DIRECT FILE TRANSFER SYSTEM VIA WEBRTC

An Alternative to E-mail Attachments with Improved Security

Robin Lunde

Mobile And Ubiquitous Internet

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

Introduction

🔑 Key points

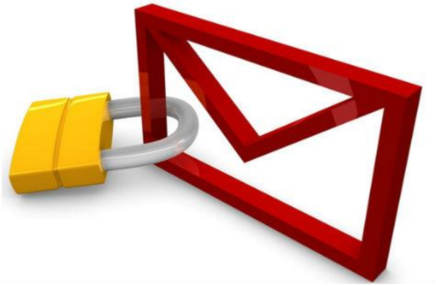
- Developed a prototype – SendIt

  Secure, *(Serverless,)* Electron & NodeJS-based, Direct Information Transfer

- Simple, user-friendly system
- Complimentary
  - Not a replacement
- Usable, improved security!
  - Absolutely security not the goal

Use cases

- Secure file transfer
  - Setup in person
  - Slightly inconvenient
  - Security comparable to PGP
- Easy way to transfer files
  - Setup over internet
  - Convenient
  - Reasonably secure
- Compliant with new regulations
  - Privacy & Data control
  - *GDPR* (EU) / *SP 800-171* (US/World)
  - Direct transfer – No storage in transit



Trust & Authentication

- Non-absolute authentication:
  - Timing
  - Files offered
  - Sender
- Used for first interaction only!
  - First time communicating
  - Share keys
- Authentication dependant on first trust:
  - Done in person – Best!

Goals & Contributions

Goals

Work

Contributions

Goals

- Improve security and ease of use for e-mail attachments
- Minimize risk of leakage and exposure of personal data
- Create a system with focus on usability, privacy and security
- Create a prototype to show feasibility

Contributions

- New type of system
- Client-only development
- New perspective on e-mail attachments
- Security to the people
- Serverless implementation
- Decentralized internet

Concepts & Design

Trust system

- Not suitable to use PKI
  - Requires setup
- Non-absolute authentication
- Used for first interaction only!
- Based on Web of Trust
  - Utilizes Public-key cryptography, like PGP
- Trust transitivity*
- Gradual trust building*
  - Negative > Positive
- Trust re-evaluated constantly*

*Not implemented but framework designed.

Authentication

- Public-key cryptography
- Identity represented by e-mail
- Authentication and connection setup combined
  - Usually separate processes
- WebRTC Offer & Answer
  - Includes endpoint authentication

- Keys shared:
  - Over P2P channel
  - Encrypted
  - During first connection
- Depends on first connection:
  - In person – More secure
    (Highest level of security in SendIt)
  - Open channel – Less secure

Implementation

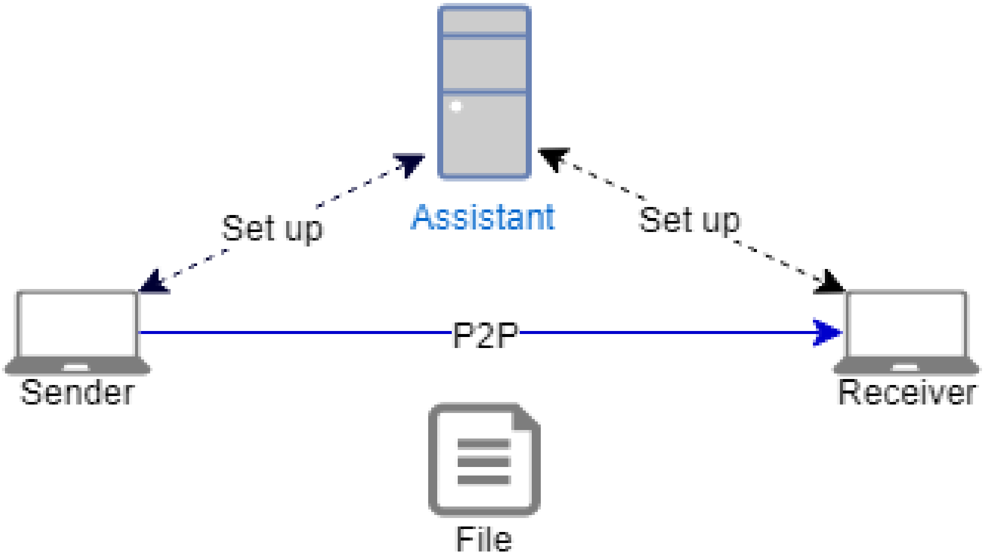
Technology

- System is built using:
  - NodeJS
  - Electron
  - WebRTC
    - Experimental technology!
- Enables easy development:
  - Desktop applications
  - Multi-platform support
  - Built-in NAT traversal
  - P2P communication
  - No server requirements

