International Journal of Scientific Research and Engineering Development-– Volume 7 Issue 2, Mar-Apr 2024
Available at www.ijsred.com
RESEARCH ARTICLE OPEN ACCESS

Security of Backend Systems: Advanced Methods for Ensuring Data Protection

Mykyta Machekhin
Senior Software Engineer, Curbside Technologies Inc. US, New York City
E-mail: machekhinn@gmail.com

Abstract:

In May 2023, the Irish Data Protection Commission (DPC) imposed a $1.2 billion fine on Meta for transferring European user data to the US without adequate protection mechanisms.

In June 2014, Code Spaces, a company providing a SAAS code-hosting service, ceased to exist after data and backup versions of all their clients' data were deleted by hackers.

There are a huge number of such examples. Currently, cumulative fines from the GDPR for web service security issues have reached almost $5 billion. This shows how important it is to build a server security system from the very beginning and adhere to best practices

The purpose of the work is to consider modern methods aimed at ensuring data protection. The methodological basis is the scientific works of both domestic and foreign authors.

Keywords: backend, software, backend security, IT, digitalization, modern technology.

Introduction

Data warehouses are an integral component of today's digital reality, storing a huge amount of confidential information: from personal data and financial transactions to intellectual property. With the constant growth in the volume and value of data, the attractiveness of databases to cybercriminals is only increasing. This makes database security a critical issue for organizations of all sizes that seek to protect their valuable data from unauthorized access, leaks, and other cyber threats [1].

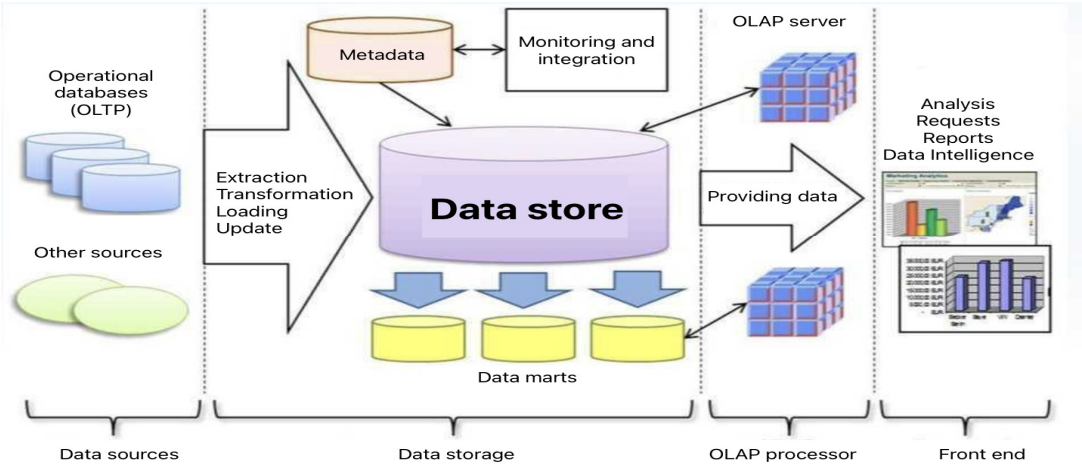

Fig.1. Data storage structure

Database security plays a key role in protecting a company's digital assets, ensuring data integrity and availability, and guaranteeing the confidentiality of customer and partner information. It also helps you comply with industry regulations such as GDPR, HIPAA, and PCI-DSS. Developing a database security strategy helps achieve the following organizational goals:

