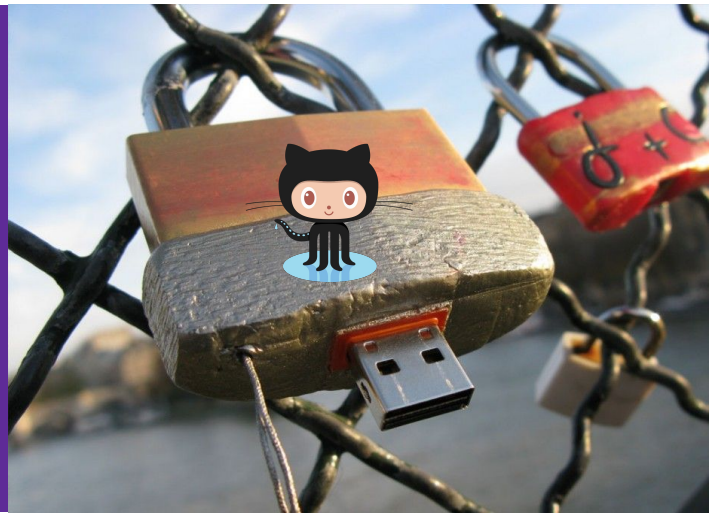
GISTER: Secure Dead Drop Using GitHub

Abusing Sharing Service for Encrypted Comms



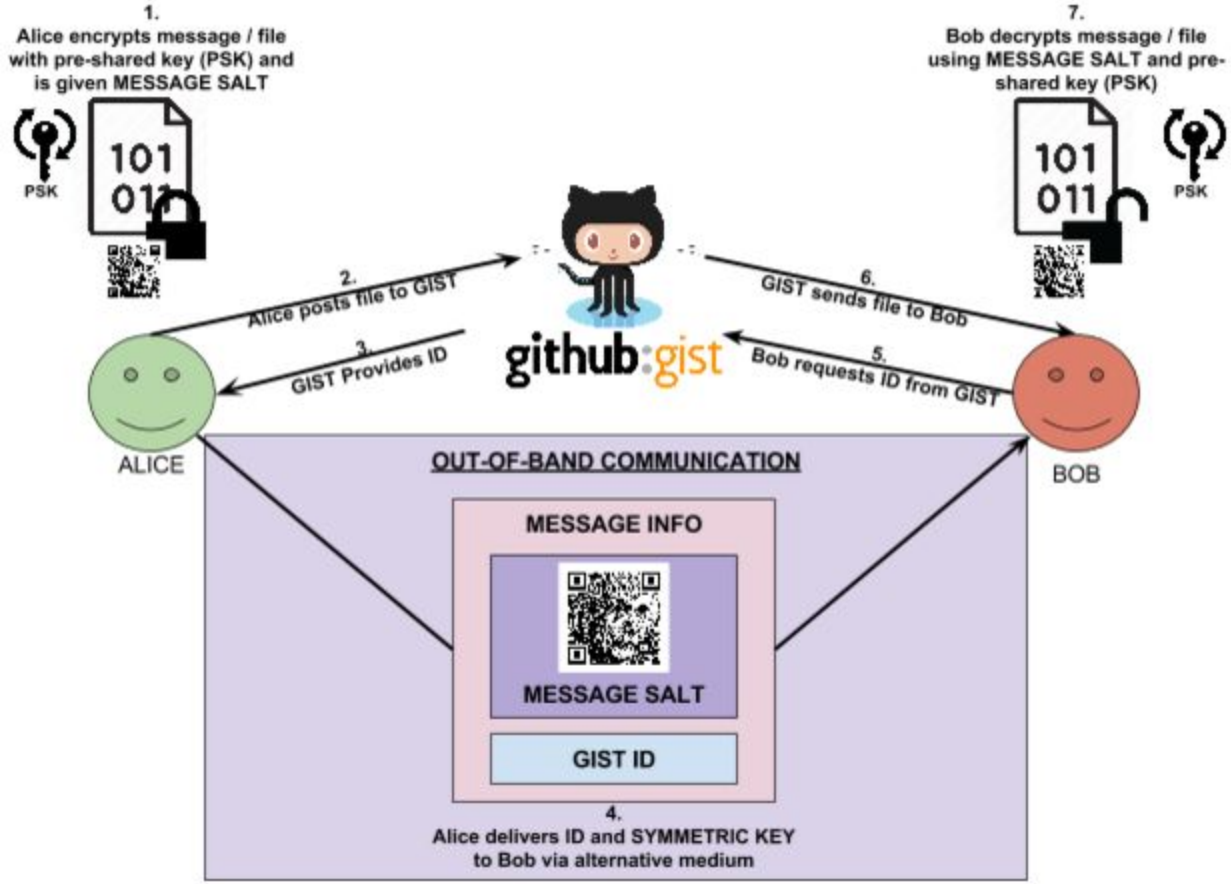
CS6903 - NYU - Fall 2015 ; Project 2
Andre Protas (ADP369) & Nate Rogers (NJR5)

