CSE 453 Week 09 Lab Report

Threat Model

**Group 1** | Ryan Brower | Bryce Sanders | Lance Riley | Harrison Parrish | Everett Tsosie | Jared Harper

Table of Contents

[Asset List 3](#_Toc150608670)

[Data Flow Diagram 4](#_Toc150608671)

[Data Flow Diagram 4](#_Toc150608672)

[Data Flow in “Main.py” 5](#_Toc150608673)

[Data Flow in “Interact.py” 5](#_Toc150608674)

[Data Flow in “Message.py” & “Messages.py”: 6](#_Toc150608675)

[Threat List 7](#_Toc150608676)

[Threat Reports 8](#_Toc150608677)

[Ryan Brower Threat Report 8](#_Toc150608678)

[Bryce Sanders Threat Report 9](#_Toc150608679)

[Lance Riley Threat Report 9](#_Toc150608680)

[Harrison Parrish Threat Report 10](#_Toc150608681)

[Everett Tsosie Threat Report 11](#_Toc150608682)

[Jared Harper Threat Report 12](#_Toc150608683)

[Summary 13](#_Toc150608684)

# Asset List

What assets are to be protected by the program? As you decompose the program, think about the assets of the program and the system it is running on. What could potentially be threatened? What needs to be protected? Review the definition of an asset in Chapter 2 of the textbook.

The program is designed to host a session for a user to view messages stored by the program. The user must provide their username and the password associated with that username to log in. Assets stored by the program include:

* Messages
* Usernames
* Passwords

The main asset that could be threatened is the messages. The usernames and passwords are simply a means that an attacker would use to access the message. Because of the sensitive nature of the messages, all these assets must be protected so the messages do not become compromised.

# Data Flow Diagram

You will need to include every instance of data flowing in and out of the program as well as all processors. It might be necessary to drill down two or three levels depending on how much detail you put on each level. The key thing here is to properly identify the relevant trust boundaries. These include the trust boundaries inside the program, and those used in the entire system in which this program is just a part.

DFD Hints:

* Start with the interactors (that which provides input and accepts output) and follow the data through the program. Identify the major trust boundaries, the major data stores, and the major processors.
* Processors: We are most interested in processors that allow data to flow across a trust boundary. You don't need to get more detailed than the function level in the program. Don't get too far into the details of each function in the program. Focus on the data going into a function and the data going out of or being modified by the function (this is not necessarily through a return statement). How is that function acting as a processor? Where does it sit on the trust boundary?
* Data stores: What are the major data stores in the program/system? Think about external data stores pulled into the program as well as major data stores within the program.

### Data Flow Diagram

A black and white diagram

Description automatically generated

### Data Flow in “Main.py”

A black and white diagram

Description automatically generated

### Data Flow in “Interact.py”

A diagram of a class boundary

Description automatically generated

### Data Flow in “Message.py” & “Messages.py”:

A diagram of a computer

Description automatically generated

# Threat List

Come up with a list of threats. Go through each component of STRIDE and try to think of any possible threats for each one. As you write out the different threats, think in terms of the assets. Look back at the definition of a Threat in chapter 2 of the textbook and the "Identify Threats" step of the Threat Modeling process. How can the DFD help you identify threats?

Though the initial development team intended to implement access control by implementing a security level attribute for users, this is not present in the code. Because of this and other shortcomings in the program, threats to the assets include:

1. Every user has the same password. If a user knows one user account password, they can easily gain access to all accounts. (Spoofing)
2. Any user has access to edit all messages and can change them to say anything. (Tampering)
3. Any user has access to delete any messages. (Tampering)
4. There is no record of who edited or deleted a message. (Repudiation)
5. All messages are stored on a single unencrypted .txt file. (Information Disclosure)
6. A list of all usernames is displayed to any user trying to log in. (Information Disclosure)
7. Any user has access to view all messages. (Information Disclosure)
8. Only one session can be handled by the system at a time. (Denial of Service)
9. There is no implementation of security clearance levels, meaning any user can perform any action in the system. (Elevation of Privilege)

# Threat Reports

**Each member of the team should create one complete threat report**. Label each complete threat report with the name of the team member who completed it.

