Gil Rabara, Dilnoza Saidova, Abdulrehim Shuba
July 29, 2022
TCSS 483
Professor Tom Capaul

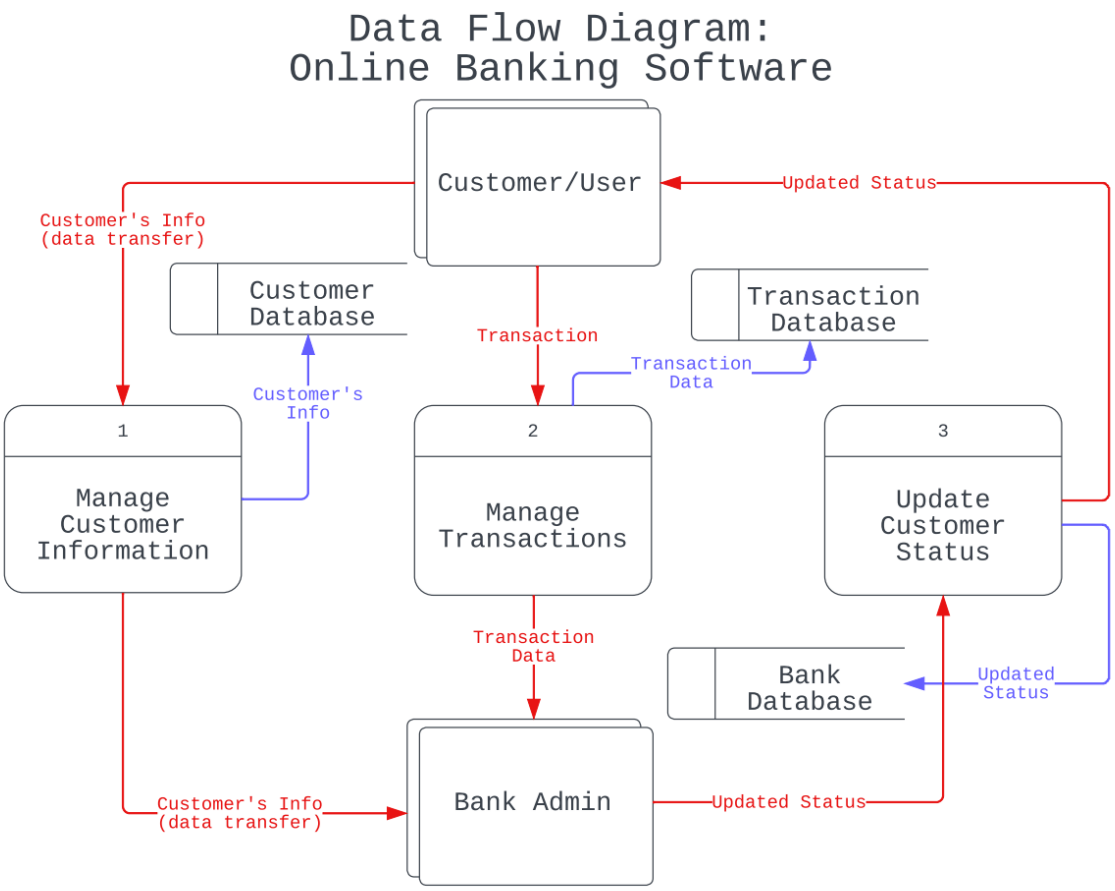
Team Assignment: Threat Modeling

Description of program/project/software:

