Testing & Evaluation Sheet

Croc

1. Tool Overview

Name:	Croc
Category:	Data Transfer, Peer-to-Peer File Sharing
Purpose:	Croc enables secure, end-to-end encrypted file transfers over the internet, ensuring user privacy and anonymity
Date Tested	4/7/2025
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☑ Operational - Actively running/maintained ☐ In Testing - Currently being evaluated or piloted ☐ Inactive/Deprecated - No longer maintained or functional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:	V10.2.2

2. Installation & Setup

OS Compatibility	Windows, macOS, Linux, Android, iOS
Installation Manual:	Yes
Installation Steps:	1. Find the Official Source a. Search for **"Croc File Transfer"** or visit the official GitHub page. 2. Follow OS-Specific Installation Instructions

	a. The **README** on GitHub provides installation steps for Windows, macOS, Linux, and Android. b. Run the installation using an administrator terminal (some systems require elevated permissions). 3. Accept the Download a. If prompted, type **"y"** or **"a"** to confirm the installation. 4. Command-Line Usage a. Croc operates entirely from the command-line interface (CLI). 5. After installation, test it by running: **a. croc --help**
Mention if command-line setup or special configurations are needed	Entire tool is a command-line tool which can be intimidating but does not require special configurations. It is relatively easy to use despite using the command line as the interface for file transfer.
Common Installation Issues & Fixes:	1. Installation fails in a regular terminal → Run the command in an administrator terminal (PowerShell, sudo, etc.). 2. 'croc' not recognized → Ensure Croc is in your system PATH or restart your terminal. 3. Permission denied → Use chmod +x croc (Linux/macOS) or run as Administrator (Windows). 4. Connection issues → Check your internet connection and firewall settings. 5. Further troubleshooting → Refer to the official GitHub documentation.
User Documentation:	Yes
Required Technical Knowledge	Intermediate

3. Testing & Evaluation

Category	Details	Score
Operational Functionality:	**Functionality** ● Test Steps: Verify the tool's core features by using all major functions, tracking any failures or bugs. ☐ The tool is mostly non-functional with many broken features and bugs.	2.3

	☐ Several broken features or bugs ☐ Minor bugs or issues ☐ Mostly functional with few bugs or no bugs ☑ Fully functional with no bugs **Internet Dependence:** ● No offline functionality the tool needs to connect to servers in order to transfer files. **Localization & Language Support** ● Croc primarily supports English (supports some others such as Spanish, French, German, Italian, through contributor localization but not officially). ● No East Asian Languages supported ● They have an active community contributing to localization **Mobile Accessibility** ● Not mobile friendly needs to be operated on a computer system and received on another computer.	
Usability for Non-Technical Users	**Ease of Installation & Deployment** ● About 5 steps for installation ● Very visual and in depth set up guide is available ● < 5 minutes installation time **User Onboarding Experience** ● Yes there is documentation for first time users on how to send and receive files. Along with visual aids on the github. **Technical Experience Level Required** ● Users with no programming background can install and start using the tool ● Relies on terminal commands	4.3
Security & Privacy Strength	**Encryption Standards** ● Croc uses Password Authenticated Key Exchange (PAKE) to establish a strong session key from a shared code phrase, ensuring end-to-end encryption during file transfers. ● Croc uses proxy servers, including compatibility with Tor, helping bypass government censorship. **Known Strength resilience** ● Since croc uses a central relay server (default: relay.croc.pm), it can be blocked if authorities detect or blacklist the relay server's IP/domain. ● No built-in circumvention tools **Comparison with Known Standards**	4.2