See Example 1 in Chapter 08 of the textbook. A "complete threat report" includes:

|  |  |
| --- | --- |
| **Description** | Completely and concisely describe the threat. |
| **Asset** | Completely and concisely describe the asset that is threatened. |
| **Threat Category** | Classify the threat using S.T.R.I.D.E. |
| **Risk** | Compute the risk (think D.R.E.A.D.). Also include a brief justification for your values. |
| **Mitigation** | Some ideas of how the threat is to be mitigated. |
| **Comments** | How the vulnerability could be exploited. It would be best if the exact steps necessary to exploit the vulnerability are given. |

### Ryan Brower Threat Report

|  |  |
| --- | --- |
| **Description** | 2. A “Public” user wants to edit message from Winston Churchill to say he is declaring war on the United States. |
| **Asset** | Storage |
| **Threat Category** | Protection Mechanism: Technology  State: Storage  Asset: Tampering |
| **Risk** | Damage Potential: **10** – The asset in this situation will be damaged beyond repair, with no history or record of the original data. This would be completely compromising.  Reproducibility: **10** – All an attacker would need is to both see the message stored by Winston Churchill and want to change it. The software contains no checks against this happening, and it can be produced identically every time.  Exploitability: **10** - Once understood, this exploit can be done by any user, anytime. It takes no extra skill than regular program operation.  Affected Users: **10** – Not only can every user be affected by this threat, but any user can also have an artificial view of reality because of an attack on a different user.  Discoverability: **7** – Those comfortable using the system would find that you can currently edit any message, not just your own. |
| **Mitigation** | This would be mitigated at the technology level, adding a check before any data is added, altered, or removed by a user. It should check the user’s authority and only allow operations that pertain to the same user (or a user with higher access).  Another recommendation would be to regularly back up all systems on a separate drive or storage mechanism. This would reduce the impact of an attack of this type, even when other mechanisms are in place for mitigation. |
| **Comments** | A vulnerability of this type results in a fatal lack of data security. Any data stored or handled in this system should not be trusted and should be removed from sensitive or protected systems. |

### Bryce Sanders Threat Report

|  |  |
| --- | --- |
| **Description** | 3. A “Confidential” user wants to edit a message from their commanding officer (a “Privileged” user) to avoid being court martialed. |
| **Asset** | Messages stored by the program. |
| **Threat Category** | Protection Mechanism: None  State: Storage  Asset: Tampering |
| **Risk** | Damage Potential: **10**, the asset is completely compromised in this situation.  Reproducibility: **9**, the attacker need only know how the system works to carry out the attack.  Exploitability: **8**, attacker only needs access to a computer that the program is running on.  Affected Users: **10**, all users are affected and can exploit this threat.  Discoverability: **8**, the attacker needs only see the menu of options to know that they can edit any message. |
| **Mitigation** | Implement access control to verify which users can and cannot edit the message. |
| **Comments** | Right now, with the previous development team failing to implement access control for messages, all messages in the system are compromised and the program is completely unusable from a security standpoint. |

### Lance Riley Threat Report

|  |  |
| --- | --- |
| **Description** | 5. A “public” user searches the program files to find where the messages are stored. |
| **Asset** | Information (messages) |
| **Threat Category** | Protection Mechanism: Policy & Practice (Access Control)  State: Storage  Asset: Information Disclosure |
| **Risk** | Damage Potential: **10**, unauthorized access to messages can lead to sensitive information.  Reproducibility: **10**, with the program in this state anyone can have access to any files that are sensitive.  Exploitability: **5**, if the file structure and access controls are not properly configured.  Affected Users: **10**, all users whose messages are stored.  Discoverability: **10**, because every user with the program state has access to all files. |
| **Mitigation** | Implement proper access controls to restrict file access, store sensitive information in encrypted form, or regularly audit and review file access permissions. |
| **Comments** | The current code does not appear to have specific access controls implemented. To enhance security, consider implementing access controls based on user roles and privileges. Encrypting the stored messages could provide an additional layer of security. |

