Payatu

Payatu Case Study

LLM Security Assessments in AI-Powered MedTech Applications

Payatu Security Consulting Pvt. Ltd

Project Overview

As the world of technology evolves, more and more companies are relying on AI to drive this growth and innovation!

Every industry now has AI integrated in atleast some of its processes or systems, and MedTech is no different.

A renowned MedTech leader known for its highly efficient medical and surgical equipment and various other patient and healthcare products jumped on the AI train to ensure that its clients receive only the best of the best services.

The company developed a chat-based application that uses LLM models to answer prompts requested by end users.

It was important for this MedTech leader to identify all security flaws and gaps in its LLM models and the application before having its customers use it consistently.

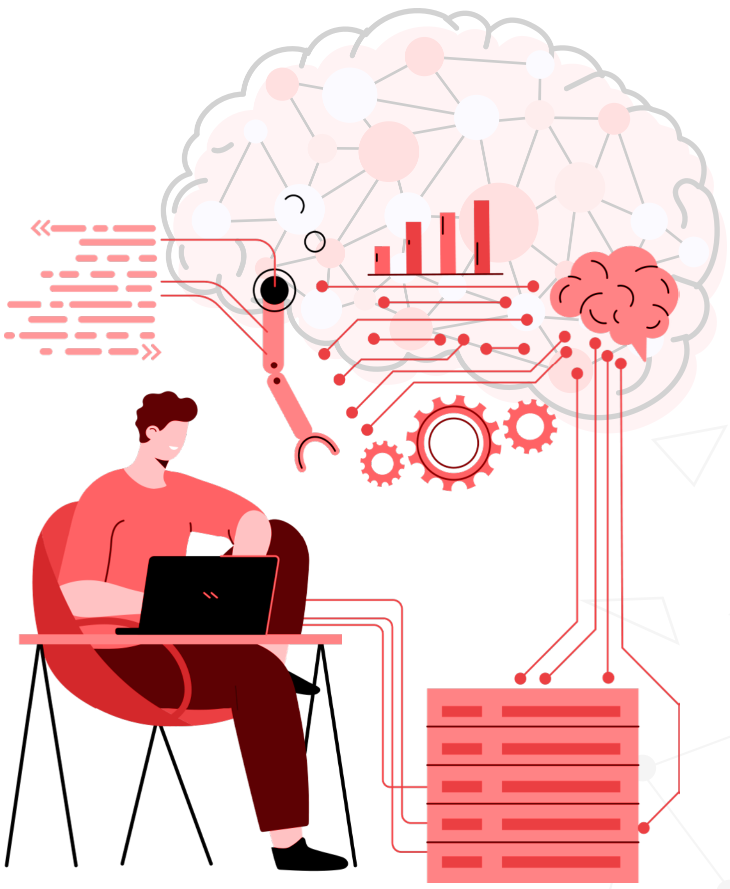
Payatu stepped up to identify any security issues, explain the risk associated with the identified issues, and guide the prioritization and remediation steps.

Let's take a look at how things panned out!

The Scope

1. LLM Security Assessment on the Web Application

2. Mobile Application Security Assessment



Process

iOS Security Assessment Process

1. Discovery

The discovery process includes information gathering which is the most important stage in a security test. The gathered information serves as a starting point for the process of looking for vulnerabilities, which can make or break a pentest.

2. Understanding the Platform

The Bandits make it a point to understand the mobile application platform from an external point of view, to aid in developing a threat model for the application. They take into account the company behind the app, its business case, and related stakeholders. The internal structures and processes are also considered.

3. Assessment

The process of assessing mobile applications is unique because it requires checking the applications before and after installation.

3.1 Static Analysis

Static Analysis is a method of examining source code without executing the application. This phase provides an understanding of code structure and makes sure the code adheres to certain standards such as OWASP.

3.2 Reverse Engineering

This involves converting the compiled applications into human-readable source code. The Bandits review the readable code in order to understand the internal application functionality and search for vulnerabilities. The iOS application's source code may be modified once reversed and recompiled.

3.3 Local File Analysis

As soon as the application is installed, it has its own directory within the filesystem. When the application is being used, it reads and writes from this sandboxed directory. Throughout the testing phase, the files in the sandboxed directory in the local storage were assessed.

3.4 Dynamic Analysis

The Bandits review the mobile application as it runs on the device. Reviews done include I/O operations done within the local storage during runtime and an assessment of the network traffic between the application and server. This phase provides an actual application behavior, and we check for vulnerabilities in several categories.

