Safety in Numbers: Anonymization Makes Keyservers Trustworthy

Lachlan J. Gunn[1]
School of Electrical and Electronic Engineering
The University of Adelaide

Andrew Allison
School of Electrical and Electronic Engineering
The University of Adelaide

Derek Abbott
School of Electrical and Electronic Engineering
The University of Adelaide

lachlan.gunn@adelaide.edu.au

1. INTRODUCTION

Suppose you want to call someone, but do not know their phone number. How do you find it? The obvious way is to look them up in a phone book, but the phone company might have placed a different address under their name. If they are particularly security conscious, then you might presume that, when they received their phone book, they checked to see whether their number is correctly listed. But what if it were not modified in every phone book, but some contain the real number and some contain a false one?

This type of equivocation is a major problem faced by end-to-end secure communications systems, and the difficulty of its solution is such that the only systems with mainstream user acceptance, such as WhatsApp and SSH, tend to ignore it altogether, depending on Trust-on-First-Use [1] behavior.

Multi-path probing [1] has been proposed by a number of authors [2, 3, 4] as a means of detecting equivocation by network adversaries, whether near the server or the client. We demonstrate that it can be used effectively for anti-equivocation, a more general class of failure than the man-in-the-middle (MitM) adversary considered by [4]. This has been considered informally by a number of authors [3, 5]; we show that a system like DetecTor [3] can be made provably secure, with some modifications to render clients indistinguishable.

Our approach has a number of advantages over other open-group anti-equivocation techniques in the literature:

- **No bootstrapping problem.** By using an existing anonymity system to audit quite general services, new systems can obtain the benefits of distributed auditing without an existing community to provide operator-diverse monitoring systems.
- **Scalability.** Users do not need to communicate with each other, except to signal that the service has misbehaved. As a result, the communication overhead is only $\mathcal{O}(\log \epsilon)$ for a given security level ϵ.
- **Computational efficiency.** Because we do not use a proof-of-work system, no computational power is wasted on what is generally pointless busywork whose only purpose is to make participation costly.

  This is relevant to our first point: a new proof-of-work system is not secure until it has enough miners to out-compute any potential attacker. This creates a chicken-and-egg problem, in that the system is not secure until it is widely adopted, which will not happen if it is insecure.
- **No server-side cooperation needed.** This approach does not require any changes on the server-side; as a result, it is quite practical for motivated users to audit existing services without the need for effort or cooperation from their operators.

The analysis discussed in this paper is given in full in [6].

2. VERIFICATION PROTOCOL

Suppose that Bob wishes to acquire a piece of information from an untrusted anonymously-accessed service, and Alice the auditor can detect whether a given response from the server is valid. The protocol that we propose is as follows:

1. At a predetermined time, Alice and Bob both request a copy of the message from the service.
2. The service responds to their requests with the message provided.
3. Steps one and two are repeated M times.
4. If Bob does not receive M identical responses, he publicly signals an error.
5. Alice checks that the messages that she has received are identical and valid, and publicly signals an error if not.

We show this in Figure 1. Clients who see evidence of equivocation know that the service is untrustworthy, and can report its misbehavior. If the responses are signed, these clients can prove to third parties that the server has equivocated, providing a substantial deterrent to misbehavior on the part of the service.

Any anonymizing system can be used for this protocol, but in general a synchronized system such as a mix-net will be more effective, as these provide little-to-no room for timing attacks. In practice, low-latency anonymization networks such as Tor are far more available than mix-nets, and therefore measures such as randomization of request times are worthwhile; in our implementation[2], we do so by defining windows at fixed time intervals relative to an epoch, then making requests at a randomly-chosen time within that window. Regular inter-request intervals allow requests to be easily linked according to their time of receipt.

[1]LJG was supported by an Australian Government Research Training Program Scholarship.

[2]https://github.com/LachlanGunn/keywatch

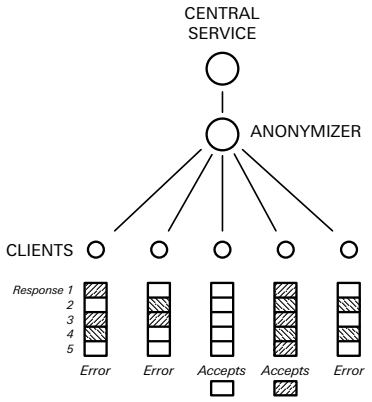

Figure 1: Interpretation of the results obtained from the protocol. Clients that have not received consistent responses from the server reject the response from the server, which they know to be faulty. Clients that have consistently received the same response accept it as unequivocated. In this figure, the server has equivocated, with the third and fourth clients being unaware of the fact and the others detecting the misbehavior of the service.