Modes

The modes only differ in how the *connection information* is shared

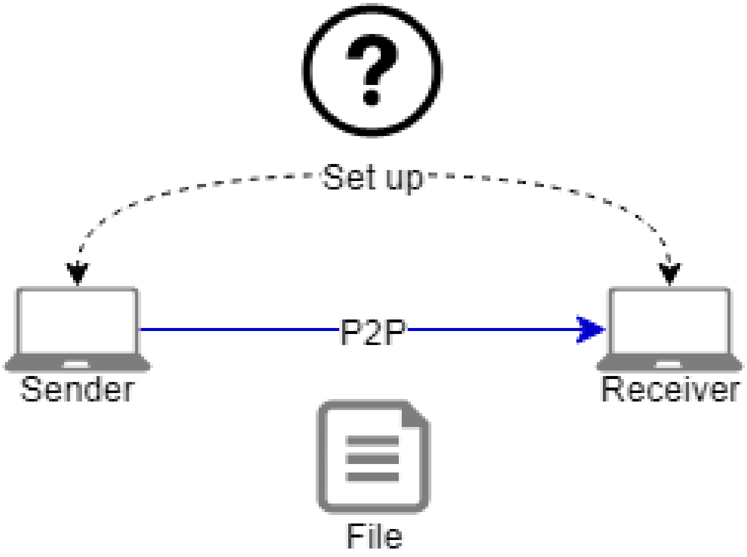
Assisted Connection Setup



__Automatic sharing:__

Easier to use – less secure

Serverless

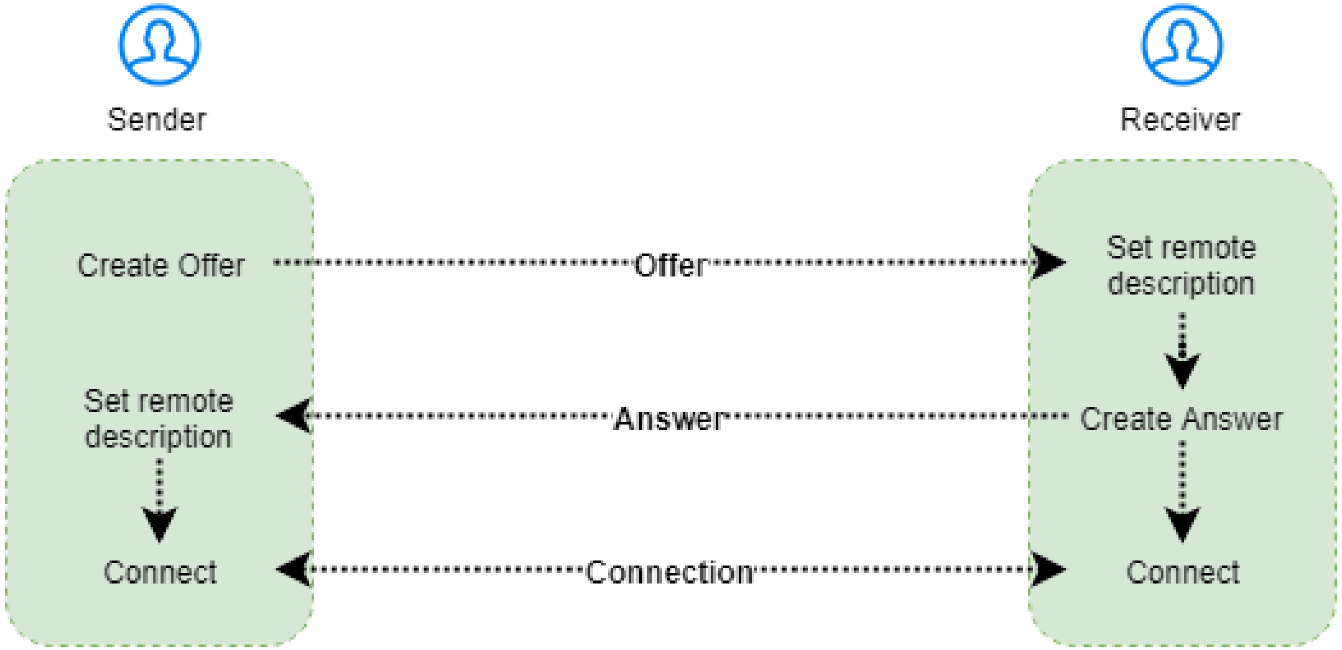


__Manual Sharing:__

Harder to use – more secure

Usability evaluation

- Only Serverless mode
- WebRTC connection setup
  - Experimental!
- Experiment
  - Simulating user behaviour
- Total setup time:
  - Around 6 minutes
- Offer last longer than Answer



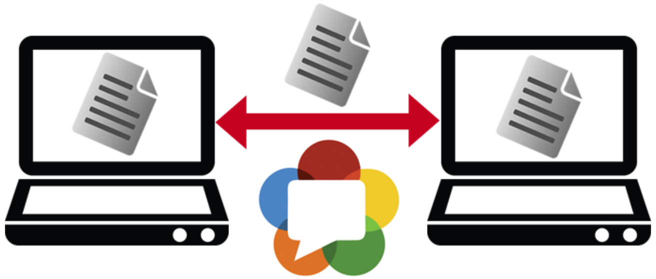
Transfer speed & efficiency

- Same as popular P2P systems
- Depends on network conditions
- Works on any network
  - Except Symmetrical NAT
  - Optimal = LAN

- Test transfer WIDE -> Yamato:
  - 40 Megabytes transferred
  - Completed: 34 Seconds
  - Speed > 10Mbyte/s!