4. Exploitation

Using the knowledge gained in the previous step, exploits are written to actively attack and subvert the target. The Bandits need to find hidden cues which can successfully shed light on different vulnerabilities, which become a determining factor between a successful and unsuccessful test.

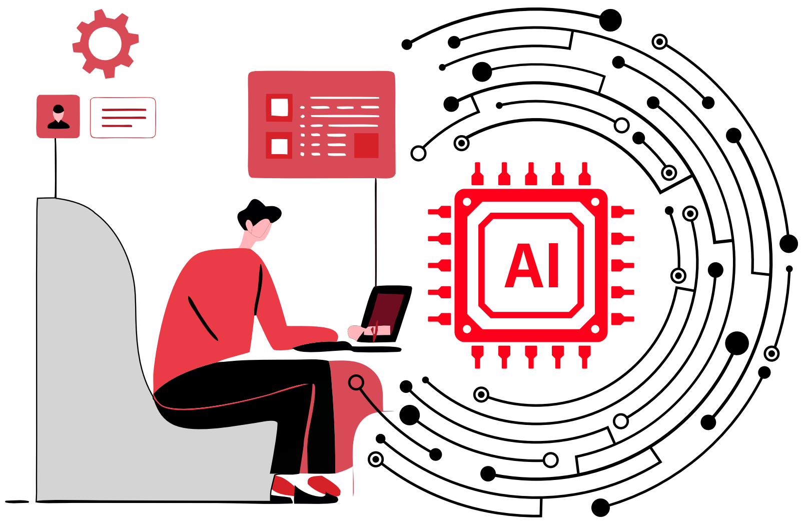
The exploitation may use individual vulnerabilities or combine a few to create complex and high severity attacks depending on the types of vulnerabilities found.

5. Reporting

This is the presentable outcome of the security assessment exercise. All exploit-able security vulnerabilities in the target system are recorded with associated CVSS v3.1 based scores and are reported to the client.

In the report, the details of the vulnerabilities, attacks, and proof of vulnerability are captured along with recommendations on how to mitigate those vulnerabilities.

LLM Testing Process



1. Identify the LLM Architecture

Determine the LLM architecture in use, such as GPT or BERT. This helps understand the model's core design and behavior.

2. Review Model Documentation

Review the model's documentation to understand its capabilities, limitations, and safety measures.

3. Identify Access Methods

Identify how the LLM can be accessed, such as through chatbots, APIs, or other interfaces.

4. Determine Authentication Mechanisms and User Permissions

Examine how users authenticate (e.g., API keys, OAuth) and what permissions are assigned.

5. Create a List of Potential Attacks

Prepare a list of known AI vulnerabilities, such as prompt injection, jailbreaking, and system prompt leakage.

6. Test for Biased or Harmful Responses

Run tests to see if the model can produce biased, harmful, or inappropriate content.

7. Test for Filter Bypass

Attempt to bypass content moderation filters using different languages, symbols, or altered phrasing.

8. Document Vulnerabilities

Keep a detailed record of vulnerabilities, including reproduction steps and potential security risks.

Test Cases

A customized list of test cases was developed for the client where some of the standard test cases were -..

iOS Application

Reverse Engineering:

- Review all plist files inside the binary
- Check for ATS-related issues
- Check for Excessive Permissions
- Check for Insecure Logging
- Check for Insecure Firebase Database
- Checking for Sensitive Information in NSUserDefaults
- Sensitive Data in Strings
- Hardcoded Secrets
- Testing for Overly Scoped API Keys
- Check for Custom URI Handlers/Deeplinks/Custom Schemes

Binary Protections:

- Check for Jailbreak Detection Implementation
- Check for SSL Pinning Implementation
- Check for ARC Flag
- Check for PIE Flag
- Check for Stack Canary Flag
- Check for SSL Pinning Bypass
- Check for Jailbreak Detection Bypass
- Frida Detection Check not Implemented
- Check for Frida Detection Bypass

Local Authentication:

- Touch ID Bypass/Local AuthN Bypass
- Face ID Bypass/Local AuthN Bypass

Insecure Data Storage:

- Check for Sensitive Information in Cached Files
- Check for Sensitive Information in PList Files
- Check for Core Data Storage (SQLite Database)
- Check for YapDatabases (SQLite Database)
- Sensitive Information in PList Files inside Local Storage

Insecure Logging:

