Testing & Evaluation Sheet

Send

1. Tool Overview

Name:	Send
Category:	File-Sharing/Transfer, Peer-to-Peer File Sharing
Purpose:	Send enables secure, end-to-end encrypted file transfers, ensuring user privacy and anonymity.
Date	3/27/25
Status:	Deployed ☑ Operational - Actively running/maintained ☐ In Testing - Currently being evaluated or piloted ☐ Inactive/Deprecated - No longer maintained or functional
Deployment Architecture:	☐ A standalone software - Runs entirely locally (e.g., runs on computer and doesn't depend on external server) ☐ A locally hosted service with separate server and client component - Run both backend/frontend yourself (e.g., backend could be on a local network, or self-hosted on cloud) ☑ A service with a local client that's hosted by a third party - You install a client on your device, but it connects to and depends on a remote server (e.g., Signal: install app (client), but Signal's servers handle message relaying, etc.) ☑ A service that is hosted by a third party but can also be self-hosted
Version:	V3.4.23

2. Installation & Setup

OS Compatibility	Windows, macOS, Linux, Android, iOS
Installation Manual:	Yes
Installation Steps:	a. **Option 1:** Use the Hosted Version i. Simply go to **send.vis.ee** and start using Send instantly—no setup required!

<table>
<tr>
<td></td>
<td>
b. Option 2: Run Send Locally (And Deploy Later)
i. Clone the Repository (if you want to keep the original repository):
1. git clone https://github.com/<your-repo>/send.git
2. cd send
3. You can also paste github repository link into version control in preferred IDE
ii. Fork (if you want to create your own version of the repository):
1. Go to the Send GitHub page.
2. Click on the Fork button on the top right to make your own copy.
iii. Install Dependencies: npm install
iv. Build the Productions Assets: npm run build
v. To view this: npm run start
1. Running this will provide localhost and port on which app is running (this is to verify if the app is built).
vi. Start the Production Server: npm run prod
1. If doesn't load properly, rerun npm run build and then try again
vii. Access Send in Your Browser: Open http://localhost:1443 (or check your terminal for the actual port). You should see the full Send UI application.
1. Can find/change port at server/config.js
<pre>
listen_port: {
 format: 'port',
 default: 1443,
 arg: 'port',
 env: 'PORT'
},
</pre>
</td>
</tr>
<tr>
<td>Mention if command-line setup or special configurations are needed</td>
<td>
- Send is designed for ease of use with no command-line setup or special configuration required for end users.
- For developers or organizations self-hosting the service, basic server setup and configuration (e.g., Node.js, hosting environment, HTTPS) are needed.
</td>
</tr>
<tr>
<td>Common Installation Issues & Fixes:</td>
<td>Troubleshooting Guide: Refer to the official GitHub documentation.</td>
</tr>
</table>

