THREAT MODELING

Secure Programming

Dara OSullivan
20080494

Table of contents

Recent attacks and vulnerabilities

Between 2017 and 2018 90% of the security breaches can be blamed on poor programming practice according to risk consulting firm Kroll. After some research it seems that the biggest threats arise from the following areas: privacy violation, insecure storage, insecure transport, insecure deployment and poor logging practice. In terms of these areas the most common attacks are ddos, phishing scams, password hacking, remote code execution and fake wireless access points. Below are a three security breaches which occurred in 2019 due to poor programming practice.

- XSS, or cross-site scripting attack performed on Fornite.

Access to 200 million players private data and accounts was gained by hackers. Hackers exploited a web page vulnerability to employ an XSS attack when victims clicked on a link. Monitoring input is the best way to prevent XSS attacks.

- Ddos smokescreen on Carphone warehouse.

Access to 2.4 million customers personal data stolen by hackers. Hackers swamped the system with junk traffic before performing a more significant data breach. Some common ways to avoid Ddos attacks are anit-Ddos hardware and protection appliance and configuring hardware against attack by configuring firewall to drop certain packets and buy more bandwidth.

- Phishing attack on Target Corp.

Credit card and personal data of 110 million customers exposed. Hackers malware-laced email phising attack to employees at a firm in which Target did business with. Prevention for phishing attacks involves requiring encryption for telecommunication, identifying phishing emails and keep up to date on security patches.

Requirements

From the moment a customer or employee (actors) create an account on a ticket booking website they are in a potentially vulnerable position if certain requirements are not met. There is an element of risk involved from the customer perspective. If a website is not secure, using credit card information of a customer, impersonating a customer and much more are all very real possibilities. Generally, when a customer uses a ticket booking website, they give information such as address, phone number, email, credit card information, this is all important private information which needs to be protected.

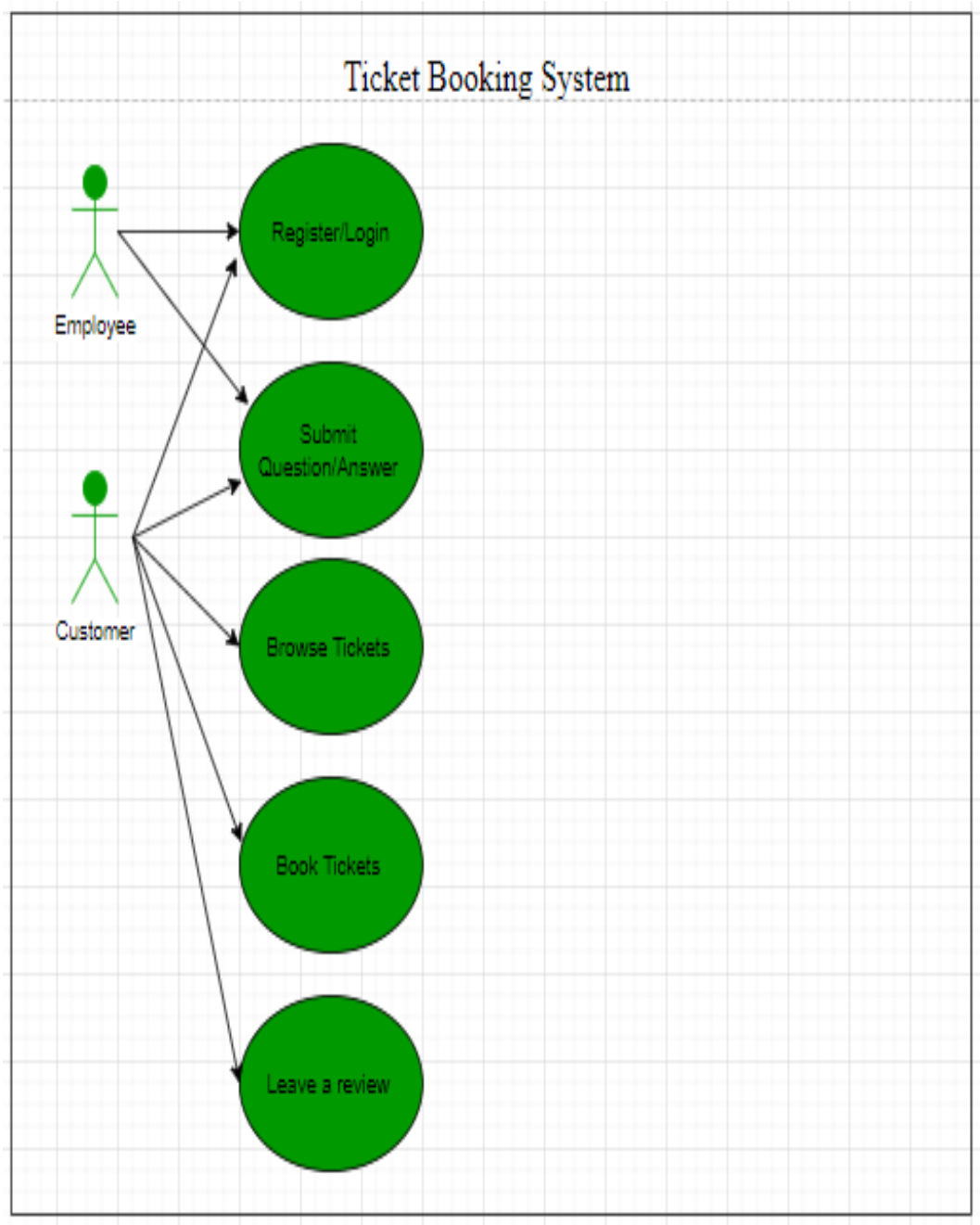
For a ticket booking online system to be functional it needs to be able to perform the following tasks:

- The system allows new customers and employees to create an account.
- Allow registered customers and employees to login and logout of sessions.
- It must allow customers to browse tickets.
- Allows customers to submit questions and employees to answer these questions.
- Customers need to be able to book and pay for tickets.
- Customers will be able to leave a review.
- System must work 24/7.
- System should support all browsers.

It is necessary to do the following to ensure a safe system.

- All transferable data for example card numbers, must be done in secured connection.
- Keep software up to date at all times. Both software used such as a content management system or a forum, and the server operating system need to be up to date. Updates in software are very important and are often updated for important reasons including security advances. If updates are not made you are leaving the website in a particularly vulnerable state.

Use cases diagram



Use cases description

I will be basing my use case on the major operations of the ticket booking system. Above you can see the customers and employees have the ability to create an account. Once it's created, they can then log in over and over with their credentials. Registered customers can search tickets and ticket availability. Registered customers can book tickets which involves payment. A customer can ask questions over the system which employees will respond to. Customers can write and submit reviews to the system.

Use case one description

This use case describes how a user and employee log into the Ticket Booking System.

Primary actor: Customer and Employee.

Stakeholders and interests: Customer and Employee.

Level: User goal.

Trigger: User needs access to online ticket booking system.

Precondition: User is not yet logged in.

Success guarantees: User is logged into system.

Main success scenario:

1. The system validates the actor's password and logs him/her into the system.
2. The system displays the Main Form and the use case ends.

Fail scenario

1. Username and/or password are not valid.

Use case two description

This use case describes how a user can search tickets and ticket availability.

Primary actor: Customer.

Stakeholders and interests: Customer.

Level: User goal.

Trigger: User wants to book ticket.

Precondition: User is logged in.

Success guarantees: User can see all available tickets.

Main success scenario:

1. The system displays all current available tickets which can be bought.
2. The system does not display bought tickets and the use case ends.

Use case three description

This use case describes how a user can submit a question to be answered by a member of staff and how an employee can reply to these questions on the Ticket Booking System.

Primary actor: Customer and Employee.

Stakeholders and interests: Customer and Employee.

Level: User goal.

Trigger: User has question they would like answered.

Precondition: Users are logged in.

Success guarantees: Employee receives message from customer and customer receives a response from employee.

Main success scenario:

1. The system provides submission area for customer to ask question.
2. The system shows this question to employee.
3. The system provides submission area for employee to answer question.
4. The system shows this response to the user.

Use case four description

This use case describes how a user can book and pay for a ticket from the Ticket Booking System.

Primary actor: Customer.

Stakeholders and interests: Customer and Employee.

Level: User goal.

Trigger: User has a ticket they want to book.

Precondition: User is logged in and has selected ticket to purchase.

Success guarantees: Ticket is booked for customer and a payment has been made.

Main success scenario:

1. The system ensures chosen ticket is available.
2. The system obtains and verifies bank details.
3. The system marks ticket as booked.

Use case five description

This use case describes how a user can leave a review on the Ticket Booking System.

Primary actor: Customer.

Stakeholders and interests: Customer and Employee.

Level: User goal

Trigger: User wants to leave review about the ticket booking system.

