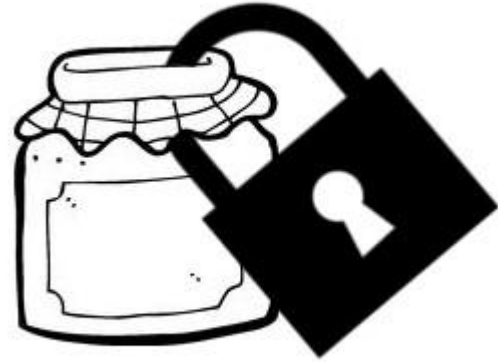


Jammed



Preserve your passwords

Secure Online Password Manager

Team Members

John Hohm

Marcos Pedreiro

Megan Carpenter

Dan Etter

System Purpose

Securely store login information on a remote server

Allow users to login and view their information

Allow users to modify/update their information

Prevent malicious entities from accessing users data

Threat Analysis

Assumptions

- Physical server safety

- Server Admins are benevolent

- Memory dumps are out of scope

Adversaries

- Outside Attacker

- Normal User

Threat Analysis

Outside Attacker

Motivation: Gain access to users' stored data

Capabilities: Monitor and intercept network traffic

Normal User

Motivation: Gain access to other users' stored data

Capabilities: Interact through normal client, create
malicious client

Security Goals

Ensure Confidentiality of sensitive user information

Maintain integrity between server and client processes

Authenticate users who present valid login credentials

Audit actions made on the server

Authorize valid actions made by authenticated users

Security Goals

A user may only view their information

A user is only authorized to view information if they have:

- Successfully authenticated with a master password

- Have access to their unique decryption key

An adversary may not gain access to users data

An adversary may not violate the integrity of the connection between the server and client processes

Design

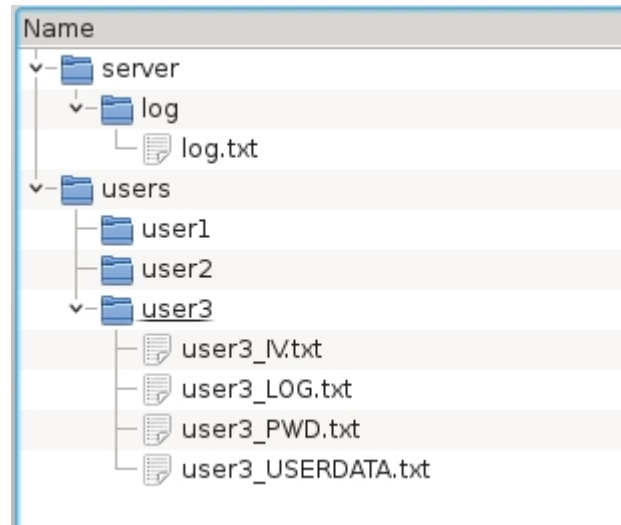
Server

Users are managed as folders, name of folder is the username

Username must be unique and consist of alphanumeric characters

User data is only ever stored as encrypted bytes on the server

User passwords are stored as salted and hashed bytes



Design

Client

- User data is encrypted with an AES key before being sent to server

- AES key is encrypted with the users master password

- Initialization vector (IV) for AES is regenerated each time the encryption function is called

- AES key may be stored in an arbitrary directory the user can access on their machines, including USB drives

Design

Communication

- Connection handler thread

- SSL Initialization moved to Jelly

- Wrapping of send and receive in Communication

Trade Offs

User data is encrypted locally

- Server never has capability to decrypt user data

- No recovery from key loss or corruption

User specifies where to store AES key and IV

- Allows for two factor authentication if stored on USB drive

- User could store key in insecure location

Trade Offs

AES key encrypted with master password

- Adversary requires both key and password to access user data

- No recovery from password loss

No password requirements

- Greater usability and freedom for users

- Users can choose an insecure password

Security

User authentication on server with password

Locally stored and encrypted secret key

Encrypted remote data storage

User authentication at login by server

Server authentication by client

Security Elements

Confidentiality

Secret keys are never transmitted over a network by this program; at no point does the server have access to them

Keys are encrypted with the user's account password on the client machine, and only stored in that state

Security Elements

Authentication

Users are authenticated with passwords stored in hashed and salted form on the server-side

Users are required to have access to their secret key files in order to decrypt and view their data

Security Elements

Integrity

SSL is used to verify that network transmissions are not modified between the server and client

There is a vanishingly small probability that modified data will decrypt without errors and will produce anything that could be mistaken for user data

Security Elements

Authorization

By interacting with the server, users are only ever allowed to touch files within their user directory

We ensure that usernames are alphanumeric (so “../otheruser” is disallowed)

Security Elements

Audit

We write to both user and server logs in order to track user data changes and account-level occurrences such as login, failed login, and account deletion

User logs contain only information pertinent to that user and are distributed to the user on request