User Documentation:	Yes
Required Technical Knowledge	Beginner

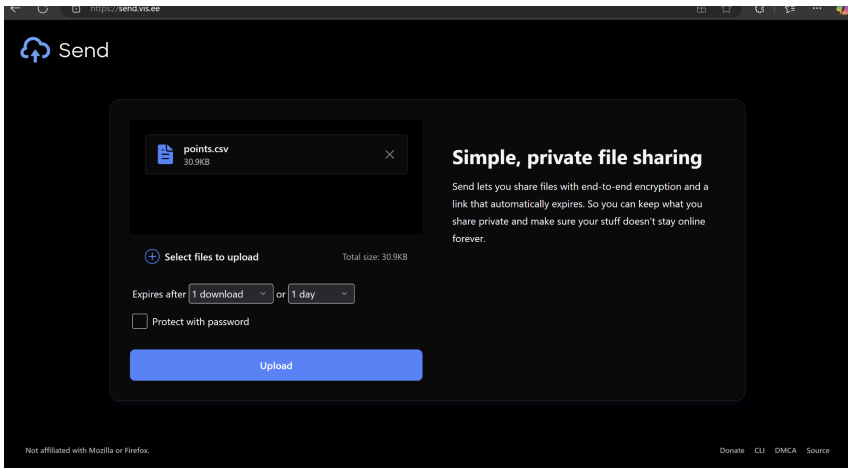
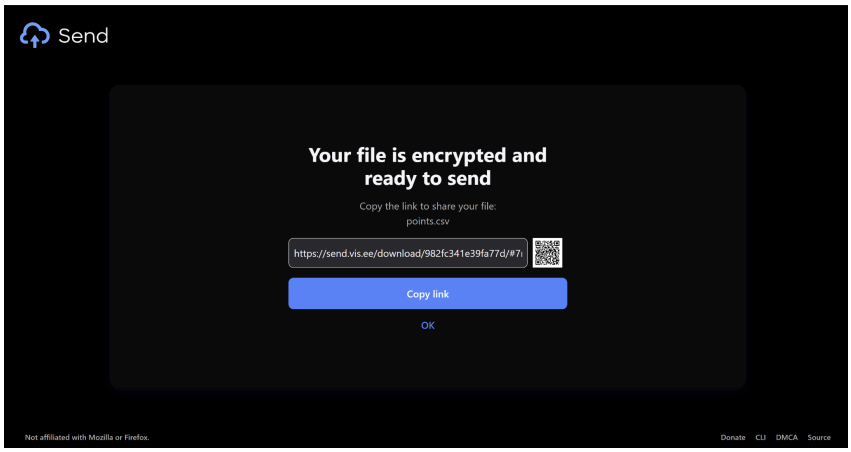
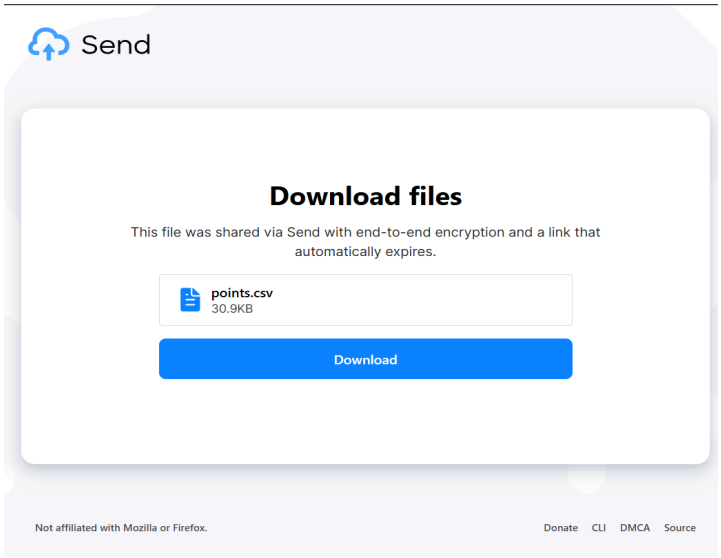
3. Testing & Evaluation