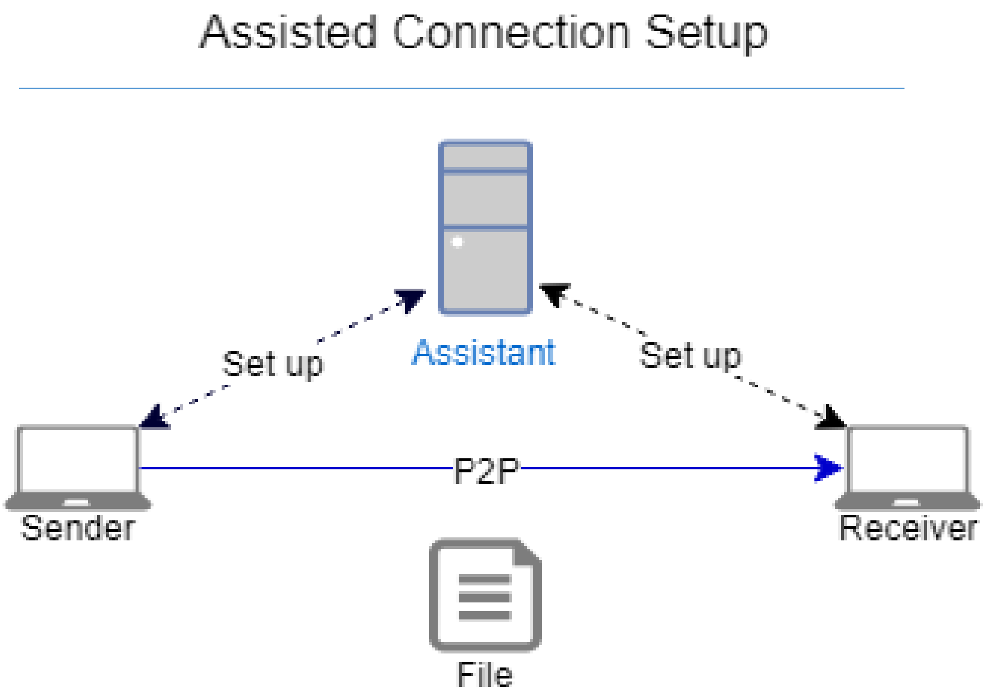
WIDE – Dl: 1000 Mbit/s, Ul: 735 Mbit/s

Yamato – DL: 85Mbit/s, Ul: 10Mbit/s

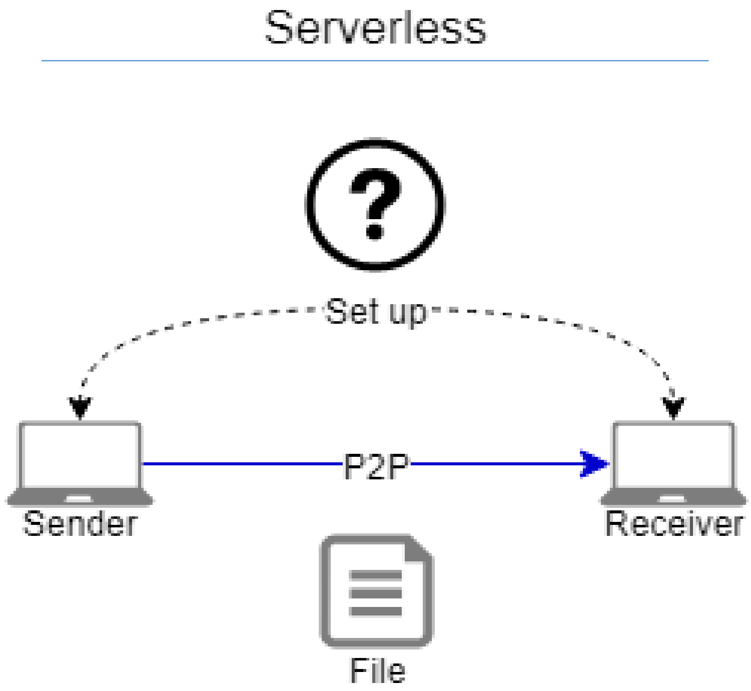


Demonstrations

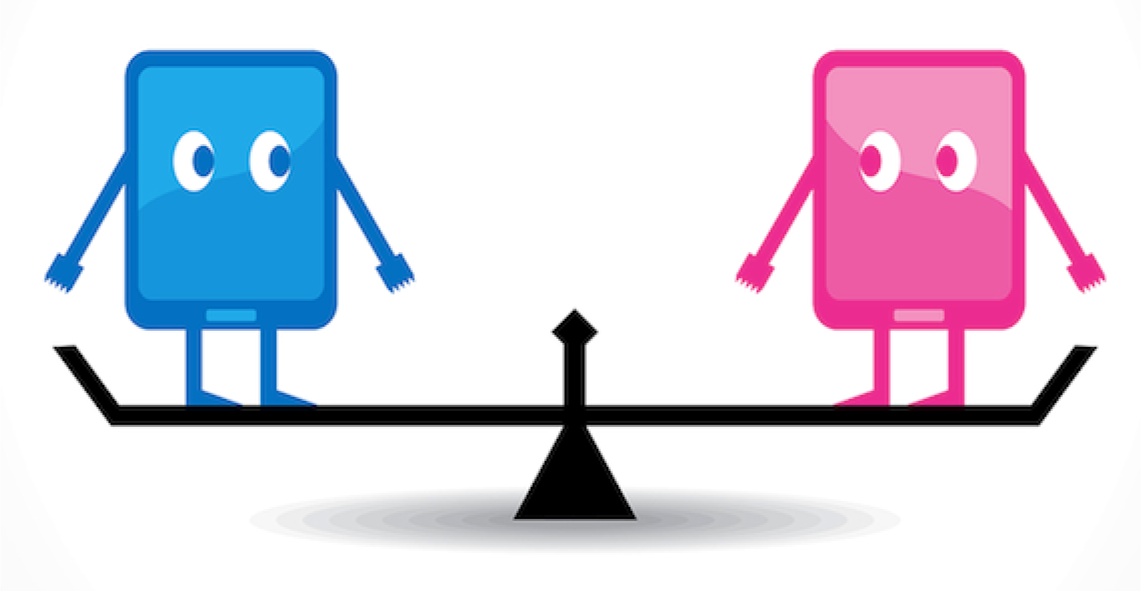
ACS Demo


Assisted Connection Setup
Assistant
Set up
Set up
P2P
Sender
Receiver
File

Serverless Demo



Evaluation & Comparison

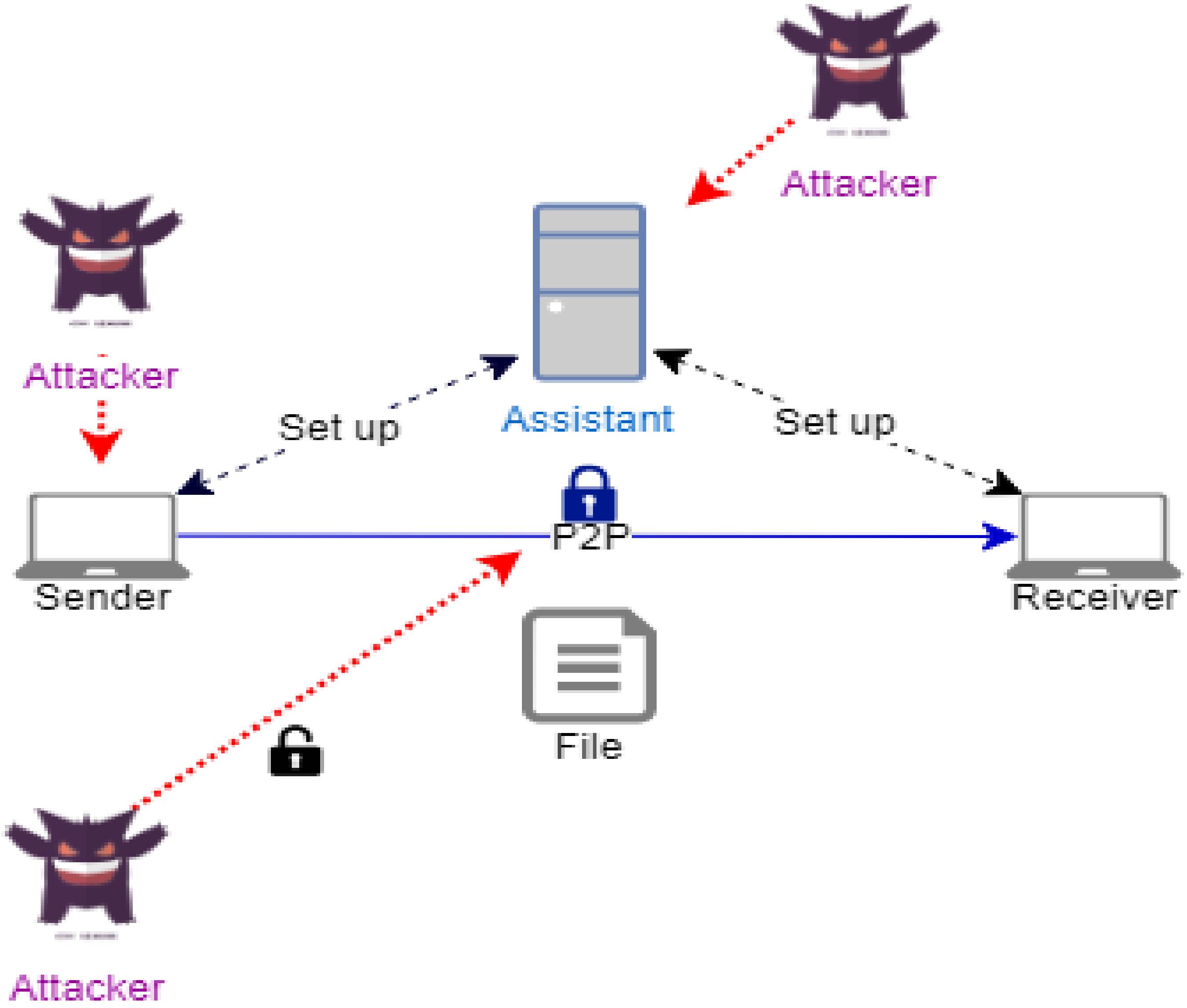


Evaluation

- Comparison based:
  - E-mail
  - Cloud- and SNS
- E-mail offers no guarantee of any security features
  - Anything is an improvement
- File storage in Cloud- and SNS-based systems:
  - Large attack surface
  - Low content control
- SendIt's main weakness:
  - First trust abuse

SendIt threat evaluation

- Authenticate with false identity *(First exchange)*
- Key theft
- XSS
- Malicious Assisted Connection Setup
- Break encryption
- Compromise Sender's computer