- Check for Sensitive information in Device Syslogs

Miscellaneous:

- The App Removes Sensitive Data from Views when Moved to the Background

Large Language Model Attacks:

- Prompt Injection
- Jailbreaking of Model
- Incorrect Output Prediction
- Excessive Agency
- Insecure Input Handling
- Model DOS
- Bias in Responses
- Sensitive Information Exposure
- Handling of Misinformation and Fact-Checking
- Compliance with Ethical Guidelines
- Malicious Content Creation
- Access to the Unauthorized Information
- System Prompt Leak

Findings

HIGH:

Handling of Misinformation and Fact-Checking

Jailbreaking of Model

Malicious Content Creation

MEDIUM:

Leaked System Prompt

Bias in Responses

LOW:

Insecure Input Handling – XSS

Insecure Input Handling - Request to External URLs

Jailbreak Detection Not Implemented

Improper Session Management

SSL Pinning Not Implemented

Frida Detection Check Not Implemented

Recommendations

Users should not be allowed to upload arbitrary documents at scale (especially accessible to other users over RAG) without proper content validation, policies, and information tracking in place.

Use open-source frameworks like NeMo Guardrails by Nvidia and Langkit library to detect and prevent the possible attempts of Jailbreak and Prompt Injection.

The system prompt (engineered prompt) should not be sent as a request from the browser to the server in any case. The feedback should be recorded based on the conversation ID and not on the entire conversation.

Engineer the prompt explicitly to not leak any information about the prompt under any circumstances. This prevents leaking of engineered prompt in LLM response.

The user input should be validated properly. The application should treat the user input as a string and not execute it as a malicious JavaScript code.

Implement validation of markdown inputs: User input validation should be implemented on the UI and the UI should not render any markdown text.

Avoid LLM response: The backend code should not relay the images from user input to the LLM model.

Implement Session Management on the server side as well across all APIs once the user has logged off from the application.

Implement SSL pinning in the mobile application.

Implement the Runtime Application Self Protection (RASP) library in the application.

Potential Technical and Business Impact

Technical Impact	Business Impact
The upload of documents containing misinformation, abuse, and bias at scale can result in the client's application generating similar responses, which may impact all users.	Successful exploitation allows an attacker to upload false information about the company, and the lack of fact-checking by the application can unintentionally lead to the spread of false information at scale.
Jailbreak prompts manipulate the LLM's predictions, causing the model to disregard the engineered prompts and rules set during fine-tuning.	Jailbroken models can be used by threat actors within the client's company for malicious purposes, such as generating ransomware, phishing content, and planning cyber-attacks, which can become a potential legal liability for the client.

Technical Impact	Business Impact
The system prompt instructs the model not to give medical advice, but after a jailbreak, the model bypassed these restrictions and suggested medicines and treatments.	LLM models prone to hallucination may generate incorrect or inconsistent medical advice, putting the company in violation of FDA regulations for AI software used in healthcare.
Leaked prompts can make it easier for attackers to understand the intent of the LLM application, allowing them tocreate more effective injection prompts to exploit the system.	System prompts are the intellectual property of the business, and their leakage can adversely affect the confidentiality of the application, potentially leading to a competitive disadvantage.
Biased responses generated by the application can perpetuate misinformation and reinforce false beliefs, contributing to a lack of trust in the system.	A biased response generated by the client application can lead to misinformation, which may result in serious legal liabilities and reputational damage for the company.

Technical Impact	Business Impact
Successful exploitation could provide attackers with temporary or long-term access to user accounts, potentially allowing them to perform actions on behalf of the user and gain access equivalent to that of a normal user.	Successful exploitation could lead to a loss of confidentiality and partial integrity, eroding user trust and damaging the company's credibility.
Requests made to external malicious servers could leak user information, such as IP addresses, locations, and browser details, compromising user privacy and security.	The application sending requests to external links for rendering markdown images could increase the risk of data leakage to third-party servers, resulting in privacy violations, regulatory non-compliance, and reputational harm.
Without SSL pinning in an iOS application, there is an increased risk of man-in-the-middle attacks, as malicious actors could intercept and manipulate communication between the application and the server.	The absence of SSL pinning in an iOS application might increase the attack surface in an application, leading to more attackers testing the APIs potentially leading to unauthorized access.

