* fitting our data was made using a Generalised Gamma distribution through maximum-likelihood estimation [11]. The motivation for using the Gen. Gamma model is due to the closeness 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, [2], while also rejecting other models using a log-likelihood test of –5790.042 at 3° of freedom, [12]. The estimated parameters found for our model

[4] 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.

were $\mu = 4.6808$, $\sigma = 0.1990$, $Q = 3.4078$, with standard errors of 0.0134, 0.123, 0.2205 respectively.

In the non-parametric plot of Fig. 3, we observe downward steps when at least one key 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, at tag 107 (January 2015). Observed proportion of keys being above the theoretical model from tag 40 to tag 100 (around 4 years) suggests that after four years the keys wouldn't be much likely to leave; at least not until tag 95 where the probability of remaining afterwards is less than the 50% chance of heads in a coin flip.

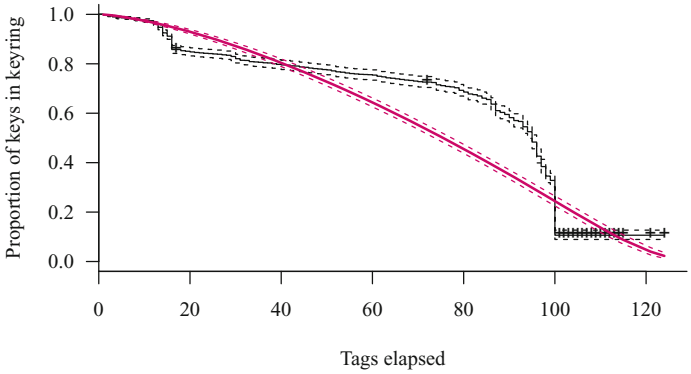


Fig. 3. Probability of key permanency. The black line follows observed (non-parametric) data from the keyring, with crosses representing the followup time for right-censored observations; the red line is the parametric estimation; dotted lines represent confidence bands. (Color figure online)

As we mentioned, due to the 1024-bit key migration, there is a clear skew that introduces a sharp drop around 100 tags; work is underway, as will be outlined in Sect. 5, to do a similar analysis based on personal identities instead of only key IDs.

Figure 4 shows the key 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-parameric exits is natural being the cumulated sum per tag of the exits over remaining keys 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 tag 40 to 100 (about 4 years), quickly increasing afterwards more than expected. It is not until near tag 100 that a key is expected to leave, but if for any reason it didn't or inmediatly enters again afterwards, it would be expected to leave again shortly after 3 tags (about 2 months).

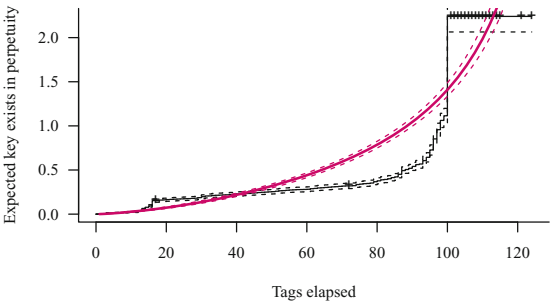


Fig. 4. Cumulated hazard of key exits. The black line follows observed (non-parametric) data from the keyring, with crosses representing the followup time for right-censored observations; the red line is the parametric estimation; dotted lines represent confidence bands. (Color figure online)

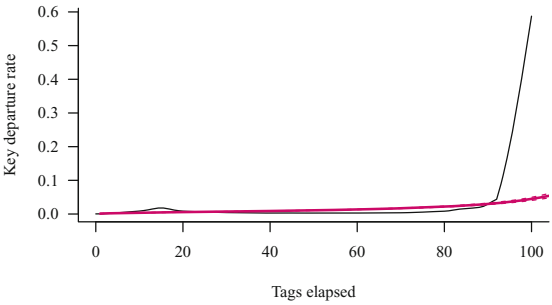


Fig. 5. Hazard rate of key exits. The black line follows observed (non-parametric) data from the keyring, the red line is the parametric estimation; dotted lines represent confidence bands. (Color figure online)

Figure 5 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 tag 90 (close to 6 years). Yet the parametric departure rate being under the non-parametric rate at the final tags shows the dramatic effect from the 1024 removal. Another remarkably finding was that departure rate in general doesn't grow far from 0 giving empirical evidence to say that 5 out of 1000 keys will leave "any time now" (from the fact that hazard rate is the instantaneous probability of failure at a specified time) in a 8 year lapsus.