Comparison

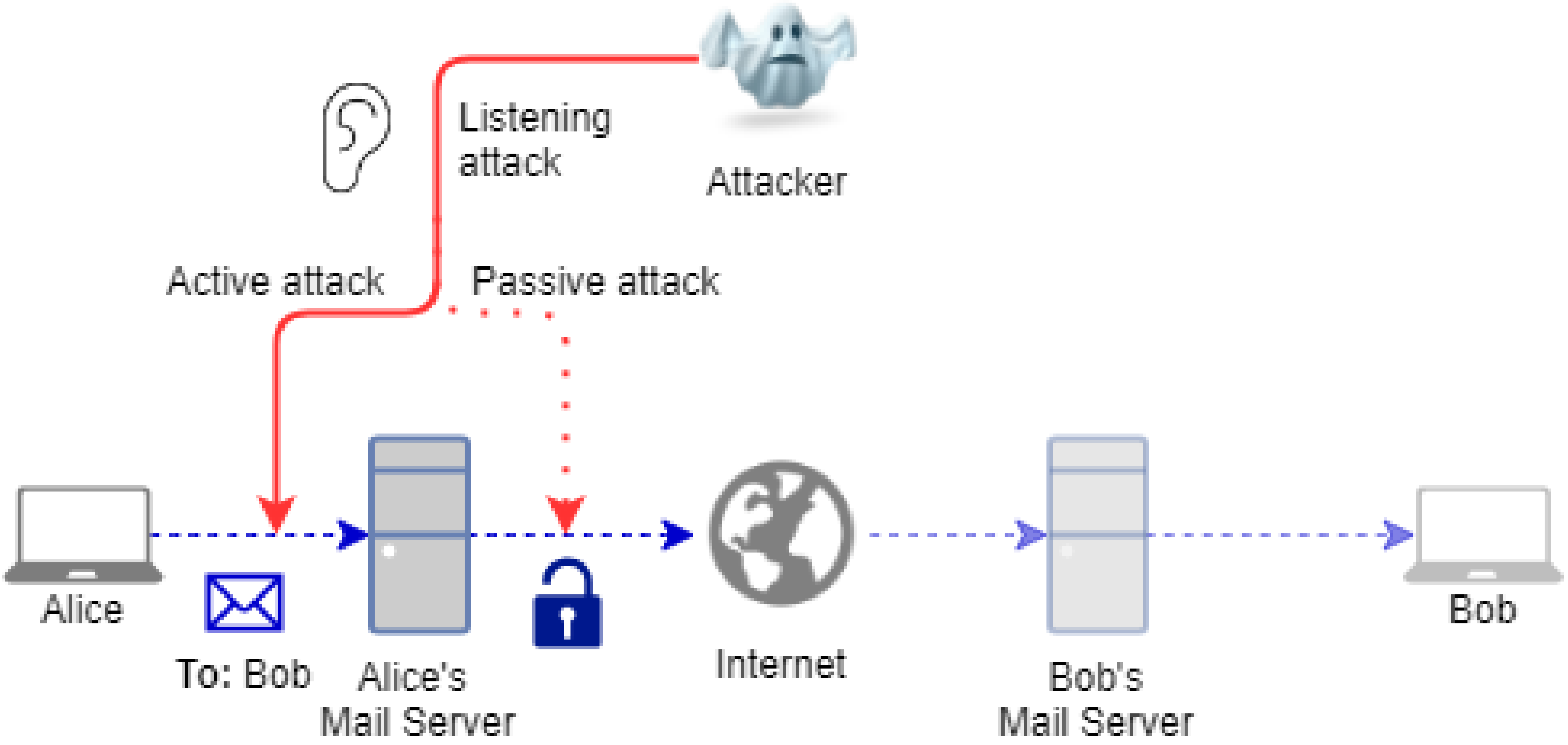
−

- Asynchronous
  - Must be online simultaniously
- First connection is trusted
- Requires connection setup
- No multicast support

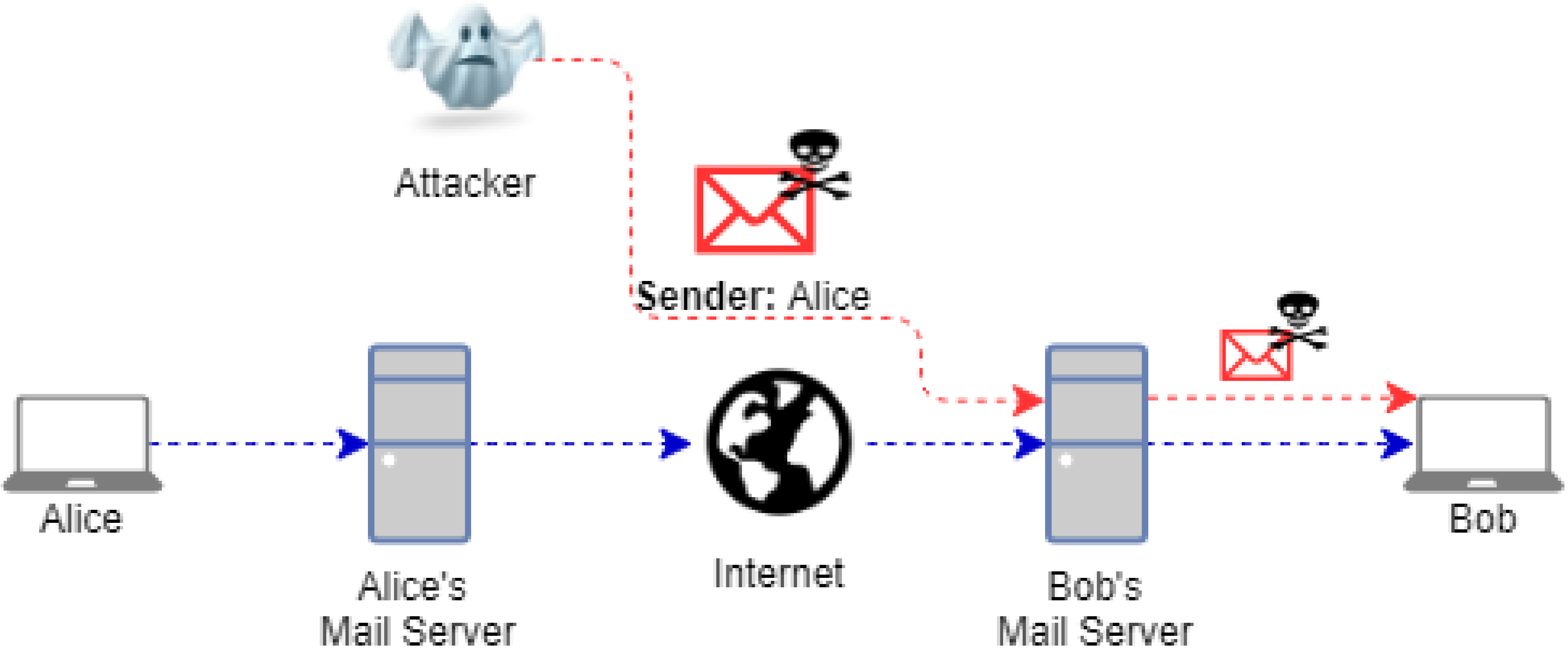
+

- **Low attack surface**
- **Direct transfer**
- Easy to use
- Simple system
- Reasonably secure
- Content control
- Authentication

