Optus Cyberattack Report

Overview of the Optus Cyberattack

- **What Happened**

In September of 2022, the Australian telecommunication giant, Optus suffered from the third largest customer data breach in the nation's history affecting nearly 10 million households. The cybercriminals were able to obtain a trove of personal data including home addresses, identities, email addresses, and medical records. Following the attack Optus characterised it as being highly 'sophisticated' in nature. However, following an investigation it was revealed through the Australian government that the problem was caused the cybercriminals exploited a vulnerability in the company's API. This vulnerability was a coding error that allowed anyone on the internet to query customer data without authorisation/verification. This type of exploit is known as is referred to as Broken Object Level Authorisation (BOLA), since the API accepted any customer identifier and returned critical data without authorisation. The hackers proceeded to extort Optus demanding a ransom of $1 500 000 worth of Monero (A privacy centric crypto currency), however they apologised and withdrew the demand due to the incident being highly publicised. In essence this cyberattack occurred due to Optus' failure to secure an API, which cybercriminals leveraged to illegally obtain millions of sensitive user data.

- **Consequences and Impact**

The breach had severe repercussions on Optus' reputation and stakeholders. The sensitive personal data of Optus users throughout the nation was exposed. This put the company's users at risk of identity fraud or theft. Optus faced severe backlash from the community and regulators. A survey sent out following the breach around 3 million (30%) Optus users considered leaving the company. In response Optus undertook a costly process to regain trust by agreeing to pay for the loss of customer data such as driver's license and passports. Government regulation bodies and officials criticised Optus for what they deemed was basic security negligence. The Australian minister for home affairs, Claire O'Neil stated "we should not have a telecommunication's provider in this country that effectively left the window open". The incident prompted a government investigation in the company and a class-action lawsuit from the Australian Communications and Media Authority (ACMA), seeking penalties for Optus' failure to protect consumer data. The breach's consequences extended far beyond the initial loss of personal data. Inviting regulatory action and class action lawsuits, while eroding consumer trust.

- **Lessons Learned**

The incident underscores many important lessons. Primarily, that even large enterprises, such as Optus must not overlook fundamental security practices. APIs must be built with security by design, ensuring systems are built securely from the start with critical features such as required authentication, enforced action controls, and isolating test APIs. Furthermore, corporations have to be vigilant over their weak points rather than neglect them. Optus was unaware that an old test API, api.www.optus.com, was left accessible since 2017. Regular audits could have helped identify this security flaw in advance. Additionally, maintaining a current database of critical assets is essential in preventing cyber-attacks. Finally, the breach showed the importance of in-depth security testing and oversight of critical systems. If this had been performed before the breach the company would have realised the lack of authentication in the test API and amended the issue. This attack serves as a cautionary tale for other corporations and businesses and showed that the investments needed to prevent an attack far outweighed the consequences of a breach, e.g., customer loyalty, regulatory investigation, reputation, and monetary consequences.

Secure Development Practices Evaluation

The Optus cyberattack provides us insight into how we can use certain secure software development practices to prevent or significantly mitigate the consequences of the attack. These approaches act as a safeguard against vulnerabilities like the ones seen in the incident.

- **Defensive Input Handling**

Rigorously validating and sanitising all inputs ensures that an application only accepts verified and expected data. This method prevents SQL injections by filtering out malicious input content. In the Optus API, there was no limitation on input parameters such as customer ID or phone number. The system blindly returned data from any ID. Proper verification would have checked the authorisation of the user to make such requests IDs fall within certain formats. In web application this method primarily helps prevent Cross Site Scripting (XSS). This would allow the application to prevent any suspicious SQL queries. If this was implemented in the Optus API, it would not have accepted an unauthorised user's request which would thwart the attack.

- **Secure APIs and Session Management**

One of the most glaring and fundamental issues with the Optus API was its lack of authentication. One of the fundamental secure software development practices is to never expose APIs without proper authentication. Every API endpoint must require an API key or other secure authentication method to ensure the user is verified to access the data. Robust session management would ensure tokens are valid, expire appropriately, and are authenticated on each request would have immediately prevented an anonymous attacker from retrieving unauthorised data. No API serving customer data should be “open to the world.” Leaving an API open without requiring proper authentication can lead to sever consequences such as those outlined in this report. Another method to secure APIs is by implementing access control rules. This means each customer exclusively retains access to their own personal data and rate limiting to block network attacks. Additionally, efficiently handling resources such as memory and sessions can prevent the exploitation of buffer overflows or improper memory usage.