In general keys in the project will constantly remain for about 6 years, as long as they went through 4 years, which in turn suggests further confirmatory analysis. It is just after the six years where keys effectively don't survive until the end.

5 Conclusions and Future Work

The Debian keyring is a very peculiar subset of the whole OpenPGP Web of Trust analyzed in [19]. 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.

As mentioned in Sect. 4, the survival study has so far been done around individual keys; work is underway so that a person's activity can be properly represented instead of following keys as separate individuals.

The resulting survival analysis can be used to generate cohort tables that further explain the keyring population for demographics, [4]. The outlined methodology can also be extended by introducing covariates such as the number of signatures received or network measures; 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. We are not aware of other projects having curated keyrings in a fashion that allows for analysis of their evolution over time.

Acknowledgments. We wish to thank GPLHost for donating the computing resources needed for this work.

References

1. Aalen, O.: Nonparametric inference for a family of counting processes. In: The Annals of Statistics, pp. 701–726 (1978)
2. Akaike, H.: A new look at the statistical model identification. IEEE Trans. Autom. Control **19**(6), 716–723 (1974). doi:10.1109/TAC.1974.1100705
3. Cederlöf, J.: Dissecting the leaf of trust (2004). http://wwwlysator.liu.se/~jc/wotsap/leafoftrust.html
4. Chiang, C.L.: Life table and its applications. In: Life Table and its Applications. Robert E. Krieger Publishing (1984)
5. Fernández-Sanguino, J., et al.: A Brief History of Debian (1997–2015). https://www.debian.org/doc/manuals/project-history/, Accessed 22 Dec 2016
6. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. **31**(1), 7–15 (1989)
7. Kaplan, E.L., Meier, P.: Nonparametric estimation from incomplete observations. J. Am. Stat. Assoc. **53**(282), 457–481 (1958)

8. Klein, J.P., Moeschberger, M.L.: Survival Analysis: Statistical Methods for Censored and Truncated Data. Springer, New York (2003). doi:10.1007/b97377
9. Muller, H.-G., Wang, J.-L.: Hazard rate estimation under random censoring with varying kernels and bandwidths. In: Biometric, pp. 61–76 (1994). doi:10.2307/2533197
10. Penning, H.P.: Analysis of the strong set in the PGP web of trust (2015). http://pgp.cs.uu.nl/plot/
11. Prentice, R.L.: A log gamma model and its maximum likelihood estimation. Biometrika **61**(3), 539–544 (1974)
12. Prentice, R.L.: Discrimination among some parametric models. Biometrika **62**(3), 607–614 (1975)
13. Robles, G., Dueñas, S., Gonzalez-Barahona, J.M.: Corporate involvement of libre software: study of presence in Debian Code over time. In: Feller, J., Fitzgerald, B., Scacchi, W., Sillitti, A. (eds.) OSS 2007. ITIFIP, vol. 234, pp. 121–132. Springer, Boston, MA (2007). doi:10.1007/978-0-387-72486-7_10
14. Smart, N.: ECRYPT II Yearly Report on Algorithms and Keysizes (2011–2012). Technical report 7th Framework Programme, European Commission (2012). http://www.ecrypt.eu.org/ecrypt2/documents/D.SPA.20.pdf, Accessed 14 Jan 2016
15. SPI et al. Debian GNU/HURD (1997–2016). https://www.debian.org/ports/kfreebsd-gnu/, Accessed 22 Dec 2016
16. SPI et al. Debian GNU/kFreeBSD (1997–2016). https://www.debian.org/ports/kfreebsd-gnu/, Accessed 22 Dec 2016
17. Synchronizing Key Servers. SKS OpenPGP Keyserver statistics (2016). http://pool.sks-keyservers.net:11371/pks/lookup?op=stats, Accessed 31 Dec 2016
18. Tutz, G., Schmid, M.: Modeling Discrete Time-to-Event Data. Springer series in statistics. Springer, Cham (2016). doi:10.1007/978-3-319-28158-2
19. Ulrich, A., Holz, R., Hauck, P., Carle, G.: Investigating the OpenPGP web of trust. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 489–507. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23822-2_27
20. Wolf, G., Gallegos-Garcí, G.: Strengthening a curated web of trust in a geographically distributed project. Cryptologia **41**, 1–16 (2017). http://www.tandfonline.com/doi/full/10.1080/01611194.2016.1238421
21. Zimmerman, P.R.: Why I Wrote PGP (1991). https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html