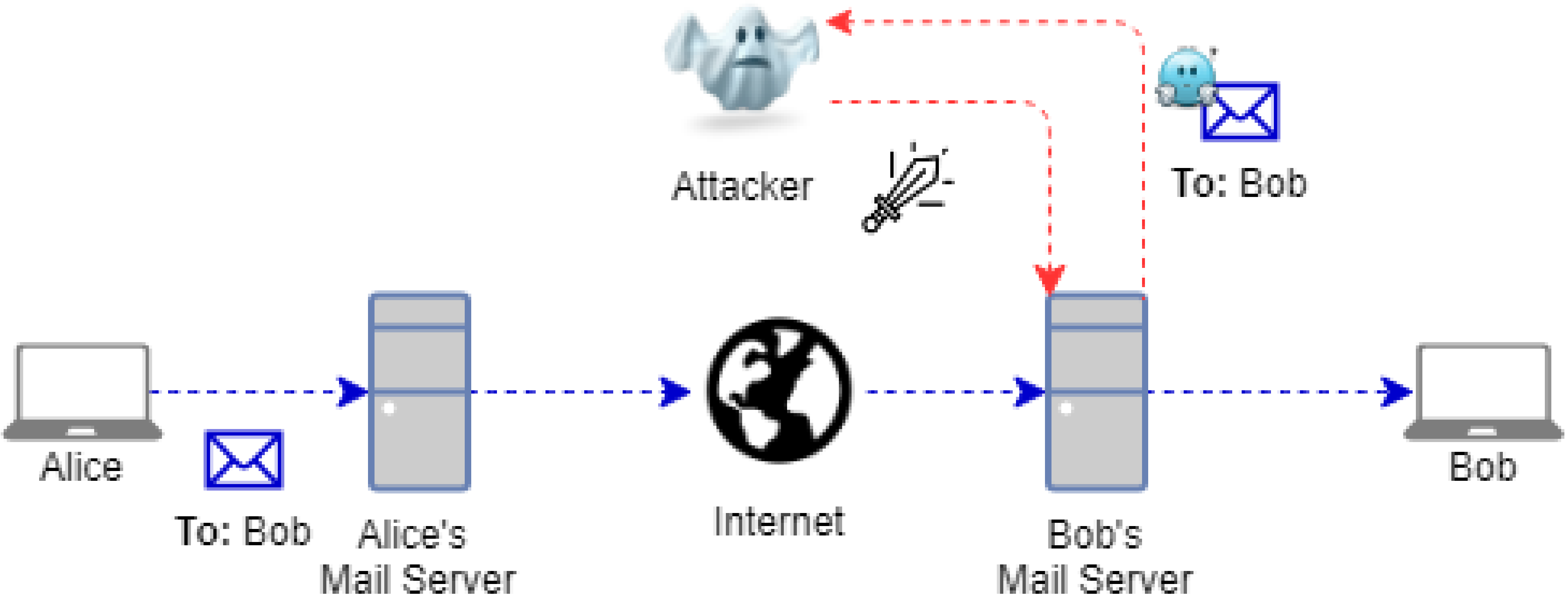
E-mail threat 1



E-mail threat 2



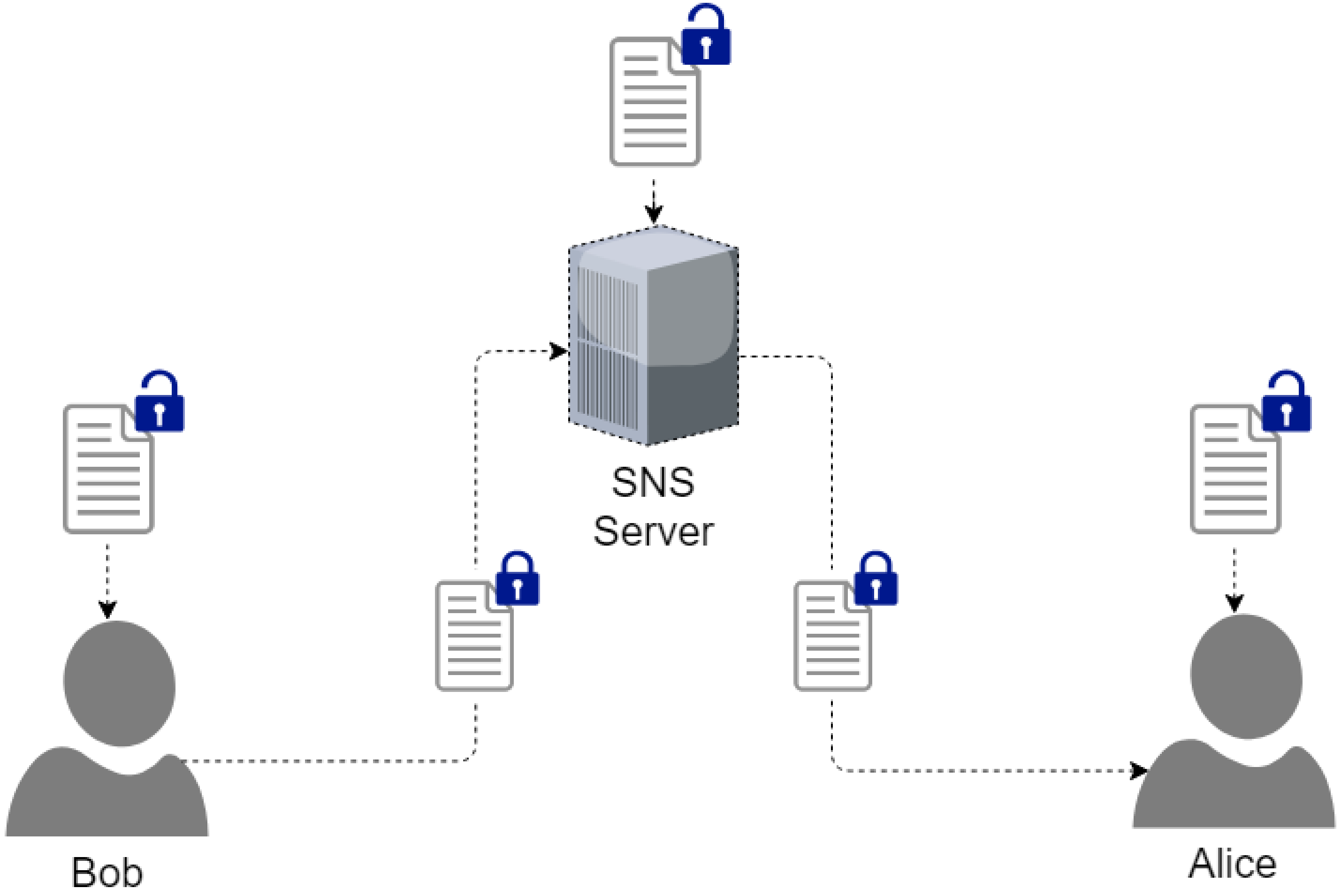
E-mail threat 3



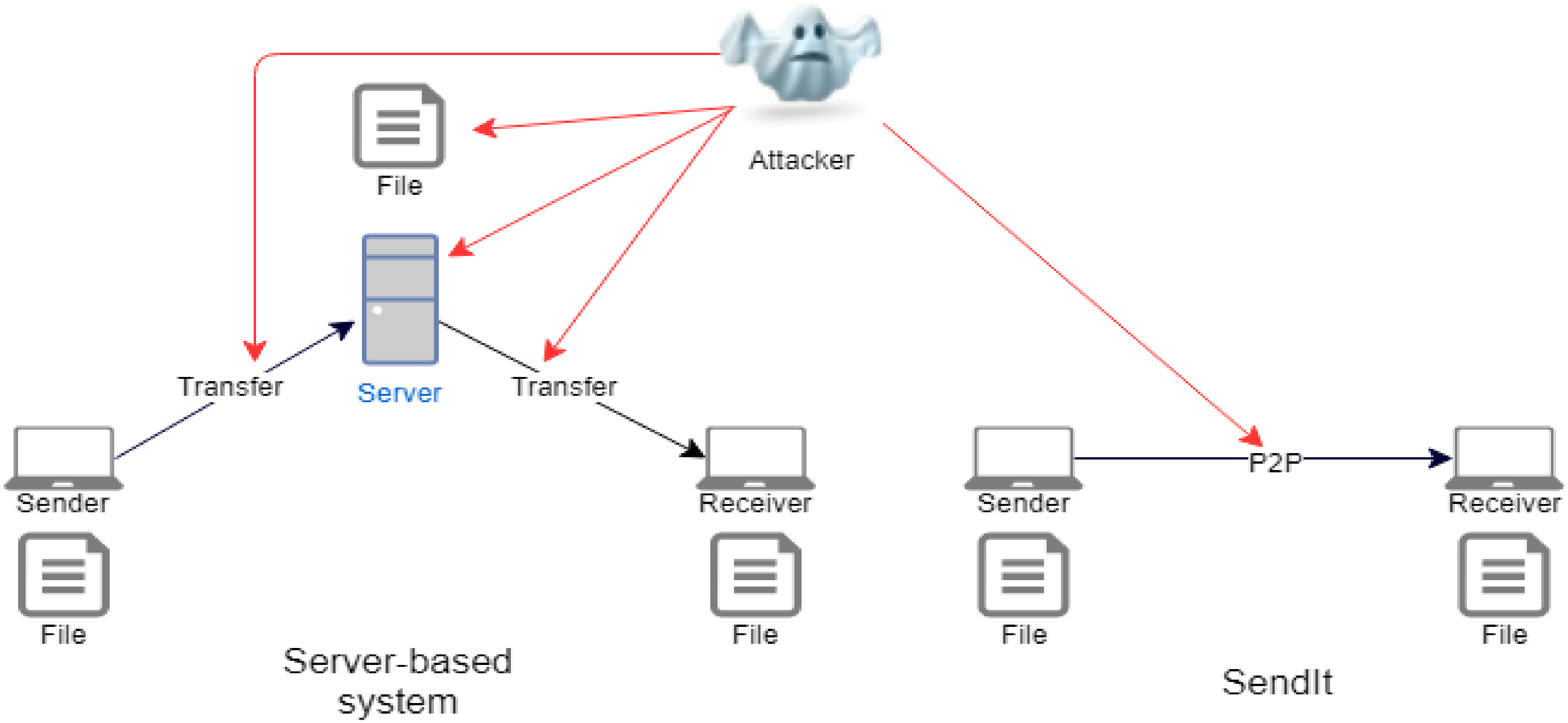
E-mail attachment issues

- Automatic spreading
- No way of stopping file
- No integrity guarantee
- Stored multiple locations
  - Server
  - PC
- Hard to notice
  - Impersonation
  - Hidden
- Common attack vector

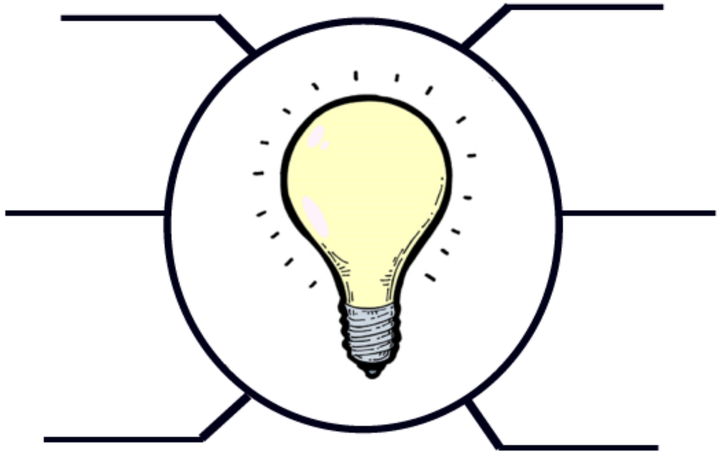
False end-to-end encryption



Threat comparison



Conclusion



Created a prototype (SendIt)

- Alternative to e-mail attachments
- Decentralized system
- Serverless
- Improvement to current e-mail system

Goals achieved:

✓Create a prototype to show feasibility.

Reduced attack surface

- Endpoint authentication
  - *Based on first trust*
- End-to-end encryption
- Direct communication
  - No temporary storage
- Continous trust evaluation

Goals achieved:

✓Improve security and ease of use for e-mail attachments.

✓Create a system with focus on usability, privacy and security.

✓Minimize risk of leakage and exposure of personal data.

Easy to use

- Automatic:
  - Key management
  - Trust system
  - Authentication
- No setup or sign-up required

Goals achieved:

✓Improve security and ease of use for e-mail attachments.

✓Create a system with focus on usability, privacy and security.

System Limitations

- First trust
- Synchronous
- Connection setup

Simple system focused on security

Improves security and ease of use for e-mail attachments!

References

- “About Electron | Electron.” Accessed March 26, 2018. /docs/tutorial/about.
- Alfarez, Abdul-Rahman. “The PGP Trust Model - Semantic Scholar.” Accessed March 26, 2018. /paper/The-PGP-Trust-Model-Abdul-Rahman/e9aa5d8032c1d925ea6a02dd3be93f42e831c965.
- Ball, Chris. *Serverless-Webrtc: A Demo of Using WebRTC with No Signaling Server*. JavaScript, 2018. https://github.com/cjb/serverless-webrtc.