- **Vulnerability Mitigation**

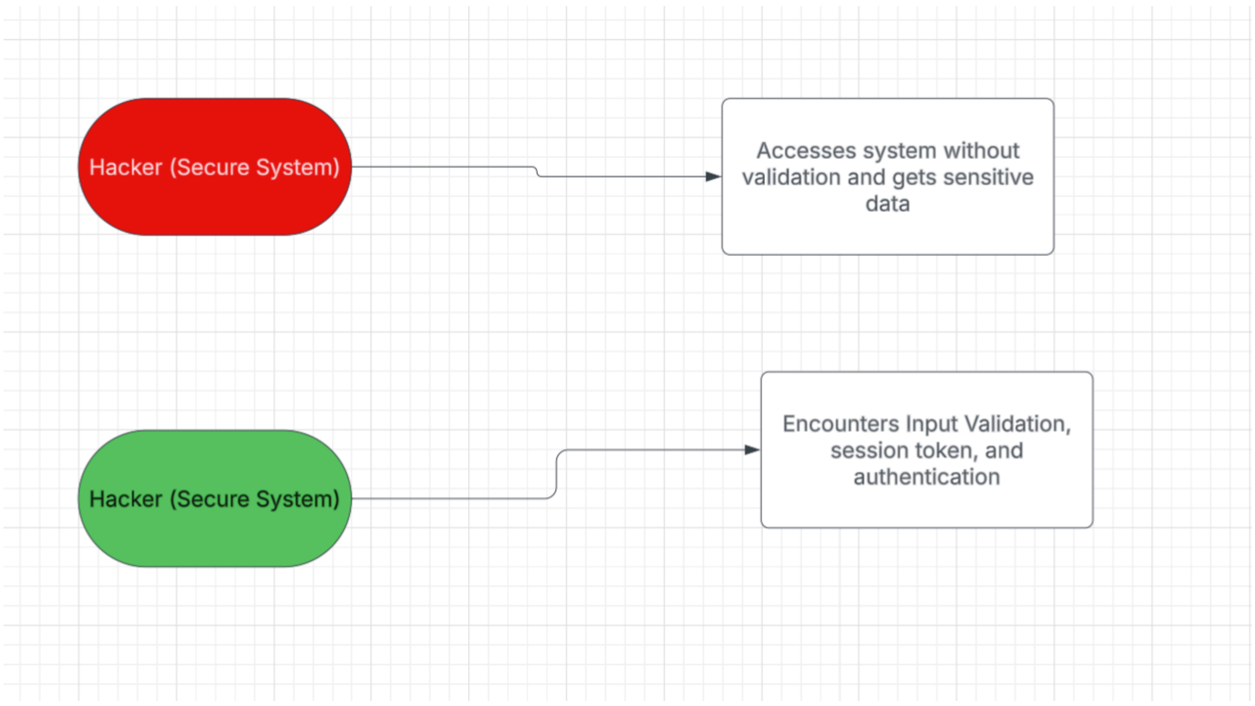
Implementing strict authorisation logic (e.g., verifying the requestor’s identity against the requested customer id) would have helped prevent unauthorised and malicious data handling. This aligns with Open Web Application Security Project’s (OWASP) guidelines for preventing broken authentication and access control. In fact, the Optus cyberattack is an access of Broken Object Level Authorisation (BOLA), which OWASP ranked the number 1 API security risk. To prevent such issues developers, employ frameworks that employ these checks universally, and provide unit tests for authorisation on sensitive functions. Additionally common web vulnerabilities such as Cross Site Scripting should be addressed with defensive input handling. To prevent Cross Site Request Forgery, applications should make use of unpredictable CSRF tokens, ensuring only legitimate users can perform actions.

Following these secure software development practices such as strict input validation, authenticated API access, and robust vulnerability mitigation would have either limited or outright prevented the effects of the Optus breach.

