Safety Meshing:
Hybrid trust models in social networks for end-to-end encryption

Authors
Max Kington, MSc (Royal Holloway, 2016)
Allan Tomlinson, ISG, Royal Holloway

Abstract

End-to-end encryption on social networks is the new vogue. New platforms, however, still face the same old problem: *trust*. We need new solutions. In a world with capable nation states, criminal gangs, hacktivists and everyone in between seeking to take advantage of the connected world, how do users stay safe? Centralised trust systems like certificate authorities ask us to put all of our trust in one place, a place we might not trust, at least not all of the time. Decentralised systems allow us to be discerning about who we trust and when, but are people really paying attention? In this article we introduce a way of taking elements of centralised and decentralised systems and combining them to combat some of their inherent weaknesses.[a]

[a]This article is published online by Computer Weekly as part of the 2017 Royal Holloway information security thesis series `http://www.computerweekly.com/ehandbook/Safety-Meshing-Hybrid-trust-models-in-social-networks-for-end-to-end-encryption`. It is based on an MSc dissertation written as part of the MSc in Information Security at the ISG, Royal Holloway, University of London. The full thesis is published on the ISG's website at `https://www.royalholloway.ac.uk/isg/`.

Online social networks have become a significant part of our lives. Three of the largest networks lay claim to over 1.5 billion users between them. These platforms, most of which are barely over 10 years old, already lay claim to a user base which spans over 20% of the world's population. In the UK at least, you would be hard pressed to find someone who does not use an online social network in one way or another. Whether used to exchange ideas, communicate or broadcast information their pervasive nature creates numerous opportunities for threat actors. As their worth to users increases, their worth to attackers also increases. In the last few years the topic of encryption has taken centre stage. Our understanding of the capabilities of well-placed and motivated actors has fundamentally evolved. As a result, there has been a major reboot of the public conversation about encryption. The level of public interest probably eclipses the last major public cryptography debate at the turn of the century, when encryption was still classed as a munition.

But encryption isn't the full story. For many users, service providers being compelled or coaxed into turning over decrypted traffic passing through their infrastructure is a serious concern. As a result, the gold standard became end-to-end encryption.

End-to-end Encryption

End-to-end encryption ensures that secrecy is maintained between the sender and receiver of a message.

The majority of systems used for building end-to-end encrypted channels rely on public key cryptography. The crux of any security system based on public keys revolves around *trust*: trust that the public key you are using really does belong to the person you think it does, that it is *authentic*. A number of different systems have been devised over the years to provide this authenticity:

- *Certificate authorities (CAs)*: Trusted entities who sign certificates attesting to the authenticity of public keys.
- *Web of trust*: A decentralised approach where individuals choose to sign the certificates of others and attest to their authenticity. Users then have to make a value judgement on whether or not to trust a given signatory.

Certificate authorities have been hugely successful. The deployment of SSL and TLS has powered commerce on the internet , but has not been without issues. The fundamental challenge with a central

trusted place is that it is a central trusted place. If a CA says it is true, our systems and our users will believe it. So long as the CA is throwing straight dice, we are happy; but this has not always been the case. There are numerous examples of compromises where fraudulent certificates have been issued and used in real world attacks. This is understandable - after all being a CA makes you a high value target for an attacker.

Web of Trust based systems have been a lot less widely deployed. Whilst these systems provide users with control, they don't necessarily provide the ease of use that CAs offer.

The challenge for engineers is to build usable systems for their users. But what if we could combine elements of both? Build a new model with the ease of use of a CA - create an authoritative source, but also make sure that we can check with others. It turns out we can, and with a model that really lends itself to social networks. The ultimate aim is to get certificates from the CA but to verify them automatically with the rest of your network and to *keep on monitoring them*.

First degree connection

Someone you can contact directly without going through another party.

The approach we take works by doing the following things:

- Asking for certificates for every first-degree connection the user knows.
- Establishing end-to-end encrypted channels with each of those connections.
- Verifying certificate information with each of these connections, both the user's own and those of others.
- Committing to automatic and ongoing monitoring of certificates.
- Gossiping between each other about certificate data.

The protocol

Step 1: The bootstrap

The first thing is for the endpoint (the software the user interacts with) to collect a list of users. A user base is vital to the success of the system. As we will see later, the stronger the relationships between users within a person's network the more robust the security of the system becomes.

A personal phone book, friend or contact list is a good starting point. With this third-party list in hand, a CA can be queried for the certificates of all on the list. This store of certificates is now maintained and monitored on an ongoing basis. Once a day, this is re-queried, keeping this list of certificates up to date.

This is the first opportunity for an attacker of the CA to try to subvert the user. It can provide a fraudulent certificate for all or some of the users. In this case the user at this point does not know if these certificates are authentic beyond trusting the CA.

