Direct File Transfer System via WebRTC
An Alternative to E-mail Attachments with Improved Security

Robin Lunde, Mobile And Ubiquitous Internet
Keio University Shonan Fujisawa Campus

Main advisor: Keiji Takeda Co-advisors: Jun Murai
Osamu Nakamura

***Abstract*— Utilizing WebRTC's P2P technology, this thesis will suggest an alternative to the way e-mail attachments currently work. In today's technical environment, there are countless new and secure ways to send files over the Internet. Yet most people still use the traditional, outdated e-mail attachment technology to share files. This thesis will propose a new system that seeks to improve overall security and usability of transferring files.**

The system will be evaluated against the current e-mail system, as well as cloud and SNS based solutions. The comparison will largely be focused on security and usability. This thesis will clearly show that the current e-mail system is not sufficient when it comes security, usability, or both. It will propose a solution that raises the standard of security and confidentiality, which improves the current conditions and offers an alternative solution.

It suggests a system utilizing the web of trust model combined with WebRTC's P2P functionality. By using the web of trust model, the end-user gets more control over whom he trusts, while simultaneously avoiding the problem of having to authenticate users against a central server or service. The P2P functionality of WebRTC allows for the direct transfer of files between users, avoiding the need for servers to store the files in transit. This reduces the risk of an attacker gaining access to the file, while also optimizing transfer-speed. This system was implemented in a prototype to demonstrate the feasibility of the proposed system.

I. Introduction

SendIt's primary goal is to be an easy to use application for improving security when transferring files. It aims to reduce the risk of data theft while still being usable by people lacking technical insight. It also aims to improve the security compared to commonly used solutions. We believe there is a lack of good solutions currently available. This is illustrated by the fact that it is not uncommon for a decryption-key and cipher to be sent over the same channel when sharing encrypted files. This holds true even for security specialists!

There is also a global trend towards higher requirements regarding data protection and handling, as demonstrated by EU's new General Data Protection Regulation [1], [2] and Special Publication 800-171 in the US. [3] There is a strong demand to minimize exposure of personal data. With this in mind, we seek to be in compliance with these trends.

We also seek to utilize peer-to-peer technology, as it keeps in mind the original idea of the internet being a decentralized system. [4] It also allows SendIt to directly transfer the file(s), without having to store it anywhere in transit. This reduces the attack surface, reducing the risk of data leakage.

II. Related work

A. Existing solutions

There exists solutions available and under development that resemble SendIt. FireFox Send [5] is a service that aims to make sending files easier in an encrypted fashion. They are using a cloud service to store the file. The link expires after 24 hours or the indicated number of downloads. This is in contrast to SendIt's direct solution.

Tox is a solution that is fairly similar to SendIt. It uses DHT to create a network layer for finding connections and another DHT network layer for connection setup between nodes. This results in a decentralized, P2P network with end-to-end encryption. Unfortunately using DHT means it is harder to maintain and deploy, and that one needs to enter the DHT through certain nodes. These problems are not present in SendIt. [6], [7]

There is also an existing implementation that we used as a reference for SendIt. The Serverless-WebRTC solution [8] was the proof of concept that inspired SendIt.

B. Contributions

Many solutions to share files already exists, with a few mentioned above. SendIt uses a direct, serverless solution, which is a rare approach. The exchange and management of identities combined with peer-to-peer transfer and trust evaluation is original to the suggested system. The way of establishing trust is also unique to the system. In summary, a direct transfer system, based on non-absolute trust for the first interaction, with trust evaluation, is our original research.

III. System

SendIt uses e-mail addresses as usernames in order to have unique identities. It relies on non-absolute trust for the first interaction, after which keys are exchanged, and used for authentication from then on. The reason for choosing this solution is that it allows for an easy and convenient way to start communicating with new partners. It is important that the system is kept simple, to make sure it stays user friendly.

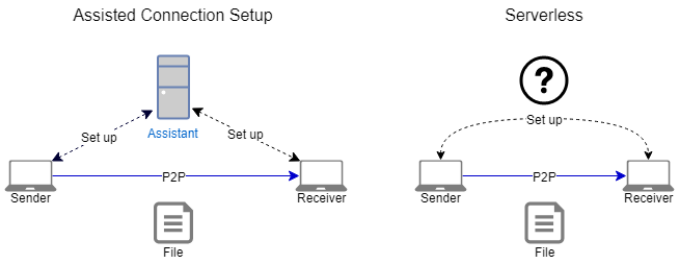

Fig. 1. Left: Assited Connection Setup Mode, Right: Serverless mode

While the first exchange is not encrypted, all subsequent communications are. This guarantees that we are communicating with the same identity as before, and that all communication is confidential and can not be changed in transit. This is done using Public-key cryptography combined with session keys. A public and private key is generated and linked to each identity. A new session key is generated for every session and used to encrypt the data being transferred. This session key is encrypted using the public key of the other node before being shared, to ensure confidentiality.