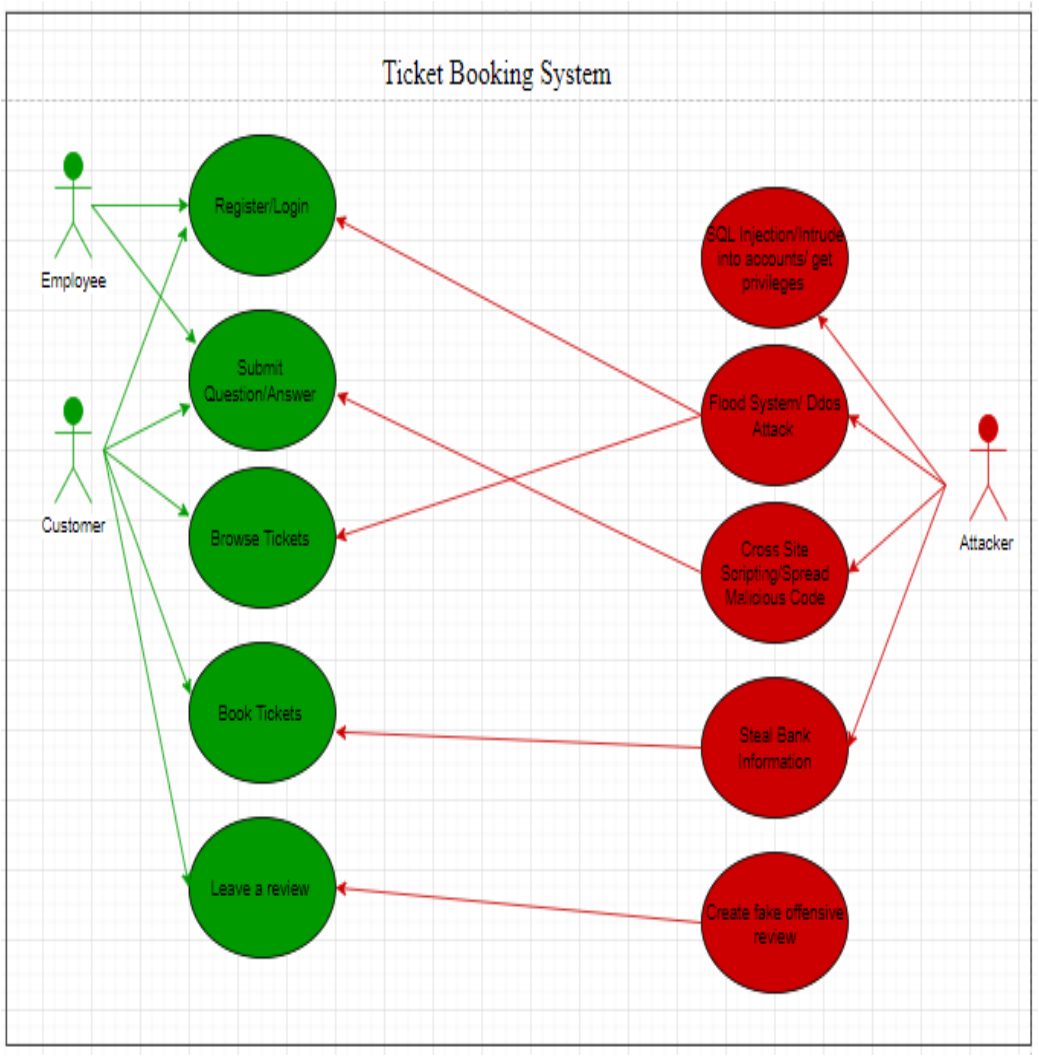
Precondition: User is logged in.

Success guarantees: User is logged into system.

Main success scenario:

1. The system provides review submission box.
2. The system makes this available to logged in customers.
3. The system will obtain any submitted reviews and display them on the website page.