Step 2: Certificate verification

The next step is to send a message to each first-degree connection, including your own certificate and their certificate (as you observe it), so as to establish a connection and perform verification. This takes the form of a bi-direction challenge-response where both users exchange certificate information. The critical part is that they send *both* their own and the other party's certificates. This allows a user to compare the remote user's certificate with the one initially obtained from the CA, allowing for detection

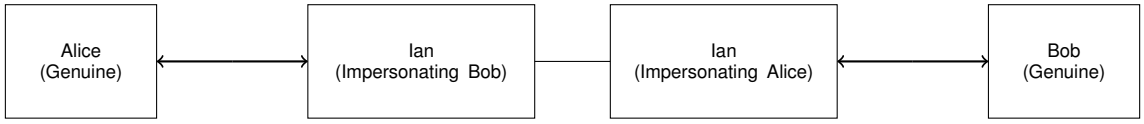

Figure 1: Back-to-back man-in-the-middle attack

of discrepancies. It also allows for verification that their partner has not been given a fraudulent version of their own certificate.

This step is vital and offers protection. By including their own certificate as well as their partner's certificates in a bi-directional fashion, an attacker would have to deploy a so-called back-to-back man-in-the-middle attack (Figure 1), where the objective is to not only deceive one user but two, delivering fraudulent certificates to both.

The other important thing is that confirmation of certificates can be done on the basis of existing knowledge. Where a user already has innate knowledge of a certificate this can be returned without querying the CA. This makes introducing a new fraudulent certificate undetected difficult.

> *A well placed attacker could drop some or all of these messages. In this case, certificates will not be confirmed and should not be used, and the channel will not be established. Ultimately, an attacker can no longer impersonate one user on a channel, but must subvert both in a back-to-back fashion.*

Step 3: Mesh confirmation

Mesh confirmation is where a user calls upon other members of the system to help with certificate confirmation. This is an important step which forces an attacker to take ever greater risks to subvert the system. During mesh confirmation the user will call upon other members of their network to confirm certificates. The mesh confirmation message contains the identity of a user, the certificate as observed by the requestor, and a response. The other important thing to note is that this certificate confirmation process ensures that an endpoint will monitor a verification process even if it was not expecting to do so. This leads to users in the system increasing their innate knowledge of certificates.

The endpoint selects a number of users to jointly perform mesh confirmation. This can be with users where they have frequent contact or infrequent contact. Where relationships within a network are strong (i.e. the frequency of communication is high) mesh confirmation messages can be included in normal communication which makes them much harder to selectively drop. This message allows a user to check with others what their view of a certificate is.

> *With mesh confirmation, an attacker now needs to perform back-to-back impersonation of more and more users. Where mesh confirmation is performed with users chosen at random the attacker will not know who to subvert ahead of time. If the attacker needs to start undermining a user by publishing a fraudulent certificate, previous knowledge held by other users will help detect this.*

Step 4: Ongoing monitoring

Each endpoint keeps an up-to-date store of certificates. The idea is that the memory of a participating user requires an endpoint compromise to force them to 'forget' a fraudulent certificate (and for it to avoid appearing in the gossip protocol). Ongoing monitoring is also important to avoid issues of very old certificates being kept around by isolated nodes or where users have not specifically interacted with a given participant leading to false assertions about fraudulent certificates being in circulation.

Step 5: Gossip

Essentially for every 'regular' content containing message, for example, "Hello Tim, would you like to go for dinner", a header is attached to the message which contains certificate and mesh confirmation messages, within the end-to-end encrypted payload. What is important is that people ask others to return a range of certificates in which they have interest. The endpoint does this by selecting a subset of known certificates at random.

Within social networks with strong ties, these confirmations can be done using innate knowledge; it is opaque to the well placed attacker which certificates are being confirmed as these will not trigger queries to the CA. Gossip messages need not be immediately replied to. This makes time-based correlation of messages harder (if the attacker can drop messages).

> *For an attacker to be able to impersonate any of the users in the gossip request message these messages must be either dropped or altered. Dropping is designed to be difficult for the attacker to do as they are embedded within an already encrypted channel and the attacker will not know if gossip messages are being included and if so for which certificates.*

Conclusion

The protocol serves to protect users against subversion of the centralised source of trust, the certificate authority. If an attacker starts to introduce fraudulent certificates they are forced to do one of two things:

1. Include more people in the deception, or
2. Isolate a user completely.

Bringing more people into the deception increases the likelihood of detection. Isolating a user means that impersonating a single person is possible but they may not ever speak to anyone else lest the certificate fraud be detected. In both cases our protocol forces an attacker to be hugely invasive in the traffic flow to perform an effective attack.

Note that key expiry and replacement and therefore certificate change is a necessary part of any key management lifecycle. If an attacker wants to start performing a back-to-back man-in-the-middle attack, they have to introduce pairs of fraudulent certificates, one for each party. A single certificate change is a normal operation but pairs or groups of simultaneous changes are cause for additional checking, and this kind of attack will likely be detected.