### Harrison Parrish Threat Report

|  |  |
| --- | --- |
| **Description** | 7. A “Public” user wants to view the messages that has a “Secret” level of security clearance. |
| **Asset** | Messages stored by the program. |
| **Threat Category** | Protection Mechanism: Policy & Practice (Access Control)  State: Storage  Asset: Information Disclosure |
| **Risk** | Damage Potential: **8**, potential compromise of sensitive information, but not as severe as a complete compromise.  Reproducibility: **7**, requires specific conditions to be met, such as improper access controls.  Exploitability: **6**, dependent on the existing vulnerabilities in the access control system.  Affected Users: **9**, users with messages classified as "Secret" clearance are at risk.  Discoverability: **7**, dependent on the user's ability to identify and exploit existing vulnerabilities. |
| **Mitigation** | Implement robust access controls based on user roles and privileges to restrict access to messages based on security clearances.  Regularly audit and review access permissions to ensure proper enforcement of access controls.  Educate users on the importance of respecting security clearances and the potential consequences of unauthorized access. |
| **Comments** | The current lack of specific access controls poses a significant risk to the confidentiality of messages, particularly those with higher security clearances. Implementation of access controls based on security clearances is crucial to prevent unauthorized access and maintain the integrity of the program. Additionally, ongoing education and awareness efforts should be emphasized to promote a security-conscious user community. |

### Everett Tsosie Threat Report

|  |  |
| --- | --- |
| **Description** | 8. A “Public” user logs on to the program so that no other users have access to it.  Denial of service, restricting anyone from sending or retrieving information. |
| **Asset** | Messages, the entire system. |
| **Threat Category** | Protection Mechanism: Technology  State: Storage  Asset: Denial of Service |
| **Risk** | Damage Potential: **8**, access to the asset would be completely cut off. The attack does not destroy the asset, but it does eliminate all usefulness thereof with the possibility of recovery.  Reproducibility: **9**, the way the program stands, there is a high chance to perform the exploit. The only thing stopping the exploit is if a valid user is already occupying the system.  Exploitability: **10**, no effort is required to execute the exploit, typing in any username and any password would be enough to open a session to lock everyone out.  Affected Users: **10**, all users within the community that uses the system would be affected.  Discoverability: **10**, the exploit is obvious to everyone; even valid users would find they would not be able to use the system simultaneously. |
| **Mitigation** | Adjust the system to be able to handle multiple sessions at a time rather than just one. Have user’s sessions automatically close after having their session open for a set amount of time. Have user’s sessions close for inactivity. |
| **Comments** | Without having access to the messaging system, there practically no system at all. The whole application is rendered useless and requires a reboot to disconnect the attacker whenever this attack is performed. Even then, the attack can be effortlessly and quickly reproduced. |

### Jared Harper Threat Report

|  |  |
| --- | --- |
| **Description** | 1. A “Public” user wants to impersonate Admiral with “Secret” security clearance. |
| **Asset** | The username & password |
| **Threat Category** | Protection Mechanism: Technology  State: Storage  Asset: Spoofing |
| **Risk** | Damage Potential: **10**, the asset is completely compromised.  Reproducibility: **10**, it is beyond simple to repeat the actions needed to gain access to the confidential security clearance with the same results every single time.  Exploitability: **10**, the attacker merely needs to know admiral’s username and the password, and they be able to gain access to the admiral’s confidential account.  Affected Users: **10**, the admiral and anyone who has messages on the program.  Discoverability: **10**, the public user merely needs to guess “password” for the admiral’s password, and they would gain access to the highly sensitive parts of the application. |
| **Mitigation** | Change the password to something a lot more secure than “password”. |
| **Comments** | Access to the highly secure areas that should only be available to top trusted officials is easily broken into. The password is guessable by even the most casual of attackers. |

# Summary

Due to the initial development team’s inability to implement access controls to the messages stored in the program, there are a wide array of security risks that make the program unusable in its current state. The lack of encrypting the data and storing all messages in a single plain-text file means that the key assets of the program can be easily accessed by an attacker without even using the program. These and other issues lead to the lengthy list of risks in our evaluation and high D.R.E.A.D scores across the board. Upon our return to the future, we urge the next development team to implement these features to move the program to a more secure state.