- Bergkvist, Adam, Daniel C. Burnett, Cullen Jennings, Anant Narayanan, Bernard Aboba, and Taylor Brandstetter. “WebRTC 1.0: Real-Time Communication Between Browsers.” Accessed March 26, 2018. https://www.w3.org/TR/webrtc/.
- Camarillo, Gonzalo, Oscar Novo, and Simon Perreault. “Traversal Using Relays around NAT (TURN) Extension for IPv6.” Accessed March 26, 2018. https://tools.ietf.org/html/rfc6156.
- “Chrome | WebRTC.” Accessed March 26, 2018. https://webrtc.org/web-apis/chrome/.
- “Chrome V8.” Google Developers. Accessed March 26, 2018. https://developers.google.com/v8/.
- contributors, Mark Otto, Jacob Thornton, and Bootstrap. “Bootstrap.” Accessed March 26, 2018. https://getbootstrap.com/.
- Zimmermann, Philip R. *The Official PGP User’s Guide*. Cambridge, MA, USA: MIT Press, 1995.

- Douceur, John R. “The Sybil Attack.” In *Peer-to-Peer Systems*, 251–60. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2002. https://doi.org/10.1007/3-540-45748-8_24.
- “Firefox Test Pilot - Send.” Accessed March 24, 2018. https://testpilot.firefox.com/experiments/send/.
- Handley, Mark, Colin Perkins, and Van Jacobson. “SDP: Session Description Protocol.” Accessed March 26, 2018. https://tools.ietf.org/html/rfc4566.
- Holmberg, Hans-Christer. “Web Real-Time Data Transport.” Helsinki Metropolia University of Applied Sciences, 2015. https://www.theseus.fi/bitstream/handle/10024/94759/FinalThesis_hanschrh_final.pdf?sequence=1&isAllowed=y.
- “How PGP Works.” Introduction to Cryptography, 1999 1990. https://users.ece.cmu.edu/~adrian/630-f04/PGP-intro.html.
- “I2P-Bote.” Accessed March 24, 2018. https://i2pbote.xyz/.
- “I2P-Bote Introduction and Tutorial | Darknet Email.” The Tin Hat. Accessed March 24, 2018. https://thetinhat.com/tutorials/messaging/i2pbote.html.
- WHITTEN, ALMA, and J D TYGAR. “A Usability Evaluation of PGP 5.0,” n.d., 24. https://people.eecs.berkeley.edu/~tygar/papers/Why_Johnny_Cant_Encrypt/OReilly.pdf.