Command Line Interface

```
Starting Jammed Client
Welcome to Jammed
...|-----|...
..|~~~~~|..
..|~JAM~|..
..|-----|..
.....
Your passwords have been preserved...

Jammed connected sending login

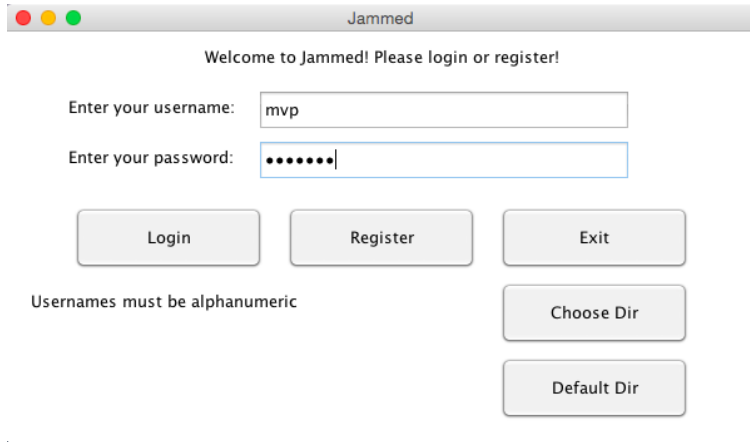
Please enter the directory your keys are located in.
If Registering, where you wish to save your key files
or hit enter to use the program 'keys' folder.
hints:(use '../' to jump up a folder)
      (end with dir name, DO NOT end the path with a '/')

Jammed/bin/: keys/
Enter "*" in the username field in order to enroll yourself as a new user.
Usernames should be chosen from the alphabet [a-zA-Z0-9].
Enter username: *
Enter desired username: mvpedreiro
Enter password: 123

Website:           Username:           Password:

Possible operations:
    "add" adds or modifies an entry
    "remove" deletes an entry
    "log" obtains log files for your account
    "change" changes the password to your account
    "exit" saves changes and exits the program
    "delete" deletes your user account
Enter type: █
```

GUI Interface



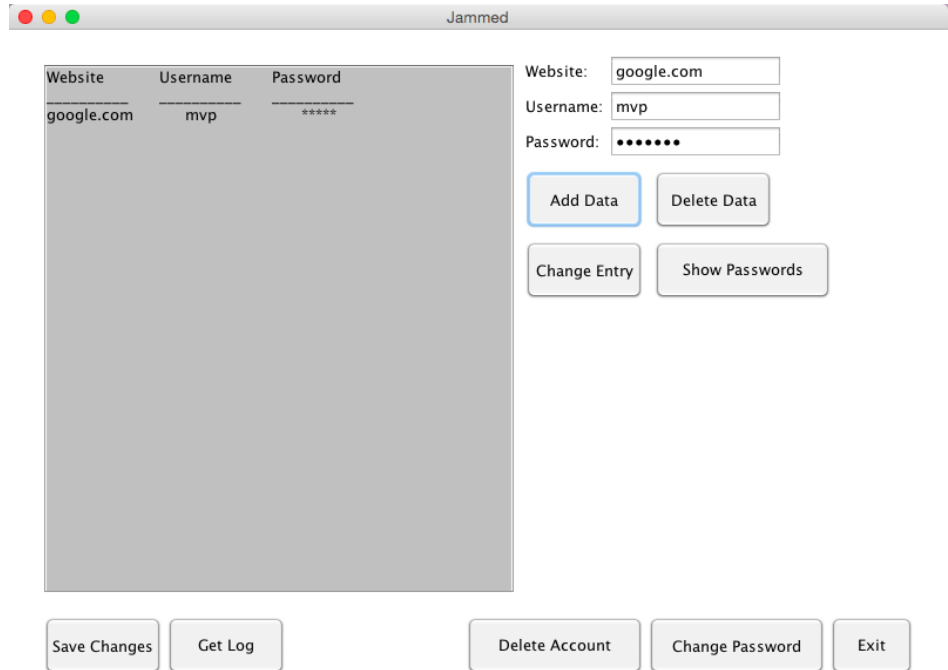
A screenshot of a macOS-style window titled "Jammed". The window contains a welcome message, input fields for username and password, and several buttons for login, registration, and account management.

Welcome to Jammed! Please login or register!

Enter your username:

Enter your password:

Usernames must be alphanumeric



A screenshot of a macOS-style window titled "Jammed". The window displays a table of stored data and a set of buttons for managing this data.

Website	Username	Password
google.com	mvp	••••••

Website:

Username:

Password:

Creating a User

Enter username and password

Select existing directory to store key and IV

Select enrollment

Server checks username and responds

If alright then proceed, else retry

Create secret key and session key locally

Program loads secret key, creates blank data

logged in!

Logging In

Enter username and password

Server checks password

Server replies with response

If okay, client application loads user keys

Server transmits user data to client

Client application decrypts and displays data

Client can now read and edit data

Changing Password

Login with current password

Decrypt data and key with current password

Submit new password to server

Encrypt data and key with new password

Save new data and key

Account deletion

Login with account you wish to delete

Select “Delete Account” and press confirm

Server accepts request, removes user files and deletes user folder

Client application exits

Note: User must manually dispose of their keys

Shortcomings

Loss of master password means data is lost forever

Logs are not tamper proof

Users must log out on their own (no timeouts)

Users must manually dispose of their AES keys upon account deletion

Client application not entirely stable

- If wrong directory is selected to read keys causes hang

Thank You

fin