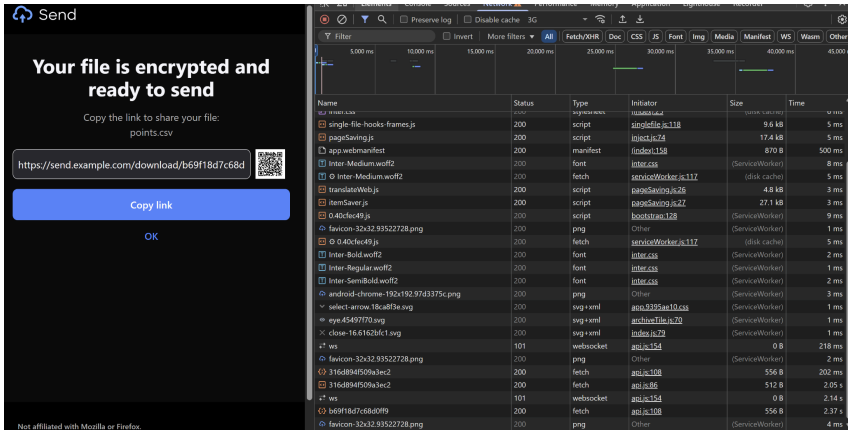
Category	Details	Score
Operational Functionality:	**Functionality** ● Yes, Send works as intended and is a forked version of the discontinued Mozilla Firefox Send(all branding removed so legally can self host) **Internet Dependence:** ● Does not have offline functionality. ● Functional on 3G and 4G networks (Slower downloads for larger files with lower generations) **Localization & Language Support** ● Over 71 languages including English and Simplified and Traditional Chinese ● Community is actively contributing to localization **Mobile Accessibility** ● It is mobile friendly ● Accessible through iOS and Android.	4.7
Usability for Non-Technical Users	**Ease of Installation & Deployment** ● No installation required for users. ● No setup guide needed **User Onboarding Experience** ● Not specific documentation for first time users but regular documentation is very straightforward **Technical Experience Level Required** ● Non-technical users can easily navigate the tool with its sleek user interface ● The interface is intuitive with simply instructions	3.3
Security & Privacy Strength	**Encryption Standards** ● End-to-End Encryption: Files are encrypted using **AES-256** encryption before being uploaded. This is a very strong encryption standard and is commonly used for securing sensitive data. ● Transport Layer Security (TLS): For secure communication between the client and server, **TLS** is used to prevent eavesdropping and ensure that data is transmitted securely over the network.	3.4

	Known Strength resilience ● If the hosting server is blocked, users may not be able to access it ● No built-in circumvention tools ● Network traffic may still be visible, even if file contents are not ● Known Threats: Like many self-hosted applications, Send could be vulnerable to attacks like: ○ Brute-force attacks on the decryption key if weak passwords are used. ○ Malware distribution if not monitored and maintained securely. ● Resilience: Send appears to be well-secured, but as with any self-hosted solution (instances), users need to follow best practices for server hardening and secure configurations to minimize vulnerabilities. **Comparison with Known Standards** ● Encryption: The use of **AES-256** encryption is a standard that is at least as secure, if not more so, than the encryption used by mainstream services like Google Drive. However, **Send** provides end-to-end encryption where the service itself cannot access the contents of your files, unlike Google Drive. ● Privacy: **Send** emphasizes privacy and anonymity. It does not require user accounts, and the server cannot decrypt your files, ensuring that your data remains private, unlike Google Drive or Dropbox, which are tied to user accounts and have the ability to access file contents. ● Features: Google Drive offers more advanced collaboration features (editing documents, sharing among groups, etc.), while **Send** is more focused on secure, one-off file sharing. **Data Minimization** ● Collects minimal/necessary data. ● Collects time stamp ● IP address collected for 24 hours **Privacy Policy Accessibility and Clarity** ● The privacy policy is minimal but Send is open-source so its code is fully transparent, which allows users to see how their data is handled at any time.	

Maintenance/Sustainability	**Community support** ● Since this is a forked version from the original project, there are less contributors than the original. However, there are still lots of contributors. ● The original repository had several contributors, with an active issue tracker and pull requests but now it is read-only. **Development active status** ● It is not frequently updated but ● The main developer is responsive to updates. **Funding and Sponsorship** ● No funding for this project. ● Can donate to the developer. ● There are no sponsorships. ● Software appears to be financially sustainable.	3.3
Performance / Effectiveness & Reliability	**Testing Environment Setup:** ● **Device:** Dell XPS 15 i9 processor ● 32 GB RAM ● **OS:** Windows ● **Network:** 4G **User Experience Observations** ● File sending is smooth **Speed & Responsiveness:** ● Send works well and should load efficiently as a web application. It is not as heavy as some large-scale file-sharing platforms like Google Drive or Dropbox but lower generation cellular technology downloads slower as seen in Figure 2. ○ 2.07 seconds for 30 kb file on 3G. **Resource Usage:** ● When sending/downloading a 88MB file (large), it used about 3% of CPU and about 10-40MB of Memory (Task Manager) **Network Performance:** ● Latency: Low in stable networks (~10-60 ms) ● Bandwidth: Send does **not throttle** speeds (limited only by browser, network, or server). **Encryption adds slight overhead**, ~1–3% extra, depending on chunking and metadata. Uses all bandwidth when possible; high bandwidth usage efficiency.. **Reliability** ● Firefox Send had received attention and contributions from the open-source community, suggesting a reasonable level of trust and reliability. However, this is a forked version to keep services running after	4.0