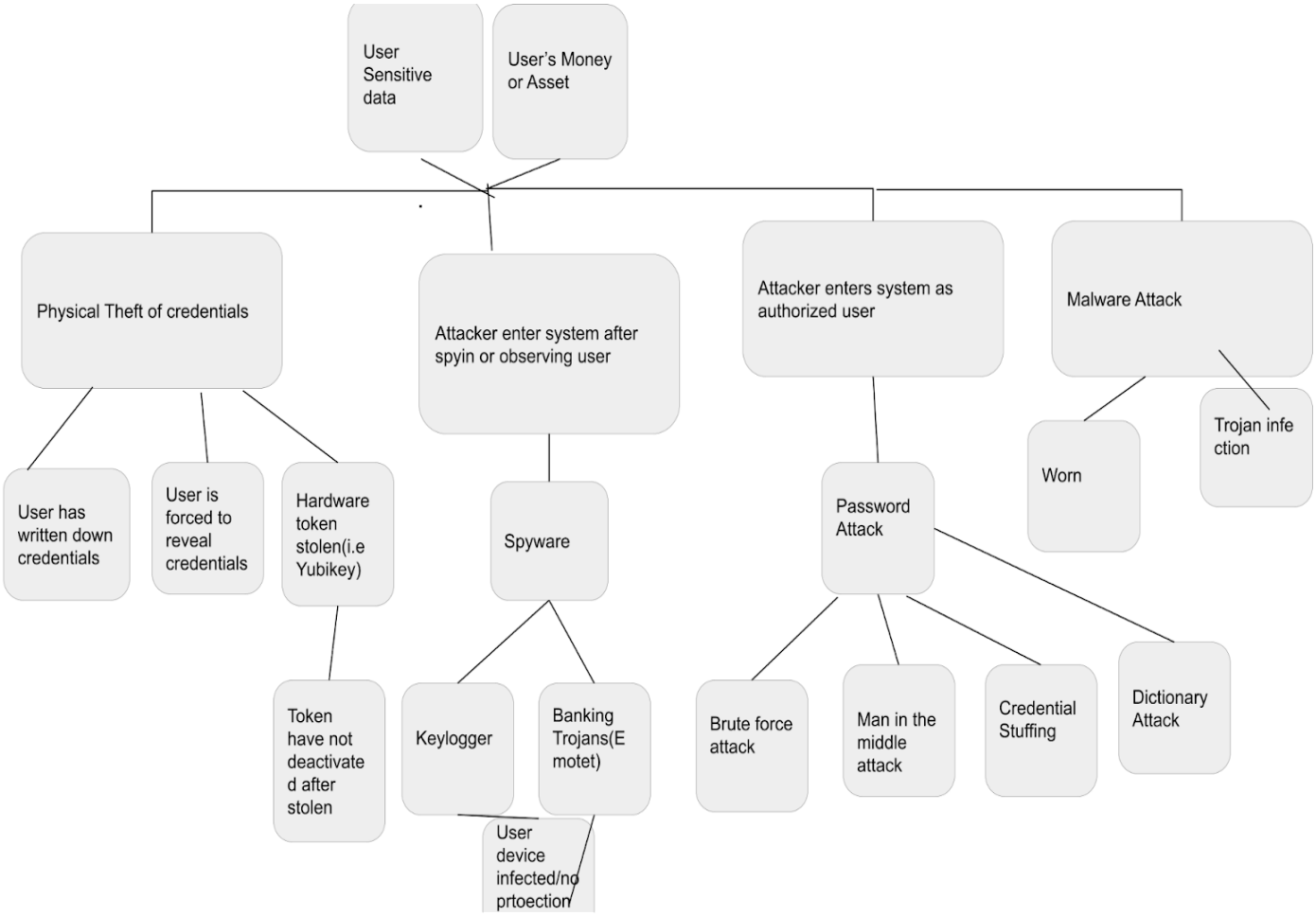
For our software we decided to go with an online banking software. In today's society we are so intertwined with technology that it is difficult to imagine a life without it. Our everyday lives have such a reliance on technology that most things we do, no matter how personal, usually involves some sort of software. Something as simple as depositing a check can entail loads of sensitive information such as social security, routing numbers, usernames, and passwords to name a few. Our group's focus was to see just how exploitable banking software can be and what potential measures need to take place to defend against an attack.