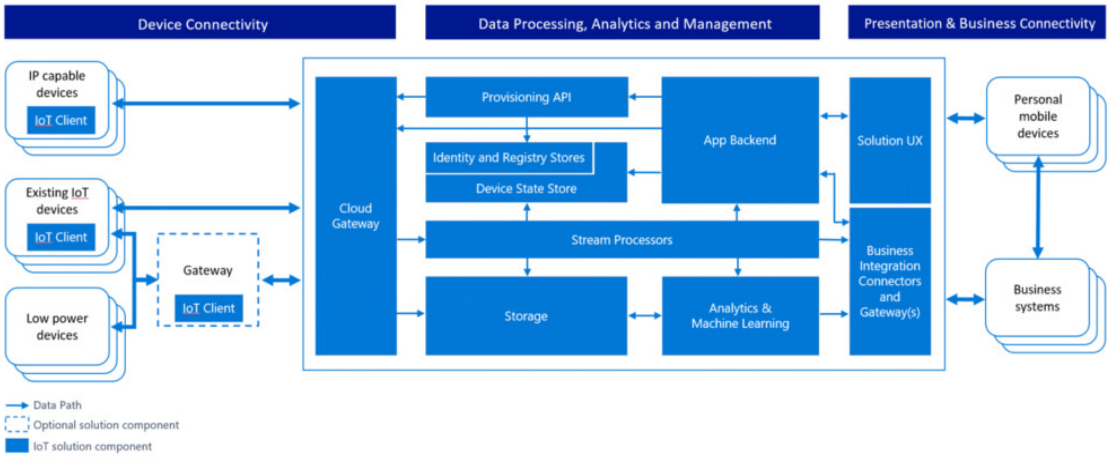

Fig.2. Database security

Data from IBM's 2020 Cost of Data Breach Report shows that half of attackers are seeking financial gain. They can steal valuable data, infect computers with ransomware, or hack them. Among the various categories of data, the most at risk is customer personal information (PII), which accounts for 80% of all data. The average cost per record reaches $175, which is a strong incentive for hackers to attack databases. Attacks primarily target compromised credentials, cloud misconfigurations, and vulnerabilities in third-party software.

Although most governments recognize hacking as a criminal activity, authorities often resort to hiring hackers for “cyber warfare.” These military operations are aimed at compromising targeted organizations or individuals, obtaining intelligence and industrial secrets, disrupting the functionality of critical infrastructure, and conducting disinformation campaigns. Nation-state entities are the most dangerous because they are well-funded and have the highest level of technical expertise.

Hacktivists are another highly motivated group of attackers. They abuse computer systems or networks to draw public attention to something considered valuable, such as freedom of information or human rights. Hacktivists typically provide unauthorized access to a website to display banners with political slogans. Another popular type of malicious behavior is distributed denial of service (DDoS) attacks.

Despite the huge number of cyber attacks that occur every day, they can all be classified into several categories. The Open Web Application Security Project (OWASP) identifies 10 types of the most common security threats.

- Among all the threats, the most destructive is the SQL injection (SQLi) attack, in which a SQL query is injected through the input data field. These attacks include various methods. Union-based SQLi abuses the UNION keyword to retrieve data from other tables in the database. Error-based SQLi causes the database to produce an error message containing information about the structure of the database. This information can be obtained by observing the server's response time to certain SQL queries or by analyzing the results of a Boolean query. Fortunately, despite its dangers, SQLi can be easily fixed by experienced backend developers.
- The second most common vulnerability that attackers prefer to exploit is authentication violation. To gain unauthorized access to the database, attackers can use compromised credentials, perform automatic brute force, or select a password from a list of default combinations. Another popular tool is the dictionary attack, which uses only the most likely entry keys.
- Violations of access rules, which should grant users rights to read and change the database, can be another method used by hackers. To gain access, third-party users can change the URL (for example, http://website.domain/user/ to http://website.domain/admin) or manipulate the metadata through reproducing or modifying the JSON Web Token (JWT). There may also be a misconfiguration of Cross-Origin Resource Sharing (CORS).

- Using insecure deserialization by passing user-modified sequential byte streams to the server is a complex but effective hacking tool. Converting processed byte streams into objects can cause errors or, worse, lead to the server being infected with a virus [2].

1. Compliance with regulatory requirements and data protection laws.

Given the critical importance of database security, organizations must emphasize adopting best practices and comprehensive solutions to protect their data from ever-evolving cyber risks.