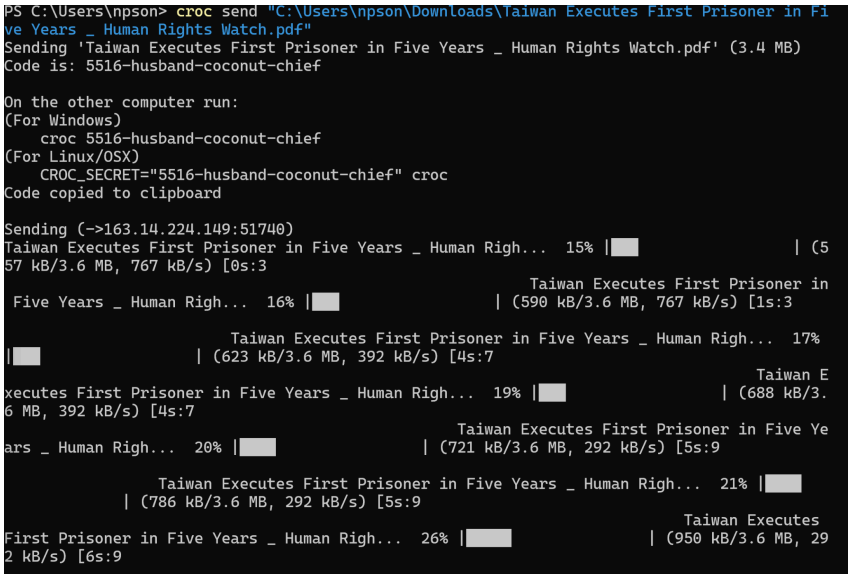
	● This application attempts to reimplement magic-wormhole with additional features **Data Minimization** ● There is no explicit information about whether Croc collects unnecessary user data. However, the absence of a detailed privacy policy makes it unclear what data, if any, is collected. **Privacy Policy Accessibility and Clarity** ● There is no detailed privacy policy which makes it unclear what data, if any, is collected.	
Maintenance/Sustainability	**Community support** ● The community is a decent size with over 100 contributors, but does not seem particularly active. ● Questions can easily be asked through the GitHub **Development active status** ● Updated within the last week: March 31st ● Pretty responsive with the latest commits occurring at least a couple times a month **Funding and Sponsorship** ● Funded by Zack Shollz (Software Engineer + Scientist & owner of project) ● Not government backed which impacts the level of security (vulnerabilities more common due to lower resources. ● There are about 20 monthly sponsors	3.3
Performance / Effectiveness & Reliability	**Testing Environment Setup:** ● **Device:** Dell XPS 15 i9 processor, 32 GB RAM ● **OS:** Windows ● **Network:** 4G **User Experience Observations** ● Minor load time for sending files ● Slight delay of response when using the computer terminal to send files **Speed & Responsiveness:** ● croc is a CLI tool, so its binary loads nearly instantly (usually < 50ms). ● File transfer setup and connection handshake are also very fast, typically within 100–300ms depending on network latency. **Resource Usage:** ● CPU: Very low usage. During idle or connection phases, usage is ~1–2%.	4.5

	● During active transfer, CPU usage might briefly spike to ~5–10%, especially for encryption/decryption. ● Memory usage stays low (~15–50MB). ● No memory leaks or spikes were observed in long transfers. **Network Performance:** ● Initial handshake: ~150ms average ● File transfer (100MB): ~5–7s over LAN, 15–25s over Wi-F ● Latency: No additional latency and performance is mostly determined by your connection to the relay or the peer. ● Uses relay servers efficiently if P2P fails. ● Transfers files at full available bandwidth when possible. ● Does not leak metadata (like file names) in packet inspection. ● Can handle large files (multi-GB) without crashes. ● Multiple simultaneous transfers work well. **Reliability** ● Over 29.7k stars on GitHub and 1.2k forks at the time of testing. ● Actively maintained and issues are regularly reviewed and patched quickly by the maintainer. ● No publicly available independent peer reviews or third-party security audits	
Deployment Considerations:	**Open Source & Transparency:** ● Yes **Cloud vs. Local Deployment:** ● Croc is local and does **not require** cloud platforms **Dependencies:** ● Croc is written in Go and compiles into a single binary so does not rely on Docker, Python, or databases **Post-Deployment Maintenance** ● Yes, Croc's single-binary nature simplifies deployment and maintenance. Updates involve replacing the binary with the latest version. ● Yes, the codebase is structured for clarity, and the use of Go ensures cross-platform compatibility. Developers familiar with Go should find it straightforward to modify.	