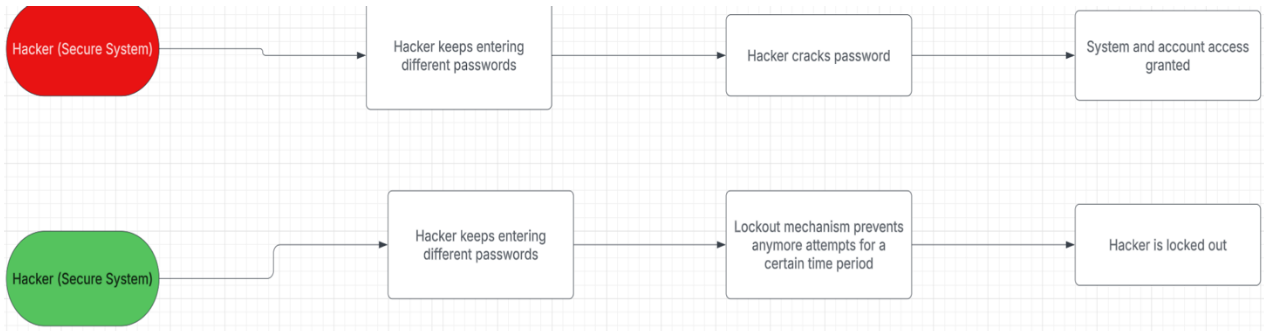
Broader Benefits of Secure Practices

- **Examples**

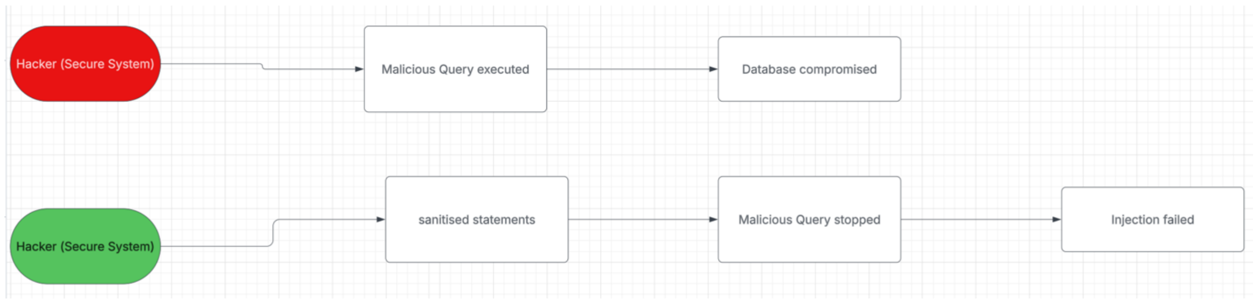
1. Hacker attempts to access system


Hacker (Secure System)
Accesses system without validation and gets sensitive data
Hacker (Secure System)
Encounters Input Validation, session token, and authentication

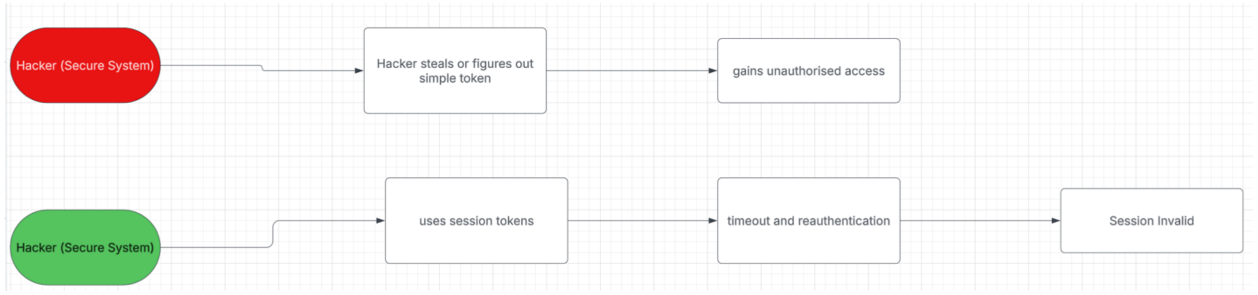
2. Brute Force Hacking


Hacker (Secure System)
Hacker keeps entering different passwords
Hacker cracks password
System and account access granted
Hacker (Secure System)
Hacker keeps entering different passwords
Lockout mechanism prevents anymore attempts for a certain time period
Hacker is locked out

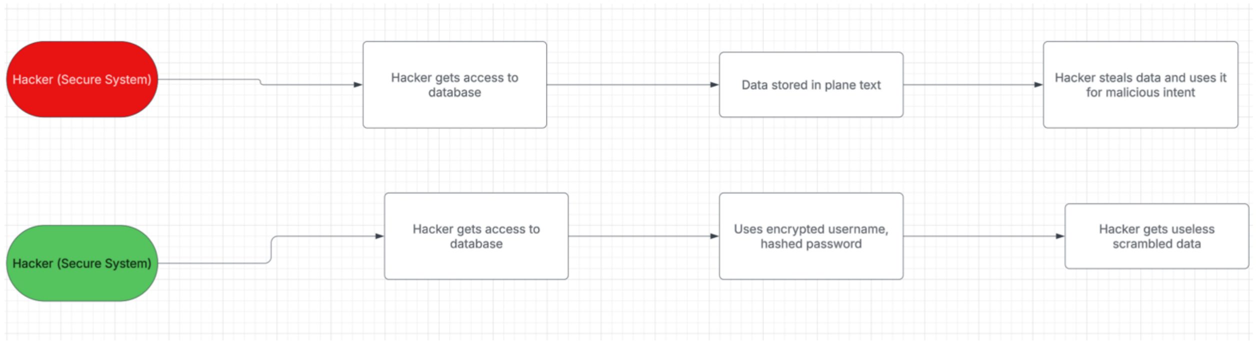
3. SQL Injection



4. Session Hijacking


Hacker (Secure System)
Hacker steals or figures out simple token
gains unauthorised access
Hacker (Secure System)
uses session tokens
timeout and reauthentication
Session Invalid

5. Data Theft/Corruption


Hacker (Secure System)
Hacker gets access to database
Data stored in plane text
Hacker steals data and uses it for malicious intent
Hacker (Secure System)
Hacker gets access to database
Uses encrypted username, hashed password
Hacker gets useless scrambled data

- **Introduction**

Adopting secure development practices helps avoid breaches and protects businesses from unwanted leaks, stakeholder dissatisfaction, reputation. Following the notorious Optus incident, we can clearly see how a commitment to secure development would have helped prevent this controversy.