For effective data security, organizations must pay special attention to protecting three key elements: database connections, data in storage (data at rest), and data in transit. Each of these aspects presents unique challenges that can be overcome using a combination of best security practices and modern technology.

Securing Database Connections Database connections are potential entry points for attackers trying to gain unauthorized access to your data. To ensure the security of your database connections, you must take the following steps:

Use of encrypted communication channels: The use of encryption protocols such as SSL/TLS secures communications between clients and the database server, preventing eavesdropping and man-in-the-middle attacks. Network access control: Restricting network access to the database server to trusted IP addresses or sources minimizes the attack surface and reduces the risk of unauthorized access. Secure authentication and authorization: Using strong authentication methods such as multi-factor authentication (MFA) ensures that only authorized users have access to the database server. Additionally, the use of role-based access control (RBAC) provides appropriate permissions based on each user's role and responsibilities.

Protecting data in storage (data at rest) Data stored in databases is especially vulnerable to unauthorized access and information leaks. To increase data security at rest, consider the following best practices:

Applying Data at Rest Encryption: Encrypting sensitive data stored in databases makes it unreadable without the appropriate decryption keys, protecting in the event of a breach or unauthorized access. Choosing a strong encryption algorithm and proper key management methods optimizes security. Secure encryption key storage: Keeping encryption keys separate from encrypted data, preferably in a dedicated key management solution or hardware security module (HSM), provides enhanced security and access control features. Applying Database Security Updates: Keeping your database software up to date with the latest security updates allows you to fix potential vulnerabilities and protect your data at rest from exploitation.

Data transferred between client applications and databases is at risk of interception, eavesdropping, and tampering. To increase the security of transmitted data, the following measures must be taken:

Use of encrypted communication channels: As with database connections, using SSL/TLS encryption protocols to protect data transferred between client applications and the database server provides an additional level of security. Ensuring API Security: When using APIs to communicate with database servers, you must ensure strong authentication, access control, and input validation to prevent unauthorized access and data manipulation. Data Transmission Monitoring: Continuously monitoring data transmissions for anomalous behavior or unauthorized access attempts can identify potential security breaches or attacks [3].

2. Some of the most popular homeland security risks and what you can personally do to prevent them.

1. Data injection is a method through which an attacker makes use of requests to compromise servers of their web application, whereby requests are made from this system regarding the desire to obtain confidential data. Without the verification of the origin and authenticity of the request, the system will simply process the request carelessly and give an attacker all the information he needs. Thus, a much better way to avoid this injection of data would be to ensure that internal

applications do not receive, and do therefore deny, input offered by unauthorized or untrusted sources for processing. This is to mean that, in this case, there will be blocking or simply denial of requests coming from invalid sources.

2. Access control errors: In the ACLs, what were detected were error configurations bound to produce differential access to personal web applications for different categories of users. For instance, the team members were to have more access than regular users. There is sensitive data in web applications that is best seen only by your team. Wrong settings in personal access control lists open up the door for unauthorized access; hence, allowing the attacker to enter through the windows that would have been closed. This internal security risk is quite common because people often forget about their access control lists. The next criterion to be considered regards access control. There is a need for the access control lists to be periodically reviewed, and this is to make sure that all parties that are using web applications have appropriate access with minimal risks. It is also necessary to prioritize access to the most valuable resources to prevent possible intrusions by attackers.

3. Error in the setting of the software. It is indicated that the activities of private web applications at the front end heavily depend on the security functions of the back end. Thus, the setting up of errors in the back-end interface could result in failures within the interface of the front-end, thereby leading to breaches that could expose the company's top secret. If any part of the backend web application is inoperative, for example, an error message will appear in the front end. Error messages in the frontend, of course, can give out very sensitive information such as data paths that can be used by attackers to compromise a company's system. Effective management of the information, as indicated in the error messages, can represent an important way to keep away from the risks connected with incorrect software settings. All the backend operations- programming language and the webserver should be optimized in a way that messages do not pop up with sensitive information.