Generally, banking apps will require the user to create an account. Usually this consists of some kind of pin, username, biometric scan (like apple's facial recognition), and a valid email address for the username portion. As for the password, much like our regular expression assignment the user will need to input a personal password that has specific requirements including characters, uppercase letters, lowercase letters, and so on. After this is done the user will need to log into the provided email and verify that the user is the one performing this account creation. After the initial setup is complete, the user should have 'user privileges' to the banking software. With user privileges the user will be able to deposit, withdraw, and transfer money as well as other relevant features.

Data flow diagram showing the flow of data into, through, and out of the banking program:



A threat tree that shows at least two assets the program has and the venues through which those assets can be compromised:



A breakdown of threats to your program based on STRIDE:

Spoofing - accessing credentials; spoofing process - replacing with rogue.

Spoofing would be highly likely to happen. With a little social engineering the victim could be likely to disclose sensitive information allowing the attacker to login as the victim. (This is a common technique used by scam callers that ask you for your credit card information or control of your machine)

Tampering - replacing existing binary image or patching existing one in memory.

Code tampering is also highly likely to happen. Through phishing, attackers can trick users into giving access to confidential data. With this, attackers could gain full access to user's information and credentials.

Repudiation - user denies making an online purchase and there is no audit trail to prove otherwise.

For the banking software, this doesn't seem to be a highly probable attack. Looking at our data flow diagram, if a user were to request a loan, the data would be sent to the loan management system and they would be able to detect any kind of malicious activity such as repudiation. If the user was to request a transaction the data would flow to the transaction management system that should be able to detect any kind of malicious activity as it can also serve as an audit trail to prove otherwise. The only other feature the user has is the account management system that would also go through cash records and get directed towards the loan management system.

Information Disclosure - reverse engineering process to discover secret data.

This attack is also highly likely if there is no encryption of data. The places to look at in the data flow diagram if the software was to be attacked by some kind of information disclosure would be in the translation management system, account management system, or the loan management system as all of these are places a customer's sensitive information could be directed to.

Denial of Service - unexpected input to an application causes it to slow down affecting all users.

If any of the management systems were to be compromised this would effectively provide a denial of service for all customers.

Elevation of Privilege - a vulnerability in app lets attacker open a shell on machine app is running on with current user privilege level on that machine which happens to be admin.

According to our data flow diagram our software doesn't seem to be easily vulnerable to an elevation of privilege attack. We can assume this because nothing connects with the manager, accountant/cashier, or bank users. Although less likely, there is always a chance of an attack.

A ranking of threats using either DREAD or PASTA:

Damage potential - (how bad would an attack be?)
Reproducibility - (how easy is it to reproduce the attack?)
Exploitability - (how much work is it to launch the attack?)
Affected users - (how many people will be impacted?)
Discoverability - (how easy it is to discover the threat?)

Scale 1 - 10 (10 is the worst)

Threats	D	R	E	A	D	Total	Rating
Spoofing	8	4	9	2	7	30	6
Tampering	9	4	4	2	6	25	5
Repudiation	5	3	10	3	4	25	5
Information Disclosure	6	4	3	6	3	22	4.4
Denial of Service	5	4	2	7	3	21	4.2
Elevation of Privilege	9	3	2	8	1	23	4.6

Work Cited:

"Data Flow Diagram for Online Banking System." *GeeksforGeeks*, GeeksforGeeks, 5 Mar. 2021, https://www.geeksforgeeks.org/data-flow-diagram-for-online-banking-system/.

Hlebowitsh, Nadia. "5 Common Security Flaws in Banking Apps." *Onsharp*, HubSpot, https://blog.onsharp.com/5-common-security-flaws-in-banking-apps.

Miller, Lawrence C., and Peter H. Gregory. "What Is Security Threat Modeling?" *Dummies*, Wiley, 12 Sept. 2016, https://www.dummies.com/article/academics-the-arts/study-skills-test-prep/cissp/security-threat-modeling-225503/.