	Firefox Send was discontinued so it does not have as many contributions. ● Reliable as lots of people use Send and add changes to it.	
Deployment Considerations:	**Open Source & Transparency:** ● The code is open for independent verification and is somewhat transparent. **Cloud vs. Local Deployment:** ● There is a list of public instances (deployed versions) on Github that can be used instead of self hosting. ● To self host, you must need some blank service **Dependencies:** ● NodeJS and npm are required ● Apache webserver is required ● Git is required ● The dependencies are clearly documented **Post-Deployment Maintenance** ● After deployment, the tool is easy to maintain as it is not very complicated. ● If a deployer decides to fork the project, the code is relatively easy to modify in terms of UI changes. Other changes may be more difficult for the deployer. **Merge/Sustainability:** ● The original project is open to contributions. ● Is it relatively hard to submit changes back to the main repository as they aren't approved as often. ● May incentivize self-hosting and making changes to own code.	
4. Testing Scenarios		

Testing file transfer using Send	   *Figure 1*: The images above show the process in which a file is selected and encrypted from one computer/device to being downloadable on another with the URL or QR code.

Testing on different generations 3G/4G	 *Figure 2*: Testing file download of a 30KB file indicating 218 ms for fast 4G connection compared to 2.14 seconds for 3G connection.

5. Insights & Recommendations

Key Findings	**Strengths:** ● End to End Encryption: Files are encrypted in the browser before being uploaded, offering high privacy and security. ● Ease of Use: Simple user interface which makes it intuitive for users with varying technical skills. ● Self-hostable: Provides documentation to help self host as well. ● One-time Downloads: Files can auto-expire after download or time limit. **Weaknesses:** ● Self-hosting Technical Skills: It is not very easy for non-technical users to be able to self host as it requires Node.js/Nginx setup. ● Limited Upload Size on Public Instances: (e.g., send.vis.ee caps at 2 GB). ● No Built-In Search or File Management: Links are one-time use only, no retrieval after expiry.
Suggested Improvements	● A step-by-step **self-hosting guide** with screenshots (especially for beginners) and Docker setup examples ● Also can provide info on setting up instances for self hosting as well. ● Documentation: Step-by-step installation guides, tutorials for technical users
Alternative Tools:	Croc, Magic Wormhole, Lufi, FilePizza, NextCloud
License	MIT License

Cost/Resource Implications	**Total Cost of Ownership:** ● Self-hosted: ~$5-$10/month and maintenance time of about an hour. ● Public Instance: Free to use (ex. send.vis.ee) ● License: MPL-2.0 License ● No subscriptions needed unless hosted on cloud ● Automatic expiry of files keeps storage use low ● Updates minimal unless running a public instance
Why is this useful to civil societies in authoritarian environments?	*Explain why this tool would be useful to civil society organizations. Include if it is unavailable in certain regions due to firewalls or other blocking systems.* Send enables secure, anonymous file sharing with end-to-end encryption and no user accounts, which is vital for CSOs handling sensitive information. It helps protect activists, journalists, and NGOs from surveillance or interception. The ability to self-host gives organizations full autonomy, helping them avoid reliance on third-party servers or Big Tech infrastructure. This is especially important in countries with strict censorship or surveillance. In regions where the official Send service may be blocked by firewalls or government filters, self-hosted versions can bypass these restrictions, ensuring safe and continuous access to the tool. Its simple interface also makes it usable by non-technical users with minimal setup.