Social network behaviour & attack resistance

As part of our research, we made some assumptions about how social networks work based on our own experiences. The system we have designed has the following property:

> *The more participants that a client can interact with, the stronger the security properties of the system.*

Helpfully there have been many studies of social networks that provide detailed information, but the grandfather of much of this is by Mark Granovetter. In his research paper, The Strength of Weak Ties, he talks about *triadic closure* - intuitively, this says "if you and I are friends, I'm likely to become friends with your friends over time". Social networks are littered with these relationally overlapping groups. Our protocol leverages these social network traits implicitly. Networks which have strong triadic closure will have a greater amount of communication which means several things:

- Users are likely to be monitoring certificates of others already in the network to a higher degree of certainty, making the numbers of parties needing to be subverted greater and therefore more risky.
- As the gossip protocol is interleaved with their 'normal' communications it becomes harder to selectively isolate participants, as it requires the breakdown of the communications channel to such an extent that it becomes total isolation.
- Where strong triadic closure occurs, even if two participants are selectively isolated they will have friends in common performing certificate monitoring who will gossip between each other. This meshing effect will provide more protection.
- Where weak ties exist, the protocol provides distinctly less protection for the same reasons we have discussed.

Of course, not all social networks provide these kinds of overlapping relationships. There are a number of networks which do not act like typical networks but these, deliberately cellular structures, exist in many cases to avoid detection and to keep participants deliberately isolated. Granovetter would describe these networks as having "weak ties".

Final thoughts

We have designed a model which works well at increasing the trust in a certificate authority by using the nature of social networks to act as verifiers. This hybrid trust model combines elements of certificate authorities and the web of trust to provide a novel mechanism for keeping CAs honest. As part of our analysis we studied the SIGNAL Protocol. We determined that our model could be used to establish certificate information. Detailed attack analysis shows that within our goals the protocol stands to be very robust.

It is not without limitations of course. Security is a field of trade-offs where nothing is perfect. Our protocol isn't magic, nor is it infallible. It *increases* the effort of an attacker trying to subvert communications and forces them to take greater and greater risks which can lead to detection.

The protocol also does not enhance the privacy of people's relationships, just the contents of their communications. Arguably it does the opposite by passing around certificate information. True anonymity is a distinct and hard problem to solve. We have also assumed that in a system where users are sending messages through a central hub, which is often the case in these social networks, a well-placed attacker would be able to observe the message flow in any case.

Lastly we discovered that it is possible to provide differing levels of performance for different distinct groups. Almost by accident we have designed a protocol that can protects those operating in 'normal' social networks but not those whose interactions are deliberately covert.

Looking forward

During our research we had other ideas which we were not able to give proper treatment. We detail some of the more interesting ones below:

- Users could use second and third degree connections to obtain assertions about certificates by asking for certificate confirmation on the behalf of others. We would need to properly explore if these assertions are of any use and what the risk would be for users who failed to monitor properly subsequently introducing errors. The theory behind this is that this in turn increases the connectivity of users and has the potential to increase non-linearly the number of monitors therefore making it harder for an attacker to subvert all of the users.

- Where clients observe certificates which diverge from their own expected view they may make a note of these certificates and from whom they were received. They may then pass this information to the affected user, explicitly allowing the target of subversion to learn that they are being attacked.
- Initially we considered the idea that clients would build a chain of observed certificates and remember when they first observed them. They would then share these chains and allow clients to see when certificates diverged. The concern we had with this was effective time synchronization in being able to relate observations which would naturally lead to the need for tolerances in comparison.
- Endpoints may remember where, when and from whom they obtained certificates. This would allow endpoints to divulge where discrepancies were introduced. This could intersect with a consensus allowing the network to shun subversive users.

Biographies

Max Kington started working as a professional programmer in 2000 during the first dotcom boom. Since then he has led international teams in numerous different industries including gaming, consulting, telecommunications and financial services. He now works for a Fortune 100 Investment Bank leading on distributed systems and information assurance architecture within the cybersecurity practice area. In 2016 he was awarded the thesis prize for outstanding project after graduating with an MSc in Information Security from Royal Holloway with a distinction.

Allan Tomlinson BSc(Strathclyde) MSc, PhD (Edinburgh) is a senior lecturer with the Information Security Group (ISG) at Royal Holloway, University of London. He was awarded a PhD in 1991 from the University Edinburgh for work on VLSI architectures for cryptography. He then joined the Institute of Microelectronics at the National University of Singapore, working on secure NICAM broadcasting and in 1994 moved to General Instrument in California to work on the Digicipher II pay-tv system. Before joining the ISG in 2003, he was Principal Engineer at Barco Communications Systems where he was responsible for the development of the "Krypton" video scrambler. His current research interests are in systems security and trusted computing.