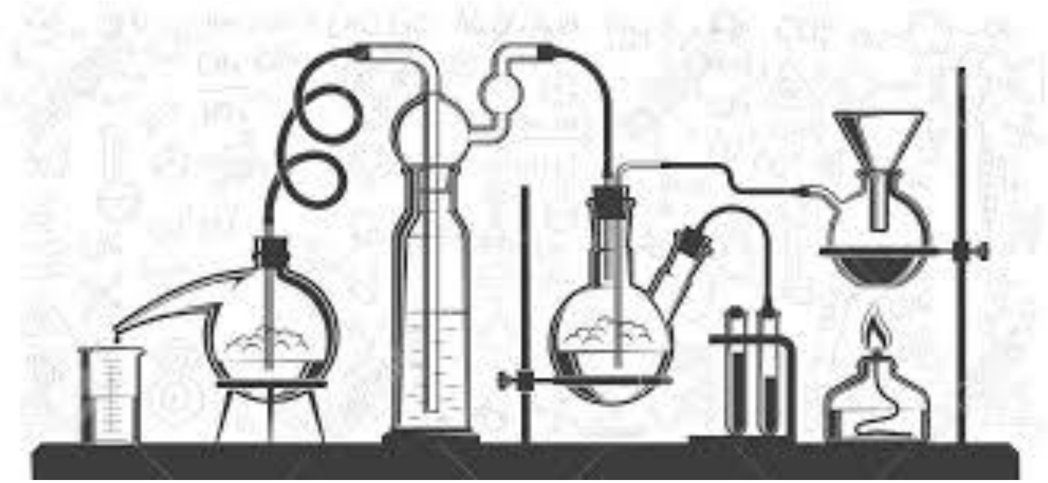
- Ibáñez, L. D., E. Simperl, F. Gandon, and H. Story. “Redecentralizing the Web with Distributed Ledgers.” *IEEE Intelligent Systems* 32, no. 1 (January 2017): 92–95. https://doi.org/10.1109/MIS.2017.18.
- “Index | Node.Js v7.10.1 Documentation.” Accessed March 26, 2018. https://nodejs.org/docs/latest-v7.x/api/.
- Ivov, Emil, Peter Saint-Andre, Eric Rescorla, and Justin Uberti. “Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol.” Accessed March 26, 2018. https://tools.ietf.org/html/draft-ietf-ice-trickle-17.
- Jesup, Randell, and Salvatore Loreto. “WebRTC Data Channels.” Accessed March 26, 2018. https://tools.ietf.org/html/draft-ietf-rtcweb-data-channel-13.
- Jøsang, Audun, Roslan Ismail, and Colin Boyd. “A Survey of Trust and Reputation Systems for Online Service Provision.” *Decision Support Systems* 43, no. 2 (March 2007): 618–44. https://doi.org/10.1016/j.dss.2005.05.019.
- Jøsang, Audun, and Simon Pope. “Semantic Constraints for Trust Transitivity.” *Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005* Vol. 43 (n.d.): 10.
- Watson, Mark. “Web Cryptography API.” Accessed March 26, 2018. https://www.w3.org/TR/WebCryptoAPI/.
- jquery.org, jQuery Foundation-. “JQuery.” Accessed March 26, 2018. https://jquery.com/.

- Kiesel, Joseph. “GDPR Compliance: Summary & Requirements You Need to Know.” *Linford & Company LLP* (blog), July 5, 2017. https://linfordco.com/blog/gdpr-compliance-requirements/.
- Kimmett, Taylor. *Rtc-Pubnub-Fileshare: A File Sharing Demo Built Using PubNub and WebRTC*. CSS, 2016. https://github.com/tskimmett/rtc-pubnub-fileshare.
- Li, Li, Wu Chou, Zhihong Qiu, and Tao Cai. “Who Is Calling Which Page on the Web?” *IEEE Internet Computing* 18, no. 6 (November 2014): 26–33. https://doi.org/10.1109/MIC.2014.105.
- Perrichon, Samuel. *Electron-Prompt: Electron Helper to Prompt for a String Value*. JavaScript, 2018. https://github.com/sperrichon/electron-prompt.
- 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), Pub. L. No. 32016R0679, 119 OJ L (2016). http://data.europa.eu/eli/reg/2016/679/oj/eng.
- Sheng, Steve, Levi Broderick, and Colleen Alison Koranda. “Why Johnny Still Can’t Encrypt: Evaluating the Usability of Email Encryption Software,” http://www.chariotsfire.com/pub/sheng-poster_abstract.pdf.
- Efail: Breaking S/MIME and OpenPGP Email Encryption using Exfiltration Channels, *Damian Poddebniak, et. Al., 27th USENIX Security Symposium*, Baltimore, August 2018.

- Rosenberg, Jonathan. “Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols.” Accessed March 26, 2018. https://tools.ietf.org/html/rfc5245.
- Rosenberg, Jonathan, Rohan Mahy, Christian Huitema, and Joel Weinberger. “STUN - Simple Traversal of UDP Through Network Address Translators.” Accessed March 26, 2018. https://tools.ietf.org/html/rfc3489.
- Ross (NIST), Author: Ron, Author: Kelley Dempsey (NIST), Author: Patrick Viscuso (NARA), Author: Mark Riddle (NARA), and Author: Gary Guissanie (IDA). “SP 800-171 Rev. 1, Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations.” Accessed March 30, 2018. https://csrc.nist.gov/publications/detail/sp/800-171/rev-1/final.
- Ross, Ron, Patrick Viscuso, Gary Guissanie, Kelley Dempsey, and Mark Riddle. “Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations.” Gaithersburg, MD: National Institute of Standards and Technology, December 2016. https://doi.org/10.6028/NIST.SP.800-171r1.
- Sorhus, Sindre. *Clipboardy: Access the System Clipboard (Copy/Paste)*. JavaScript, 2018. https://github.com/sindresorhus/clipboardy.
- “Web Crypto API.” MDN Web Docs. Accessed March 26, 2018. https://developer.mozilla.org/en-US/docs/Web/API/Web_Crypto_API.

Appendix

Experiments

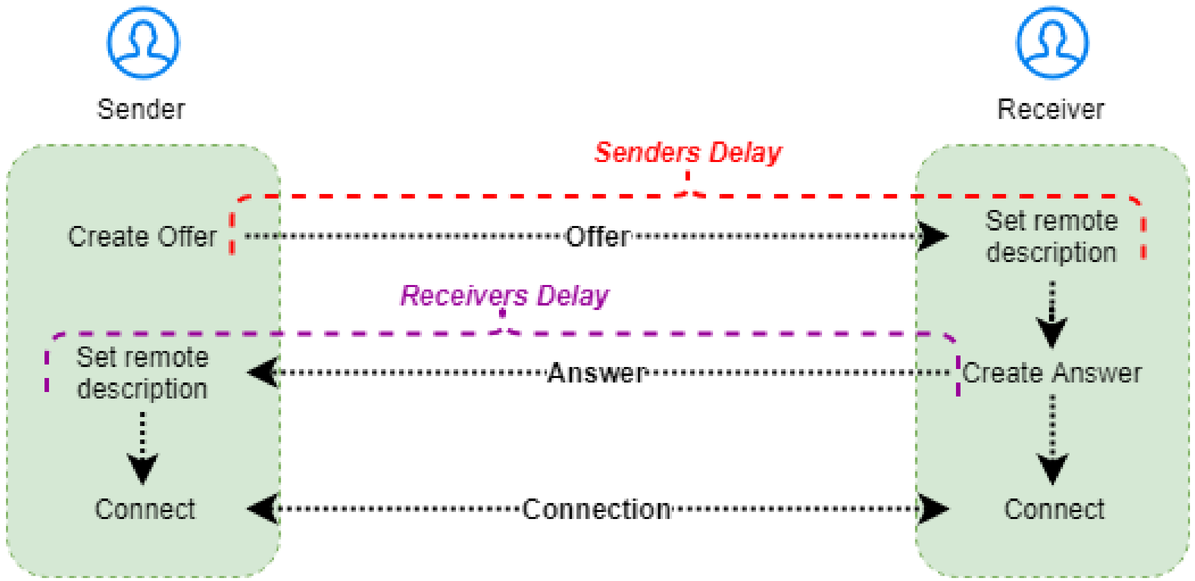


Goals

FIND:

- Average lifetime of Offer/Answer
- Most influential factor of the two
- Average lifetime of whole exchange

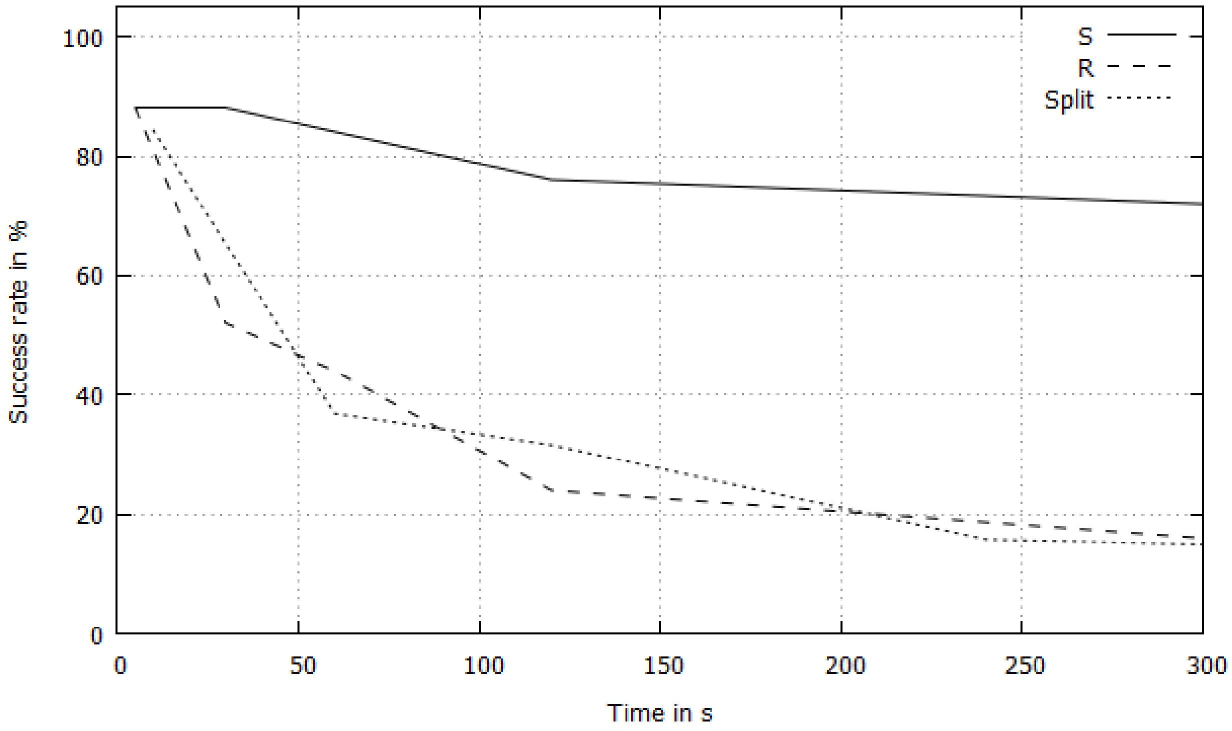
Terminology



Contribution to SendIt

- Which part is more time-constrained?
- How usable is the serverless mode?
- How to improve the serverless mode?

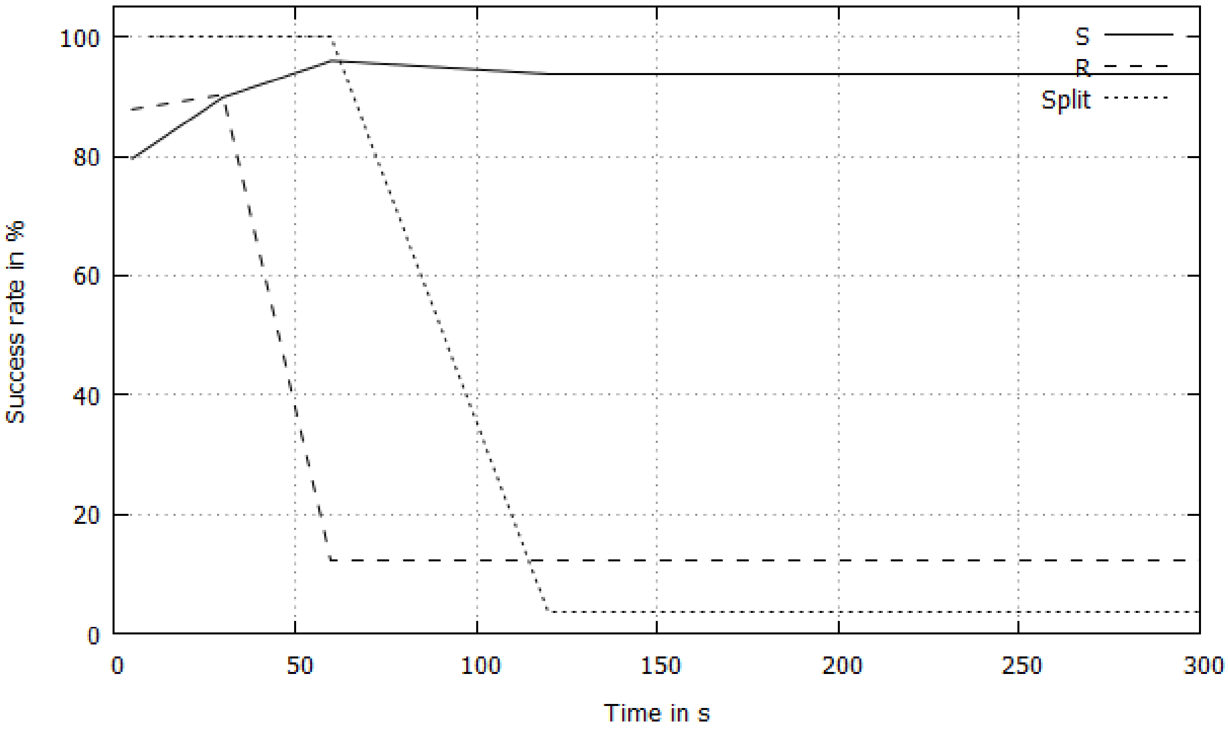
Result experiment 1



Method

- Server and Client tries to establish WebRTC connection
- Test with different delays when sharing offer and/or answer
    - Did experiment twice to verify results
- Simulates user behaviour

Result experiment 2



Results

- Both experiments gave similar results
- Only relevant for Serverless mode
- Offer: ~ 5 min
- Answer: ~1 min
- Total: ~6 minutes