- Improved Products and Services

By baking security into the development of software products and services, it often leads to higher quality products and services. By addressing potential design flaws, developers catch bugs that might have otherwise caused the system to fail or not function to a suitable level. Secure development leads to an improved reliability of the software leading to a more robust and resilient system. This ensures that developers can identify and mitigate vulnerabilities early, reducing the likelihood of costly post breach development and redeployment. This would benefit the user as they are less likely to experience crashes, malfunctions, or any other issues, thus leading to a significantly improved user experience. If Optus had undergone thorough security review and they would have fixed the vulnerable API, the product would have been significantly trustworthy and safe. Secure features such as robust authentication, hardware accelerated security (such as memory encryption), etc often help companies differentiate themselves from competitors which becomes a selling point. In contrast having insecure products can leave to a lost reputation and consumer mistrust. In essence implementing secure software features helps create a better product for the users and increase reputation.

- Enhanced Productivity and Work Practices

While one might think implementing security practices into development might be counter intuitive for enhanced productivity. Security breaches can lead to a lot of internal stress from workforce and impulsive decisions being made from executive branches, which can overall further ruin the company’s reputation. This will also reduce overtime for workers which would improve the company’s work ethic. According to industry data, fixing a bug in production can cost exponentially more than it would to simply have implemented the appropriate fixes in the first place. This also prevents a product from having to be taken down and redeployed down the line which would require significant effort from the development team and would lose the company active users for the entire period and many users will leave due to lack of trust. In a Secure Software Development Life Cycle (SSDLC), developers often adopt automation (such as automated scans, security linters) that will improve efficiency. The result is faster development with fewer disruptions. Additionally, building an awareness of security quality standards to developers makes developers and QA teams communicate more about potential issues the program could encounter and how to solve them. This also means the responsibility of the program meeting the standards falls upon every party involved. This mindset helps boost productivity and can lead to a greater collaboration effort. From Optus’ perspective this would have saved them from the PR disaster and would have saved the company from making impulsive decisions that could have potentially led to even greater problems. Investing in secure software practices upfront would have been significantly more cost effective in the long run and would save the company from the PR crisis. Furthermore, implementing secure software practices would reduce stress and overtime from employers and empower them to focus on innovating rather than fixing.

- **Increased Trust and Stakeholder Confidence**

When a corporation consistently implements and demonstrates strong security practices, it earns trust from their stakeholders. Users have become increasingly conscious of how their data is handled as our lives have become more reliant on software. By investing and implementing in measures to help safeguard their data, a user feels the company has prioritised their privacy. Users can be assured that their data is in safe hands which reduces concern for what might be happening behind the scenes. Using secure development practices such as encryption, access controls, etc assures stakeholders that their software system is safe and reliable while providing them with the service. In contrast a breach significantly undermines trust and could permanently alienate stakeholders. The Optus breach provides a cautionary tale of what happens when stakeholder confidence and trust is lost. Three years on from the breach and their reputation is yet to be fully recovered with customers increasingly criticising decisions being made by executives at Optus. Developing reliable and secure software also opens the door to partnerships and interest from high value workers. Furthermore, developing secure software will also reduce government intervention which can interfere with innovation and developments. In summary, implementing secure development practices leads to increased business trust and interactivity. Customers are more likely to stay loyal and partners are more willing to collaborate and there will be less government intervention

Conclusion

The Optus cyberattack serves as a stark reminder that security must be embedded in software development. Secure coding, authentication mechanisms, and proactive security audits could have prevented this breach. Businesses that invest in security not only protect user data but also gain customer trust and regulatory compliance. The case of Optus demonstrates that the cost of preventing a cyberattack is far lower than the cost of recovering from one.

References

Australian Communications and Media Authority. (2022). *Optus data breach investigation: Regulatory response and implications*. Retrieved from https://www.acma.gov.au

Australian Government - Department of Home Affairs. (2022). *Cybersecurity report on major data breaches in Australia*. Retrieved from https://www.homeaffairs.gov.au

Australian Cyber Security Centre. (2022). *Optus cyberattack analysis: Lessons and response strategies*. Retrieved from https://www.cyber.gov.au

Hunt, T. (2022). *Optus data breach: Analyzing the vulnerabilities and attack surface*. Retrieved from https://www.troyhunt.com