Introduction & Purpose

Introduction & Purpose

*Use free **dead drop** website to create a **simple** and **secure** method of transmitting messages between two users **anonymously***

Implementation



Implementation Diagram

Implementation Details - Gist.GitHub.com

- GitHub Gist was perfect
- Effectively pastebin over SSL on GITHUB.com servers
- Many great “features” that lend itself to this project:
  - Well-documented REST API(simple to prototype and interact with)
  - Does not require account creation (supports anonymity)
  - Uses SSL for network security (we verify SSL chain)
  - Allows for large file sizes (up to 10mb)
  - Files can be easily retrieved by any user (dead drop)
  - Files cannot be deleted by others (preserves integrity)
  - Posting base64 files to GIST service is normal practice (hide in the noise)
  - Reputable API.GITHUB.com domain (not abnormal network traffic)
  - Revision history shows if files are manipulated (preserves integrity)

Implementation Details - Code

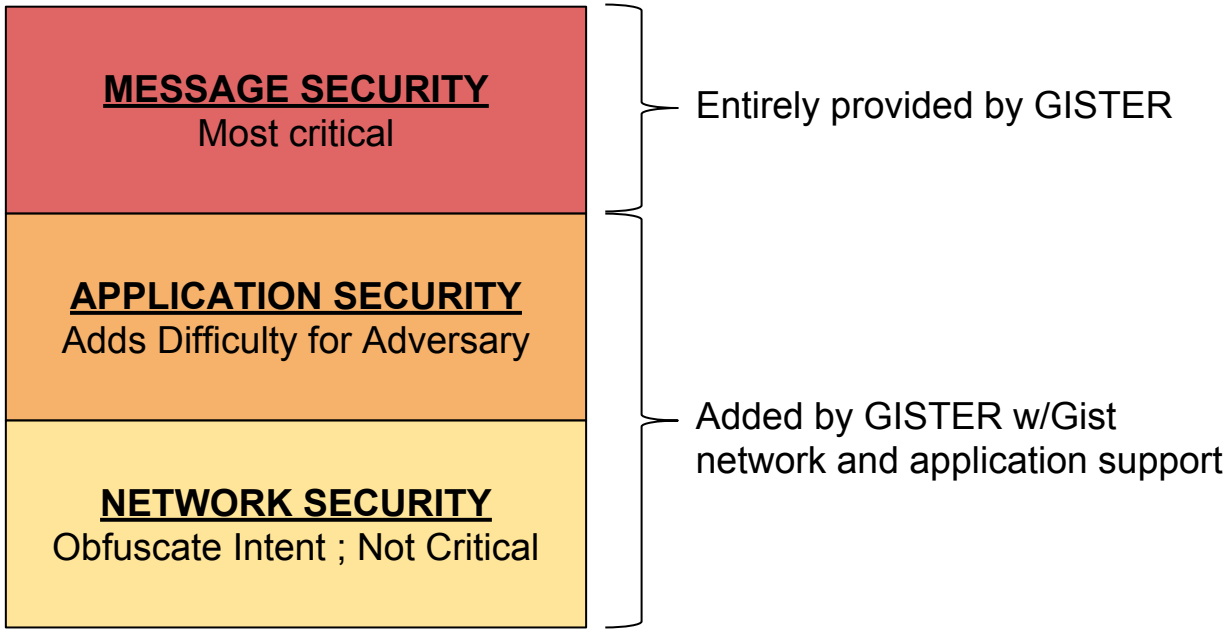
- Written entirely in Python
- Only 3 non-native libraries used
  - pyaes
  - requests
  - pyqrcode
- Supports multiple platforms (“just worked” on Ubuntu and Windows)
- Utilizes best possible OS PRNG subroutines automatically (SystemRandom)
- Write a set of unit tests to test all components of the system