A. Use and use-cases

SendIt's primary use-case is when sharing sensitive or critical data. It can also be used if you want comparatively better privacy, simple file-sharing, et cetra, compared to regular e-mail attachments. It is also compliant with the regulations previously mentioned. In short, it is a user friendly system with focus on privacy and security.

SendIt achieves this by implementing two different ways to create a peer-to-peer connection between two nodes as shown in Figure 1. The file is always transferred over a secure P2P-channel and with end-to-end encryption.

B. Technology

There are three base technologies used in SendIt. The first is WebRTC, which is used to create peer-to-peer connection for transferring the data directly. This is done by exchanging an offer and an answer containing information on how to create the connection. The second is Electron; used to create a client application without needing any server involvement. It also makes building a desktop application using web-technology possible, as well as easy application deployment. Finally, the web of trust model is used to manage and evaluate trust in identities. This is used to minimize risk of attacks and to update and re-evaluate each identity's trustworthiness. It also allows there to be a somewhat shared consensus in the system, about which identities are trustworthy.

IV. Evaluation of SendIt

To understand the contribution and usability of SendIt, an analysis of existing solutions is needed. By comparing SendIt with existing solutions it is possible to get an idea of what has been improved, what stayed the same, and what still needs to be improved. In the following section SendIt will be compared to the e-mail system and a generic SNS system, and the findings will be evaluated and analyzed. This evaluation will not include all attack vectors, but rather focus on the weaknesses that come as a consequence of the proposed system and its design.

A. Possible limitations

1) First trust: The main limitation is the first trust. This can allow an attacker to appear as the legitimate owner of an identity, without this being the case. This is a risk that users should be aware of, as it is a critical part of the system. Even though it is a part of the design choice, it has to be acknowledged that this would ideally be done in a more optimal way.

The mitigation for such attacks is split in two parts. The first is relying on the user to evaluate whether the user is legitimate. By using social cues like timing, each user should evaluate if it is likely that the other identity would initiate a transfer. One can also contact the other endpoint and confirm that it is actually them. This is, of course, **only necessary for the first connection**.

The second is the trust model, which is an automatic part of the system. It combats this issue by sharing information and allowing users to have as much data as possible, for all identities. This allows for detection of false identities and will eventually make these identities untrusted entities in the whole system.

2) ACS: Another attack vector is the ACS server. If it is not trusted, it can pose a security risk, as the endpoint has to rely on the fact that the ACS server relays the data to the correct recipient. If the data sent is encrypted, it will not pose any immediate risk, but it will stop the endpoint from connecting to anyone, as the other endpoint will not get authenticated properly. If it is the first time these endpoints are connecting, however, it puts the victim at risk of being put in touch with the wrong identity.

The ACS server can also act as a logger that keeps track of which identities are communicating with each other. While not directly an attack, it can raise privacy issues, since it allows for monitoring of traffic. It can also be argued that it can be used for reconnaissance, which allows for a targeted attack on one of the endpoints. Because of this, endpoints should only use ACS servers they trust.

B. Advantages

1) Connection setup: While P2P transfer is direct communication between two endpoints, it needs some way to set up the connection. DHT solutions rely on connecting to specific endpoints, called bootstrapping, to enter the network. This is predictable behavior that can be exploited by attackers, and it also increases the difficulty of using the system.

As such, the system suggested leaves it up to the end users to decide how to share the information, as it is unpredictable and allows for easier usage of the system. It removes the need for any bootstrapping or server, and does not specify any set way to share the connection information.

2) Direct transfer: The biggest advantage SendIt has over the other systems, is that it transfers the file directly. None of the issues stemming from files being stored in transit applies

to SendIt. These issues range from attackers hacking the server and getting access to the files, to allowing attackers to monitor data going through the server, since the data will always pass through this point. As a general rule, a resource is more secure, the fewer copies exist, since it limits the attack surface.

Another point to take into consideration is that even if only encrypted data is stored on a server, if an attacker gets access to the data, it is only a question of how long it takes to break the encryption. As such, minimizing the risk of leaking any data should be considered critical. This is why direct transfer of files offer such an advantage.

The attack surface which is exposed is also important to evaluate. Having a server store the files, means the attack surface is quite large, since it is always online. With SendIt's direct transfer function, the file is only available during transfer. The time difference for how long the file is available to be attacked is huge. As such, this is another advantage of direct transfer, since the attack surface is significantly reduced.