3. PRACTICAL SIGNIFICANCE

Our primary motivation for developing this protocol stems from the difficulty of remotely acquiring PGP key fingerprints. This normally requires manual verification of identification and email addresses, unless a trusted intermediary can be found; this type of key management is too hard for most users, and thus PGP is generally not used.

A user might theoretically verify that the public key-servers do not return inappropriate results when they search for their own name or email address, but a compromised server might equivocate. In order to use gossip protocols or monitor servers, the data to be authenticated must be small and common. This is normally achieved via Merkle tree structures, such as in CONIKS, but such wholesale replacement of the current keyserver architecture is not realistic in the short-term.

The approach that we describe here allows organic growth—users do not need to change the way that they publish their keys, but need only run a daemon that will perform background verification of their keyring. This dramatically reduces the barrier to entry, making adoption far more plausible in the short-term than cryptographically-protected systems such as CONIKS.

Our own implementation works in a similar way to this; the protocol is used via Tor to access the PGP keyserver network, with user-specified search terms. The response to each request is taken to be the first valid match, and a warning is emitted if this changes between requests.

4. RESULTS

We lack the space here to present a derivation of our results; however we summarize them in Table 1. Probabilities are with respect to a perfect anonymizer; our security reduction incorporates an imperfect anonymizer, and all probability bounds over M rounds are multiplied by $(1+\varepsilon)^M$, where ε is the probability of deanonymizing all users in one round.

	Diverse Requests	Common Request
Number of users assumed	2	$N \gg 2$
Items validated per user	L	L
Number of requests per user	ML	$M + L$
Probability of undetected failure	2^{-M}	$(N-1)$ $/N^M$
Legacy system support	Yes	No

Table 1: Costs and security of the proposed protocol for arbitrary services and Merkle tree-like systems, with N clients and M request-response rounds.

When all clients make the same request and expect the same response—for example, if the data is in a Merkle-tree as with CONIKS [7]—then they can assume that other clients are also making indistinguishable requests, resulting in a stronger security bound. In either case, the probability of non-detection falls exponentially with time.

5. CONCLUSION

We have shown how an anonymizing service such as Tor can be used to perform multi-path probing, and so create a web service that permits clients to detect equivocation. Failed attempts to provide different messages to different parties can be proven by the detecting party with the aid of digital signatures. Such a protocol has the potential to provide remote verification of public keys, rendering end-to-end public-key cryptography possible without the need for trust in certificate authorities or for potentially insecure approaches such as trust-on-first-use. We have bounded the probability that an equivocating service can succeed in deceiving its users, and have reduced its security to that of the underlying anonymizer.

6. REFERENCES

[1] D. Wendlandt, D. Andersen, and A. Perrig, "Perspectives: Improving SSH-style host authentication with multi-path probing," in *Proceedings of the USENIX Annual Technical Conference*, 2008.

[2] M. Alicherry and A. D. Keromytis, "Doublecheck: Multi-path verification against man-in-the-middle attacks," in *IEEE Symposium on Computers and Communications*, Sousse, Tunisia, 2009, pp. 557–563.

[3] K. Engert. (2013) DetecTor.io. Accessed 2017-02-20. [Online]. Available: http://detector.io/

[4] Y. Gilad and A. Herzberg, "Plug-and-play IP security," J. Crampton, S. Jajodia, and K. Mayes, Eds., Berlin, Heidelberg, 2013, pp. 255–272.

[5] The Tor Project, "Users of Tor," https://www.torproject.org/about/torusers.html.en, 2015, accessed 2015-12-31.

[6] L. J. Gunn, A. Allison, and D. Abbott, "Safety in numbers: Anonymization makes centralized systems trustworthy," 2017, ArXiV:1602.03316v2.

[7] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, "CONIKS: Bringing key transparency to end users," in *Proceedings of the 24th USENIX Security Symposium*, Washington, D.C., 2015, pp. 383–398.