Security & Threat Modeling

Threat Modeling

- We assume adversary has access to code
  - No “security through obscurity”
- We assume out-of-band communication is not known to adversary
  - Out of scope for GISTER
  - Recomendation: face-to-face pre-shared password, Message ID / Salt transmitted using different, rotating comms
  - Critical for dead drop scenarios
- We assume adversary may get access to GitHub traffic / logs
  - We don’t “trust” GitHub
- We assume adversary may have network interception capability
  - We attempt to force secure comms, but even if captured by outside adversary (e.g., with GitHub collaboration), our security is preserved
- 3 implemented layers of security: ***Message***, ***Application***, and ***Network***

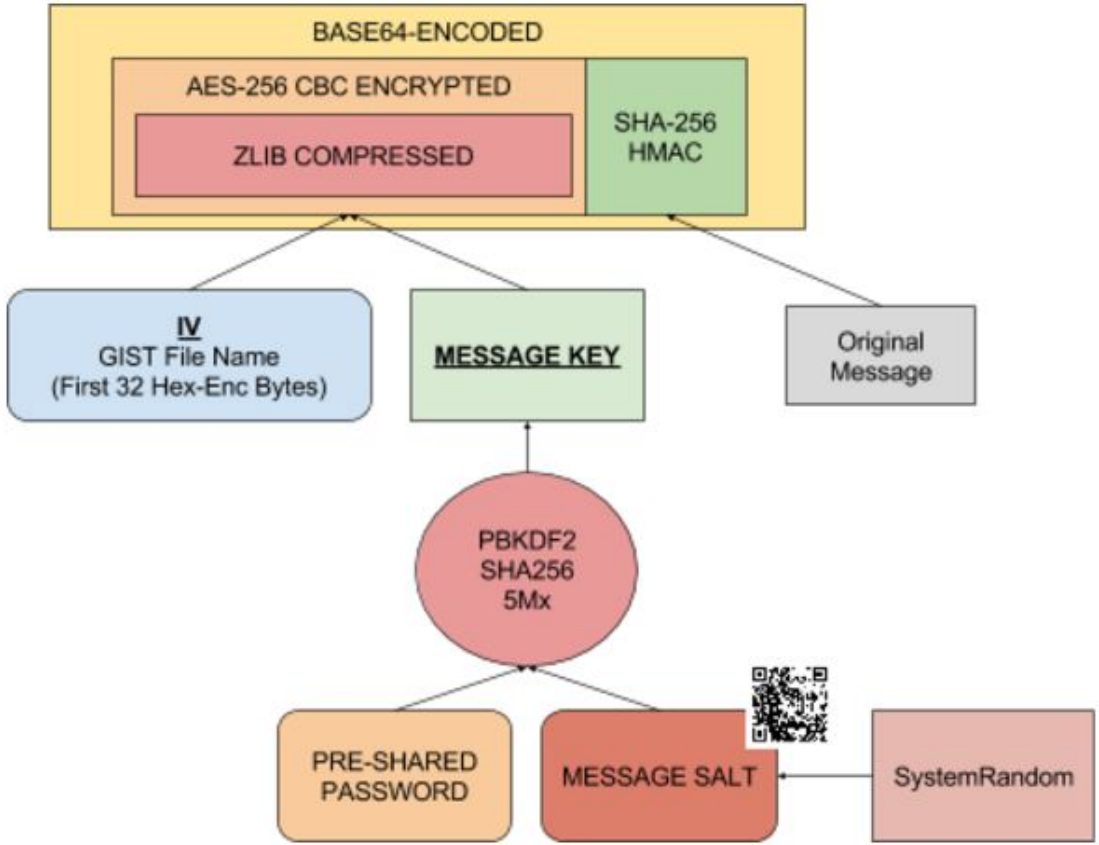
Security Layers



Message Security

Message Security

- Most focused-on component
  - Assumes other layers offer complications for adversaries, but Message Security is paramount to overall security
- AES-256 in CBC mode
  - IV provided in the clear as part of Gist metadata
- Uses a key derived using Password-Based Key Derivation Function 2 (PBKDF2) with:
  - Pre-shared password as the password
  - Randomly generated salt for each message
    - Salt is transmitted to recipient out-of-band (QR code provided)
- Integrity verified using SHA-256 HMAC after decryption
  - Use defenses against timing-based-attacks native to python libraries

