Question 3

10 marks

600 words

Authentication is proving the identify of a person to help control access for websites or applications.

Assuming the admin application is hosted on an intranet with admin user accounts, basic authentication could be implemented. This HTTP method encodes users passwords, before validating it on the server to allow access. However, for more protection I would recommend digest authentication for admin applications. This uses the SHA-3 hash to encrypt passwords along with a ‘nonce’, which is provided by the servers ‘unauthorised’ response. On return, the stored password is encrypted with the same hash and local copy of the ‘nonce’ and is then compared against the users credentials. If the hash values are equal, authenticity is confirmed and the client is granted access. This is more secure as the password itself is never transmitted.

Digital certificates could be used authenticate clubs members. This binds a public key to the users email address or network credentials, creating a certificate which can be used to authenticate a user on the server. However, if possible I would recommend using sessions and authenticate users via their OU account as authenticates they are OU students and removes the strain of password management.

To secure data in transit for basic and digest authentication, the TLS protocol which is closely based on SSL, should be adopted and any URL redirected to https://. This forces the browser to request a secure encrypted connection, making an attack more difficult as they would need the services private key to decrypt data. TLS could also be configured to raise alerts if repeat requests were made or a change in the data mid transit, both suggesting and exposing attacks.

However, for this to work the server has to provide its TLS certificate so the browser can authenticate the server itself, before encrypting the payload using the servers public key. Checking the servers certificate prevents data being sent to a unknown source, as the authentication of the server has be validated by a third party.

Securing authentication data at rest is equally important. Raw passwords should not be stored; they should instead be an encrypted combination of a hash of the password with a ‘salt’ of randomly generated text. This would prevent theft of sensitive data as an attacker couldn’t read the passwords.

Additionally, a separate database could be used to store sensitive data with stricter permissions and enhanced security, both physically and via configuration. This would reduce the impact of a breach if the main database was targeted, as the attacked wouldn’t be able to gain access to the more secure databases.

All back up databases should be equally protected and system patches kept up to date, protecting the software from weaknesses that attackers could exploit.

Authorisation is giving users certain permissions and access, based on who they are and what they are allowed to do.

The club website could have two levels of authorisation, one for general users and one for club members. All users could view planned walks, news and other events, but only club members would be authorised to propose their own walk events. Additionally, only members would be authorised to view and edit their personal details.

The admin application could group users, authorising some groups with greater privileges. For example, a generic group may allow every admin member to approve and create walk events, however only management groups would be allowed to edit club members details.

Authorisation is important to control access, protect individuals rights to privacy and encourage trust of the application. For example, authorisation reassures club members that walk events have been authorised by the admin team otherwise anybody could create fake events. Authorisation for the admin application ensures specific actions are only permitted by authorised people, preventing a rogue admin member making every member inactive or editing member consent to receive emails.

(636 words)

Add references