4. Authentication: No such need arises when accessing any of the internal web application components of any given company.

Access to the following is required to be authenticated:

- console/operating system level
- database level

In the same way, access to operating system login credentials is to be granted. At the same time, after the user logs in, minimal vulnerability at the same level as the operating system opens doors You may restrict logins by only your selected users and IP addresses; that means you will be able to assure security by your authentication system. You can make use of HTTP authentication in development. Besides, you have automated systems that can detect attacks on the network of your company.

5. Outdated software components. A web application consists of several software elements that provide its functionality. Each of these components plays a unique role in the efficient operation of a web application. If one component is vulnerable, it can create a vulnerability for other elements of the application.

6. Disclosure of Sensitive Data To improve the user experience on your My Website, native applications may temporarily store information that users create. It is expected that access to such data will be limited to relevant users only. However, hackers can gain unauthorized access to the folder containing the information if it is not properly secured and use the data for their own malicious purposes.

7. Lack of vulnerability scanning There may be vulnerabilities in a company's web applications that may go undetected. A company's network may appear reliable on the surface, but there may be potential risks lurking underneath. Regular vulnerability scanning will help identify any potential risks existing in your web applications. Vulnerability scans should be performed regularly and then carefully reviewed to assess the security of web applications. Take necessary actions based on the findings to prevent possible attacks.

8. Lack of Encryption Between Front-End and Back-End Applications The front-end and back-end of a company's web applications may be located at different levels, but they work together to

keep the entire application running. Sometimes the encryption of messages between these parts may be missed [4].

3. Modern methods of data protection and security when developing high-load applications

Ensuring application security is not only a matter of functionality and performance but also the need to protect data from leaks and attacks. Effective security techniques include using various encryption algorithms such as AES and RSA to protect data from unauthorized access.

It is extremely important to choose and implement encryption algorithms to develop high-load applications. User authentication and authorization also play a key role in security. Use strong methods such as two-factor authentication and access tokens to authenticate users and control their permissions in the application.

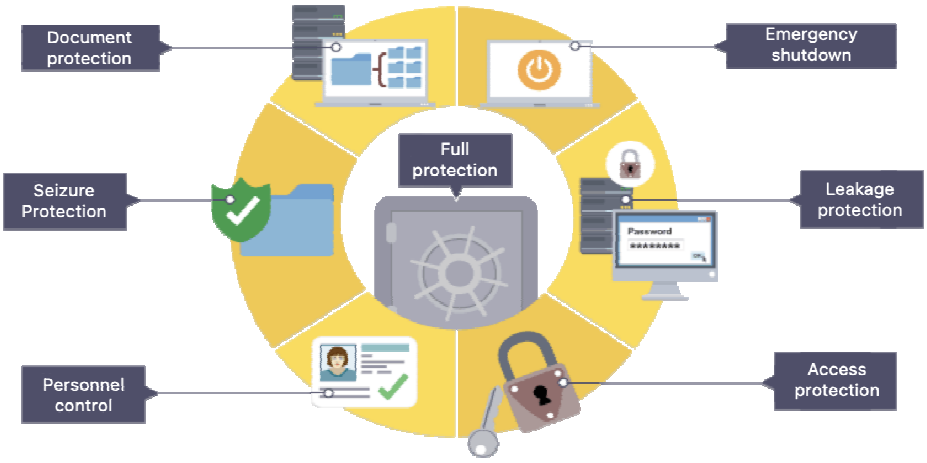

Fig.3. Data leakage protection system

1. Keep software dependencies up to date Outdated libraries and frameworks can create vulnerabilities in a company's internal system. Regularly update and fix software dependencies to ensure the company has the latest security patches. Use package managers to effectively manage dependencies and keep track of any security recommendations for the native libraries you use.

2. Implement user authentication and authorization. Proper authentication and authorization mechanisms are vital to preventing unauthorized access to internal company resources. Use secure protocols such as OAuth or JSON Web Tokens (JWT) to authenticate users and provide role- and permission-based access controls. Implement secure password storage techniques, such as salt hashing, to protect user credentials.