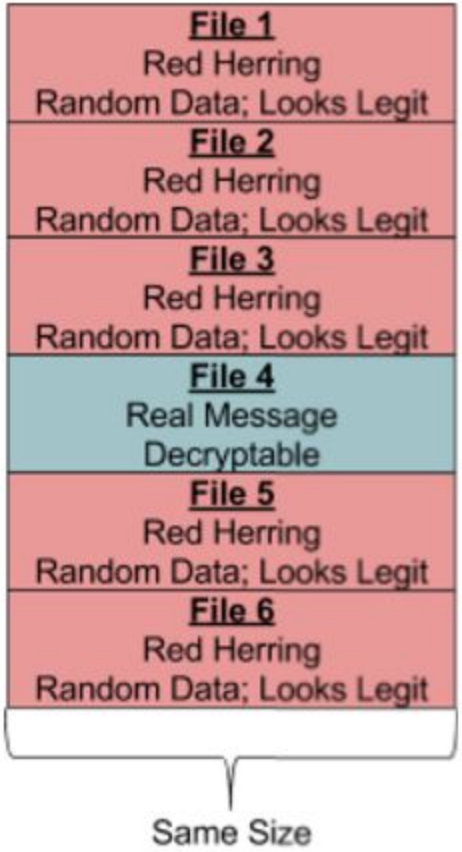


Message Security

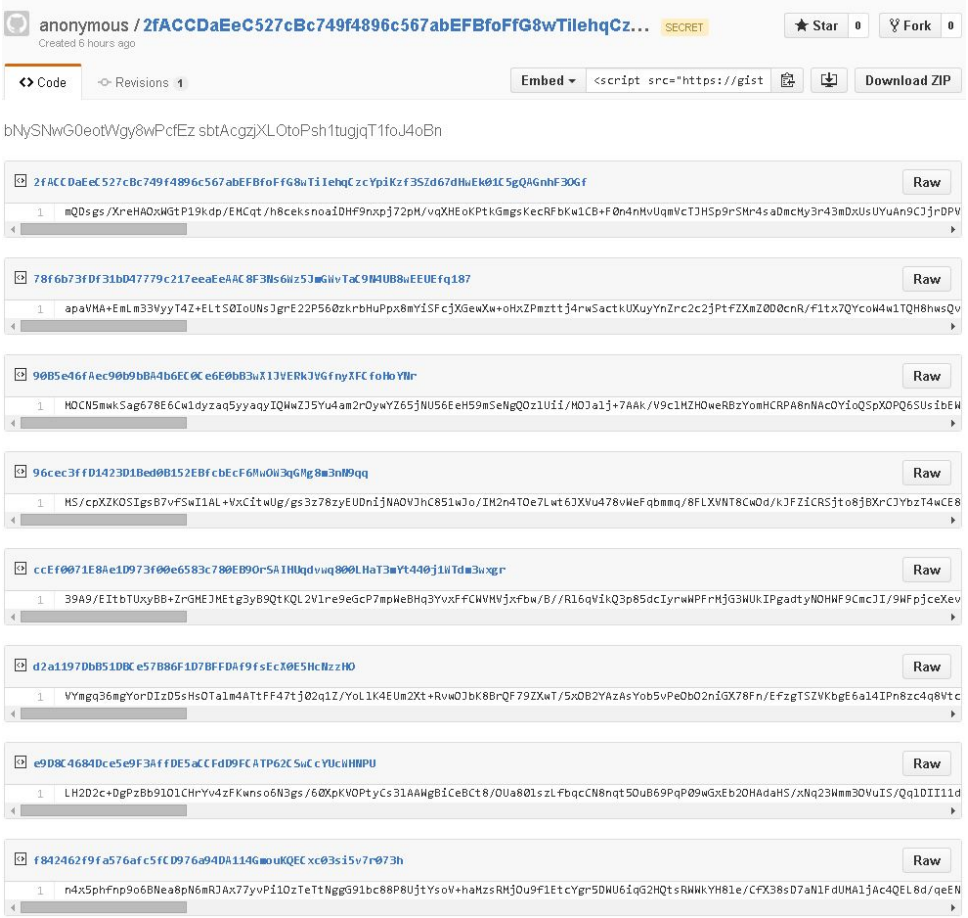
Application Security

Application Security

- We focus on using Gist API functionality to:
  - Avoid searchability for an adversary (randomize attributes)
  - Add extraneous data that requires additional power to decrypt
- Randomize all of our metadata and attributes
  - Would require an actor to find all Gist ID's with Base64 data for all messages
    - Not a possible query w/o GitHub collaboration
  - Gist ID already randomized by API
- Modification Detection
  - Anonymous nature of Gist allows for modification by third party
  - We only use the original post, and alert the user if modifications have been made later
- Red Herring Support
  - We can create any number of IDs or additional files as part of our message
  - We add additional files to our Gist ID's to increase the processing time needed to break encryption



Red Herring Depiction


anonymous / 2fACCDaEeC527cBc749f4896c567abEFBfoFfG8wTilehqCz...
SECRET
Created 6 hours ago
Star 0
Fork 0
Code
Revisions 1
Embed
<script src="https://gist
Download ZIP
bNySNwG0eotWgy8wPcfEz sbtAcgzjXLOtoPsh1tugjqT1foJ4oBn
2fACCDaEeC527cBc749f4896c567abEFBfoFfG8wTiIehqCzcYpiKzf3SZd67dHwEk01C5gQAGnhF3OGf
Raw
1 mQDsgs/XreHAOxWGtP19kdp/EMCqt/h8ceksnoaIDHf9nxpj72pM/vqXHEoKPtkGmgsKecRFbKw1CB+F0n4nMvUqmVcTJHSp9rSMr4saDmcMy3r43mDxUsUYuAn9CJjrDPV