3) Content control: In the other two systems, one is not able to stop unwanted content. If someone shares a file, it will be delivered to the receiver's PC, or at least their e-mail server / cloud account. This is unfortunate, since receiving malicious files may cause them to be executed at a later time, or by an unintended action. It is also possible that they may execute without the user knowingly doing so. As such, it is optimal for the user to be able to stop the transfer of unwanted files, something which SendIt supports. Automatic spreading of malicious files is a big problem. Files are spread to all the victim's contacts and keeps spreading in a similar fashion. While SendIt has no direct solution to mitigate this, the fact that it does not support multicast, and that the files cannot be automatically received, naturally resolves this issue.

C. False E2E encryption

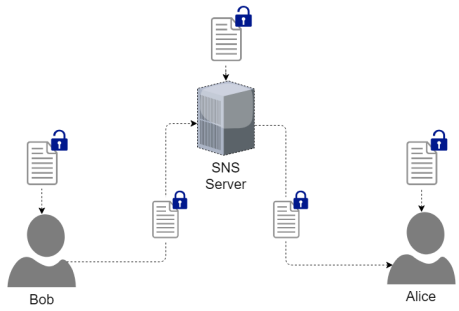

Fig. 2. False end-to-end encryption used in some SNS. The service has access to the keys used to encrypt the data. This allows the service to decrypt the data you send.

As shown in Figure 2, false E2E encryption is sometimes utilized in cloud- and SNS-systems [11], [12], [13]. This is a scary thing, as a consumer of these services, because both you and your communication partner are unable to verify if the communication has true E2E encryption or not. Effectively, this is a MITM-attack by design. As such, when using these services, one should assume that the data can be decrypted and read by the service and whoever they disclose the data to. For SendIt, the public-key cryptography guarantees end-to-end encryption, since the keys are exchanged over P2P, which means no central service can tamper with the keys sent.

V. Conclusion

We have proposed a new prototype and system for transferring files. SendIt is using P2P communication to avoid having to go through a server to share files. Since this makes the system vulnerable to multiple kinds of attacks, an authentication scheme based on public-key cryptography is suggested.

Since there is no central authority to vouch for any of the keys or identities used, the solution combines the above technologies with a trust-system based on the web of trust model. This is a reputation based, decentralized model which allows each user to make their own choices, while still having a shared understanding of how trustworthy each identity is. This system clearly offers some improvements to the current solutions and gives a new perspective on trust establishment. All in all, we believe that SendIt is a good alternative to the way current e-mail file attachments work, while still having room for improvement.

References

[1] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance). (2016).

[2] Kiesel, J.: GDPR Compliance: Summary & Requirements You Need to Know, https://linfordco.com/blog/gdpr-compliance-requirements/, (2017).

[3] Ross, R., Viscuso, P., Guissanie, G., Dempsey, K., Riddle, M.: Protecting controlled unclassified information in nonfederal systems and organizations. National Institute of Standards and Technology, Gaithersburg, MD (2016).

[4] Ibanez, L.-D., Simperl, E., Gandon, F., Story, H.: Redecentralizing the Web with Distributed Ledgers. IEEE Intelligent Systems. 32, 9295 (2017).

[5] Firefox Test Pilot - Send, https://testpilot.firefox.com/experiments/send/. Last accessed 20 Mar 2018

[6] The Tox Project, https://tox.chat/about.html, Last accessed 15 May 2018

[7] Tox protocol specification, https://toktok.ltd/spec.html#introduction, Last accessed 15 May 2018

[8] Serverless-webRTC, https://github.com/cjb/serverless-webrtc, Last accessed 20 Mar 2018

[9] Jesup, R., Loreto, S. and Tuexen, M.: WebRTC data channels. draft-ietf-rtcweb-data-channel-13. txt, work in progress. (2015)

[10] Alfarez A.: The PGP Trust Model. EDI-Forum: the Journal of Electronic Commerce **10**(3), 27–31 (1997)

[11] Greenwald, G. et al. Microsoft Handed the NSA Access to Encrypted Messages. In: The Guardian. US news (). ISSN: 0261-3077. http://www.theguardian.com/world/2013/jul/11/microsoft-nsa-collaboration-user-data, Last accessed 06 Jul 2018

[12] Popa, B. Skype Provided Backdoor Access to the NSA Before Microsoft Takeover [NYT]. http://news.softpedia.com/news/Skype-Provided-BackdoorAccess-to-the-NSA-Before-Microsoft-Takeover-NYT-362384.shtml, Last accessed 06 Jul 2018

[13] Is It Safe to Transfer Files via Skype?, http://smallbusiness.chron.com/safe-transfer-files-via-skype-66706.html, Last accessed 06 Jul 2018