Technical Impact	Business Impact
The absence of proper session management in the API poses significant security risks. Without session validation through cookies or tokens, unauthorized users can freely access sensitive endpoints, potentially compromising user data, system integrity, and confidentiality.	Unauthorized access may result in data breaches, financial losses, and reputational damage. Customers' trust could be eroded, leading to decreased user engagement and potential legal repercussions. Implementing robust session management is crucial to safeguarding user information and maintaining business credibility.
The lack of jailbreak detection allows malicious applications to access system directories, including the system partition, enabling unauthorized data access and information gathering.	As jailbreak detection is not implemented, the application can be installed and used on a jailbroken device. A jailbroken device allows access to all the system directories from the device including the system partition. This can lead to reduced user engagement and legal repercussions.

About Payatu

Payatu is a Research-powered cybersecurity services and training company specialized in IoT, Embedded Web, Mobile, Cloud, & Infrastructure security assessments with a proven track record of securing software, hardware and infrastructure for customers across 20+ countries.

AI / ML Security Audit 🔗

Incorrectly implemented AI/ML systems can lead to security and privacy issues. The severity of which depends on how critical the use case is. The repercussions of the same include misclassification of unauthorized entities, theft of intellectual property such as application train models, etc. With a dedicated team capable of effectively assessing and strengthening AI/ML systems, we can provide specific methods to prevent potentially damaging threats before they potentially derail your project.

Mobile Security Testing 🔗

Detect complex vulnerabilities & security loopholes. Guard your mobile application and user's data against cyberattacks, by having Payatu test the security of your mobile application.

Web Security Testing 🔗

Internet attackers are everywhere. Sometimes they are evident. Many times, they are undetectable. Their motive is to attack web applications every day, stealing personal information and user data. With Payatu, you can spot complex vulnerabilities that are easy to miss and guard your website and user's data against cyberattacks.

Security Operations Center 🔗

Cyber threats are everywhere, often operating in the shadows. Their goal: to breach networks, compromise systems, and steal critical data. With Payatu's SOC service, you can uncover these hidden threats, bolster your defenses, and protect your data from relentless cyber attacks.

IoT Security Testing

IoT product security assessment is a complete security audit of embedded systems, network services, applications and firmware. Payatu uses its expertise in this domain to detect complex vulnerabilities & security loopholes to guard your IoT products against cyberattacks.

Product Security

Save time while still delivering a secure end-product with Payatu. Make sure that each component maintains a uniform level of security so that all the components "fit" together in your mega-product.

Cloud Security Assessment

As long as cloud servers live on, the need to protect them will not diminish. Both cloud providers and users have a shared. As long as cloud servers live on, the need to protect them will not diminish.
Both cloud providers and users have a shared responsibility to secure the information stored in their cloud Payatu's expertise in cloud protection helps you with the same. Its layered security review enables you to mitigate this by building scalable and secure applications & identifying potential vulnerabilities in your cloud environment.

Code Review

Payatu's Secure Code Review includes inspecting, scanning and evaluating source code for defects and weaknesses. It includes the best secure coding practices that apply security consideration and defend the software from attacks.

Red Team Assessment

Red Team Assessment is a goal-directed, multidimensional & malicious threat emulation. Payatu uses offensive tactics, techniques, and procedures to access an organization's crown jewels and test its readiness to detect and withstand a targeted attack.

DevSecOps Consulting 🔗

DevSecOps is DevOps done the right way. With security compromises and data breaches happening left, right & center, making security an integral part of the development workflow is more important than ever. With Payatu, you get an insight to security measures that can be taken in integration with the CI/CD pipeline to increase the visibility of security threats.

Critical Infrastructure Assessment 🔗

There are various security threats focusing on Critical Infrastructures like Oil and Gas, Chemical Plants, Pharmaceuticals, Electrical Grids, Manufacturing Plants, Transportation systems etc. and can significantly impact your production operations. With Payatu's OT security expertise you can get a thorough ICS Maturity, Risk and Compliance Assessment done to protect your critical infrastructure.

CTI 🔗

The area of expertise in the wide arena of cybersecurity that is focused on collecting and analyzing the existing and potential threats is known as Cyber Threat Intelligence or CTI. Clients can benefit from Payatu's CTI by getting – social media monitoring, repository monitoring, darkweb monitoring, mobile app monitoring, domain monitoring, and document sharing platform monitoring done for their brand.

More Services Offered

- Trainings 🔗

More Products Offered

- EXPLIoT 🔗

Payatu

Payatu Security Consulting Pvt. Ltd.

www.payatu.com

info@payatu.com

+91 20 41207726