78f6b73fDf31bD47779c217eeaEeAAC8F3Ns6Wz5JwGWvTaC9N4UB8wEEUEfq187
Raw
1 apaVMA+EmLm33VyyT4Z+ELtS0IoUNsJgrE22P56QzkrbHuPpx8mYiSFcjXGewXw+oHxZPmzttj4rwSactkUXuyYnZrc2c2jPtfZXmZ0D0cnR/f1tx7QYcoW4w1TQH8hwsQv
90B5e46fAec90b9bBA4b6EC0Ce6E0bB3wX1JVERkJVGfnyXFCfoHoYNr
Raw
1 MOCN5mwkSag678E6Cw1dyzaq5yyaqyIQWwZJ5Yu4am2rOywYZ65jNU56EeH59mSeNgQOzlUii/MOJalj+7AAk/V9clMZHOweRBzYomHCRPA8nNAcOYioQSpXOPQ6SUsibEW
96cec3ffD1423D1Bed0B152EBfcbEcF6MwOW3qGMg8m3nM9qq
Raw
1 MS/cpXZKOSIgsB7vfSwI1AL+VxCitwUg/gs3z78zyEUDnijNAOVJhC851wJo/IM2n4TOe7Lwt6JXVu478vWeFqbmmq/8FLXVNT8CwOd/kJFZiCRSjto8jBXrCJYbzT4wCE8
ccEf0071E8Ae1D973f00e6583c780EB9OrSAIHUqdvwq800LHaT3mYt440j1WTdm3wxgr
Raw
1 39A9/EItbTUxyBB+ZrGMEJMEtg3yB9QtKQL2Vlre9eGcP7mpWeBHq3YvxFfCWVMVjxfbw/B//Rl6qVikQ3p85dcIyrwWPFrMjG3WUkIPgadtyNOHWF9CmcJI/9WFpjceXev
d2a1197DbB51DBCe57B86F1D7BFFDAf9fsEcX0E5HcNzzHO
Raw
1 VYmgq36mgYorDIzD5sHsOTalm4ATtFF47tj02q1I/YoL1K4EUm2Xt+RvwOJbK8BrQF79ZXwT/5xOB2YAzAsYob5vPeObO2niGX78Fn/EfzgTSZVKbgE6al4IPn8zc4q8Vtc
e9D8C4684Dce5e9F3AffDE5aCCFdD9FCATP62CSwCcYUcWHNPU
Raw
1 LH2D2c+DgPzBb91OlCHrYv4zFKwnso6N3gs/60XpKVOPtyCs31AAWgBiCeBCt8/OUa801szLfbqcCN8nqt5OuB69PqP09wGxEb2OHAdaHS/xNq23Wmm3OVuIS/QqlDII11d
f842462f9fa576afc5fCD976a94DA114GmouKQECxc03si5v7r073h
Raw
1 n4x5phfnp9o6BNea8pN6mRJAx77yvPi1OzTeTtNggG91bc88P8UjtYsoV+haMzsRMjOu9f1EtcYgr5DWU6iqG2HQtsRWWkYH81e/CfX38sD7aN1FdUMA1jAc4QEL8d/qeEN
Publicly Accessible