	Merge/Sustainability: ● The project is open to contributions, with an active community and a history of accepting pull requests and addressing issues. ● Yes, it is easy for contributors to fork the repository, make modifications, and submit pull requests.	

4. Testing Scenarios

How To Use File Transfer (Basic):



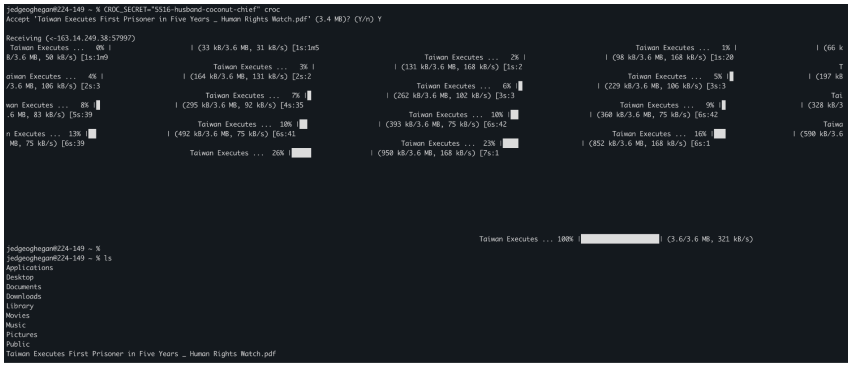


Figure 1: The two images above are from two different laptops showing how the file was sent to the other laptop using Croc.

5. Insights & Recommendations

Key Findings	**Strengths:**

	● Key benefits of the tool, such as ease of use, security features, etc. ● Optimizes file transfers using a relay server for full-duplex communication, ensuring data flows simultaneously rather than sequentially. ● Supports sending and receiving multiple files in a single session. ● Uses PAKE (Password Authenticated Key Exchange) for end-to-end encryption **Weaknesses:** ● Address major drawbacks, such as poor usability for non-technical users, etc. ● Croc is a CLI-only tool and does not work in web browsers(No use through Tor to increase security of file transfer) ● While Croc supports NAT traversal to some extent, it lacks more advanced or varied options like WebRTC or ICE-based STUN, which can better penetrate firewalls and complex NAT scenarios. ● Using the public relay server could expose metadata (e.g., IP addresses, timing information) to the relay operator
Suggested Improvements	● Step-by-step installation guides, tutorials for technical users ● Very simple installation guide along with visual video on how to use the tool
Alternative Tools:	● Magic Wormhole ● Send
License	MIT License
Cost/Resource Implications	**Total Cost of Ownership:** ● Limited knowledge on the funding and cost behind Croc ● There is no subscription necessary to keep Croc maintained
Why is this useful to civil societies in authoritarian environments?	- **Cross Platform and Peer-to-Peer:** An individual from one NGO can send a file to another NGO in a different country easily with peer-to-peer transfer. Also supports multiple platforms, making it useful across a variety of devices and operating systems. - **Privacy:** Croc does not leak any Metadata which is ideal for whistleblowers or individuals sharing data in repressive environments - **Secure File Transfer:** Since there is no need for VPNs, servers, or cloud accounts and Croc has End to End

	Encryption, it is useful for sharing sensitive documents, witness reports, or other legal or important evidence. - **Bulk Transfer:** Can be run from command line so it can be automated through scripts for secure bulk transfer. - **Ease of Use:** Command line transfer allows for fast transfer of files during protests, humanitarian crises, or censorship. - **Bypass Censorship:** Able to be used in countries with strict censorship and surveillance.
Notes	● While Croc uses a relay server to facilitate connections when direct peer-to-peer isn't possible, the relay server does not store any information, and all data passing through it is end-to-end encrypted.