Abuse cases diagram

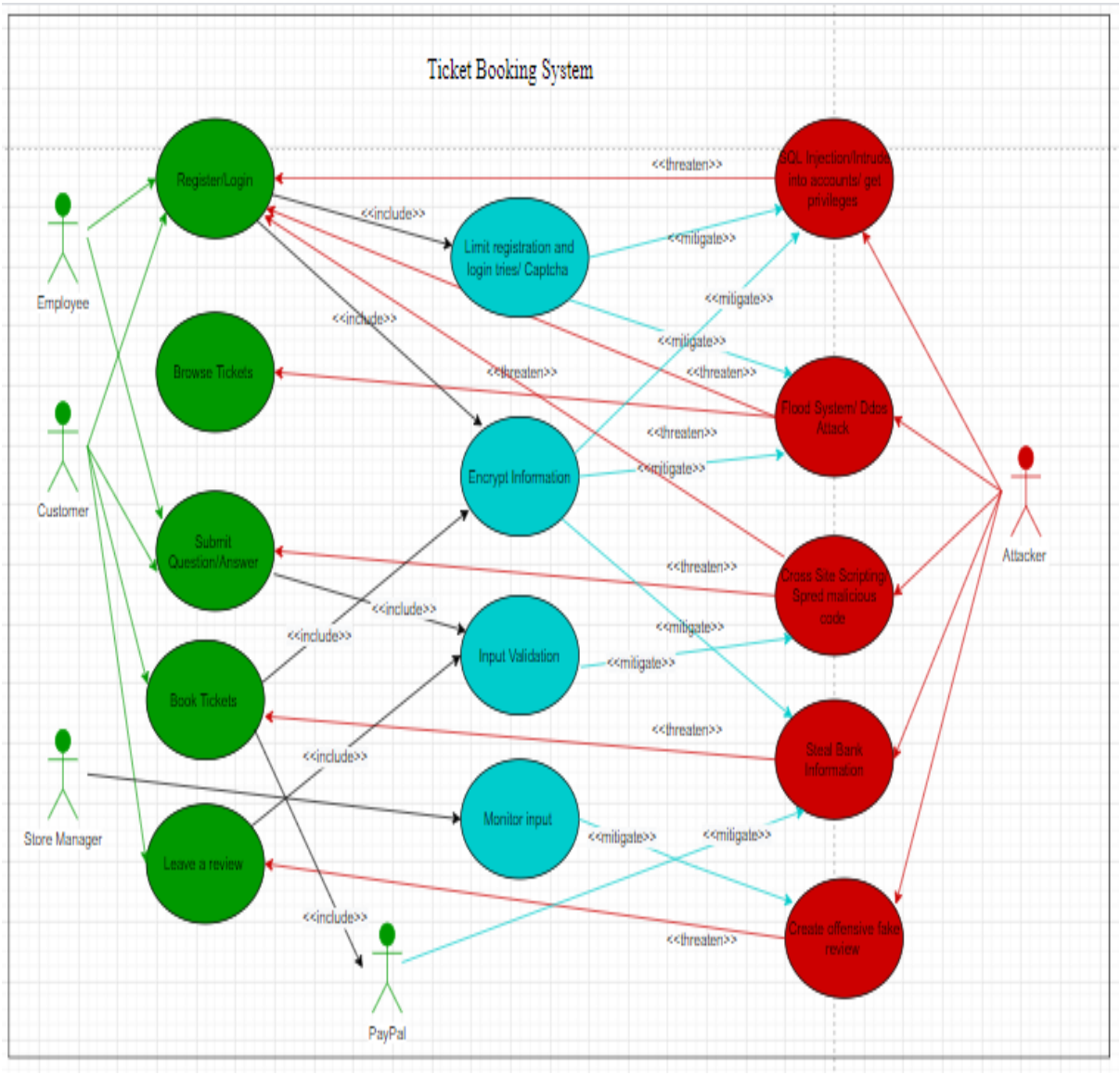


Abuse cases description

Above I have added the attacker and different potential ways this particular system could be attacked. There are five new use cases as a result.

- The requirement of registering and logging in allows a chance for the attacker to do an SQL injection which could destroy the database. When asked for a username input and password the attacker can place malicious code which can run on the database.
- The attacker could flood the system and perform a distributed denial of service attack (Ddos) on the system through the register/login requirement and the browsing tickets requirement. A distributed denial-of-service (DDoS) attack occurs when multiple systems flood the bandwidth or resources of a targeted system. Such an attack will result in the system shutting down, making it unavailable to customers.
- When booking a ticket, a customer needs to enter their bank details, this gives the attacker an opportunity to access them. Once one is a registered customer a portion of their data will be stored under their account within the ticket booking system. This data could be stolen and exposed if not secured.
- Another possible attack or threat is giving false reviews which could be used to damage the ticket booking systems reputation.
- When the customers and employees ask and answer questions it gives opportunity for a cross-site scripting attack. This attack is a type of injection. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. The user has no way of knowing this script is from an untrustworthy user and the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser.

Augmented requirements



Above you can see what I used as methods of mitigation.

- Encryption ensures all data stored by the system is essentially unusable to any other party which resultantly prevents the threats of identity theft and the ability to expose data. For these reasons' encryption can be used as a method of mitigation for SQL injections on the register and login requirement and also for stealing bank information. I have also added a new actor Paypal which can be used when booking tickets as a middle man to ensure and secure and safe payment.
- Adding a captcha and limiting the amount of login tries in the login requirement will ensure only users rather than systems are gaining access to the system and untrustworthy users are kept out.
- Authentication and monitoring the input will prevent an attacker from making false reviews as during the authentication process the credentials are cross checked. We can use monitoring input to mitigate cross site scripting when users ask and answer questions.