Network Security

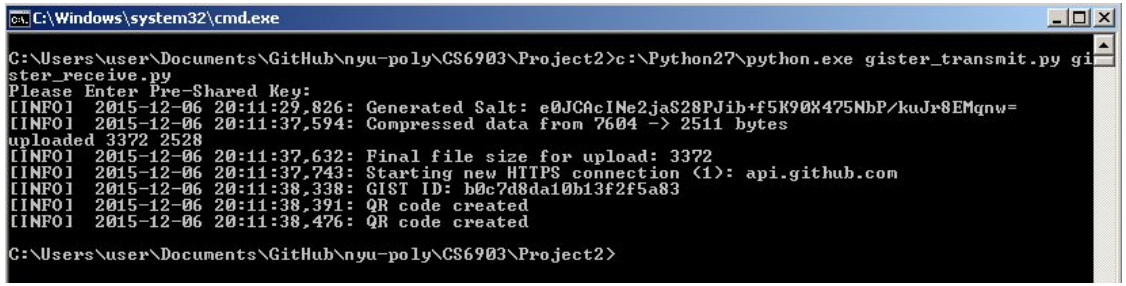
Network Security

- Focus on SSL
- SSL chain certified to Digi-Cert CA
  - We carry our own CA_BUNDLE with our src
  - Avoid extremely powerful adversary that may have ability to sign as other CA
- Any SSL failure is a fail-closed execution
- Only traffic is to https://API.GITHUB.com
  - Very common DNS entry to communicate with
- Support network proxy
  - E.g., post or receive messages using TOR or another method
  - Better to do this via secure environment, such as WHONIX
- With all of this, we still assume an actor can capture our network traffic
  - E.g., collaboration with GitHub
  - That’s why Application and Message security is so critical

User Simplicity & Demo

User-Simplicity

- The code should be as easy to use as possible
- Sending data should be fast (<5 minutes) for even large files (10mb)
- Retrieval of data should be easy
- We want to simplify ability to transmit out-of-band information quickly and easily
    - QR code added as a simple piece of data to transmit following a message upload



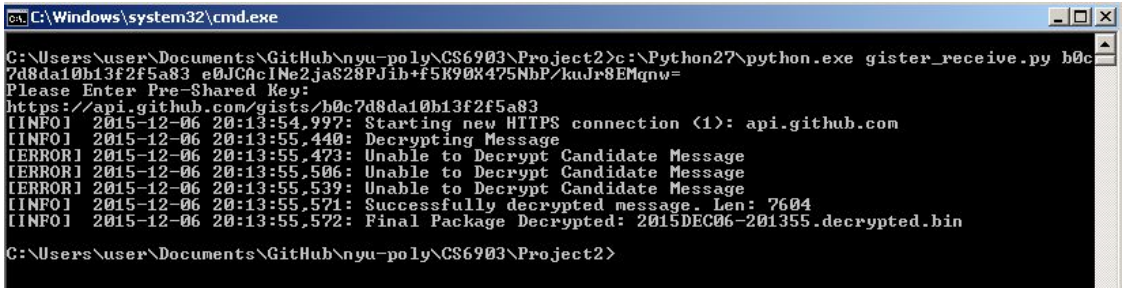
MESSAGE ID:

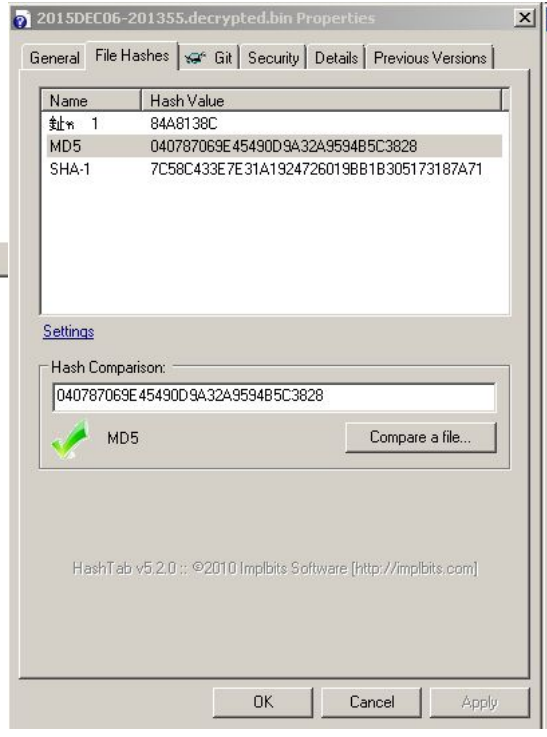
b0c7d8da10b13f2f5a83

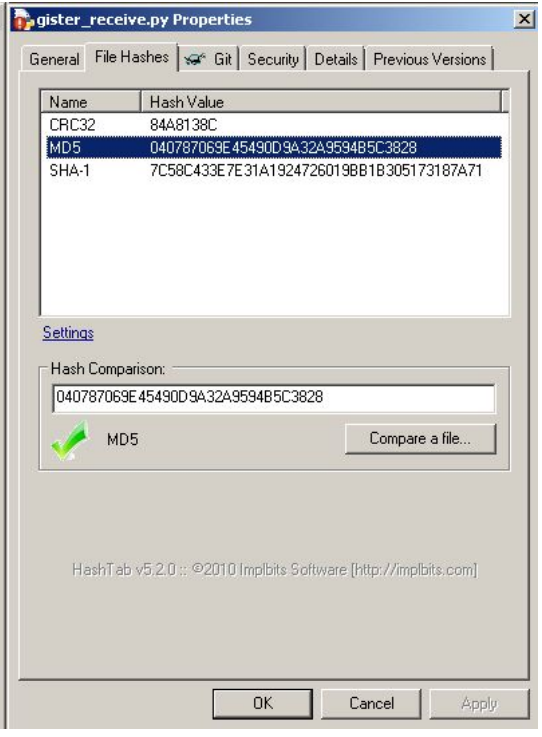
MESSAGE SALT:

e0JCAcINe2jaS28PJib+f5K90X475NbP/kuJr8EMqnw=

Message Transmission







Message Receiving

Considerations & Potential Improvements

Improvements

- Using a GIST ID or Account for a long-running conversation
  - Continuous comms over one channel
  - Easily identifiable, but can be obfuscated
- Transmit the next MESSAGE SALT in each transmission
  - Hard to manage multiple recipients. We decided on atomic message transactions.
- Protect receiving IP address with multiple requestors
  - Create “noise” of other “users” accessing the Gist ID to avoid revealing real intended recipient IP
- Expiration for messages
  - Using Gist would require creating accounts for transmissions
  - Can be done w/
- Increased number of red herring messages
  - Increase brute force computing requirements

Conclusion

Conclusion

- We believe GISTER is an effective system for communicating anonymously through GitHub Gist
- We believe that even against a powerful adversary with widespread reach and collaboration, GISTER would be difficult to defeat
    - Focus would likely go another route, such as trojanizing GISTER operating systems instead of trying to break the encryption itself
- We believe this system could be used for most dead drop scenarios, such as:
    - Covert communication
    - Command & Control communication (e.g., malware)
    - Long-term storage of sensitive data