William Austin

07/06/2023

WEB 335 - Discussion 7.1 - Database Security

Web security has always been a vital part of any web development project. There are entire degrees focusing on digital security, and we can only touch the service of what all is necessary to secure our databases. However, even that basic knowledge can help us design our databases with 'best practices’ in mind. Here we will discuss Mongo’s ideas around securing data, users, and queries, as well as discuss encryption methods and password recommendations.

First, let’s tackle securing MongoDB users. MongoDB recommends a database “enable authorization and enforce authentication” (Chodorow, 2017). Authentication verifies who the user is, and authorization determines what a user can do. We want to create roles for anyone with access to our database and authorize them to operate only within their roles. After we’ve given ever user a role we want to create login credentials for each user and limit each user’s access to only what their role requires. Ideally, this means that every person is given specific access to what they need, and never general access to the whole database. To that end, it is recommended that we regularly audit users accounts. This entails a periodic review of who has access to what and enforces a policy of ‘least access’(MongoDB), meaning each person is given access ONLY to the parts they need.

Next, let’s look at securing MongoDB queries. We want to make sure that any user input our database receives is sanitized and validated. Sanitizing input means removing or encoding potentially harmful or unwanted characters from the input. (MongoDB) Basically, you want to make sure the input isn’t interpreted as code. This might mean removing special characters that have meaning in a coding language like JS. By validating input we mean making sure that the input is something we expect. Whether that is format validation, or length validation, or even type validation. We want to make sure our queries are valid and not malicious code. The idea here is once again to prevent injection attacks and data corruption.

Now let’s look at securing data. MongoDB again recommends access control. Give every user access to what they need, not everything on the database. It is also recommended that you regularly back up your data through a robust backup strategy. Keep MongoDB up to date. (This goes for anything). The newest security features are always part of the latest update, so be sure your MongoDB is updated regularly.

Another recommendation for protecting data is encryption. Encryption basically transforms data into illegible text unless you have the appropriate ‘key’. Encryption at rest means encryption when the data is stored in a stable state (i.e. not being transferred) (Keary, 2022). This is a level of encryption when data is on hard drives. This ensures that data isn’t accessed even when someone steals the physical hard drive. Without the key, no luck, buttercup. Encryption in transit refers to encryption as the data is moving across the web. It protects data from being intercepted during travel.

Finally, passwords. Mongo recommends the use of strong, complex passwords with a combination of lower-care and upper-case letters, special character, and numbers. (MongoDB). They recommend enforcing password expiration, where users must change their passwords at given time intervals. Finally, they advise a clear lockout policy. Let it be known how many attempts a user has to gain access before they are locked out.

Security is crucial to success and attacks can come in many forms. If we follow theses best practices we limit our exposure to such attacks and ensure a safer database for everyone.

SOURCES

Chodorow, K. (2017). *MONGODB: The definitive guide*. O’REILLY MEDIA, INC, USA.

Keary, T. (2022, April 5). *How to secure mongodb step-by-step & best mongodb monitoring tools*. Comparitech. https://www.comparitech.com/net-admin/secure-mongodb/

Mongo Atlas. (n.d.). *7 best practices for mongodb security*. MongoDB. https://www.mongodb.com/features/security/best-practices