3. Validate and clean up input data. Use input validation libraries tailored to your programming language/framework or implement custom validation logic to ensure data integrity. Never trust user input or sanitize any input that is used in queries or displayed in the front end.

4. Implement speed limiting and brute force protection. Rate-limiting mechanisms are the ones capable of averting the effect of DoS (denial of service) attacks by brute forcing. This involves setting limits of requests per minute/hour for a single user or one IP address. Consider using specialized libraries or services that provide built-in rate-limiting functionality.

The protection from external attacks, for instance, hacking and DDoS attacks, will be able to be sustained with filtering traffic, detection, and mechanisms for prevention of attacks—all the aforementioned in combination with constant application updating, besides patches in place, and frequent use of the database-prepared queries, restricting access rights, and constant backing up of data.

This also includes keeping up with software regularly and keeping dependencies updated through patching vulnerabilities. Keeping up with and updating the dependencies so that vulnerabilities are also patched is very important in the cardinal steps toward reducing security risk. It includes

mechanisms to authenticate and authorize users: input data validation and rate limits on inputs. These mechanisms must also be able to log, monitor, and audit the security events to react quickly in case of a real or assumed threat [5].

5. Web Application Firewalls (WABs): Web application firewalls are a security application positioned between the native web application and potential danger. It scrutinizes incoming requests by filtering those from malicious traffic and evasion of common attacks like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). The implementation of WAF can significantly enhance the security of a company's server infrastructure.

6. Enforce strict password policies: The use of passwords, that are not strong enough, remains a vulnerability for access by unauthorized. Therefore, there is a strict password policy adhered to, where the inclusion of the uppercase and lowercase letters, numbers, and special characters is part of it. Also, encourage regularly updating passwords and avoid reusing them across different platforms.

7. Continually back up the server infrastructure to ensure that, in the event of a security breach or system failure, there is always a valid set of company data. Further, test the restore process from time to time to confirm the backup and that it was a successful process. A proper backup and recovery strategy is key to minimizing possible incidents [6,7].

Conclusion

Thus, protecting the server infrastructure of your application becomes a key factor in ensuring overall system security. By following these recommendations, the company will significantly reduce the likelihood of security vulnerabilities and ensure the protection of user data.

Implementing these best practices will not only strengthen the security of your application but also earn the trust of users. Stay alert, continue to actively update your knowledge, and pay particular attention to ensuring the security of your internal systems.

References

1. How to create secure user authorization using UUID. [Electronic resource] - Access mode: https://dou.ua/forums/topic/34491/ – (accessed 25.01.2024).
2. Backend Security Best Practices. Ensure the Safety of Your Product! [Electronic resource] - Access mode: https://www.ideamotive.co/blog/backend-security-best-practices – (accessed 25.01.2024).
3. Best practices for internal security. Ensure the safety of your product. [Electronic resource] - Access mode: https://appmaster.io/ru/blog/peredovye-metody-obespecheniia-bezopasnosti-baz-dannykh – (accessed 01/25/2024).
4. 8 Backend Security Risks and How to Prevent Them. [Electronic resource] - Access mode: https://www.makeuseof.com/backend-security-risks-and-preventions/ – (accessed 01/25/2024).
5. Best Practices in Backend Security. [Electronic resource] - Access mode: https://www.codingdrills.com/tutorial/backend-development-tutorial/best-practices-in-backend-security – (accessed 01/25/2024).
6. 10 Advanced Tips for Securing Your Backend Infrastructure. [Electronic resource] - Access mode: https://opensudo.org/10-advanced-tips-for-securing-your-backend-infrastructure/ – (accessed 01/25/2024).
7. 5 tips to secure your backend application. [Electronic resource] - Access mode: https://dev.to/tolustar/5-tips-to-secure-your-backend-application-446e – (accessed 25.01.2024).

