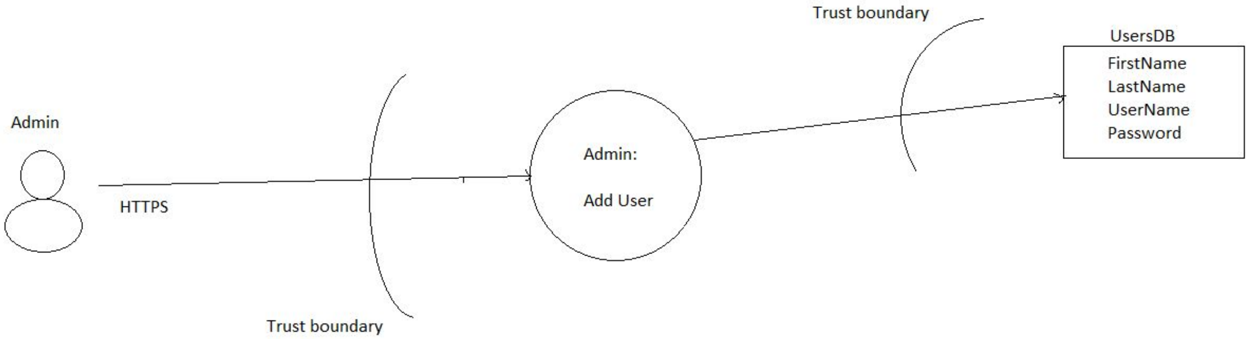
D3CRYPT

Threat Modeling

Overview

1. **Scope**:
   a. Take one feature/functionality at a time. Note down intended/unintended actors of that feature
2. **Visualize**:
   a. Draw the big picture. Consider,
      i. How the actor will interact with the feature. Eg. web interface, desktop application, APIs
      ii. What data is moving through the flow? How?
      iii. What are the trust boundaries as per the developer’s original design
3. **Enumerate threats**:
   a. STRIDE
4. **Remediate**:
   a. Controls
5. **Document**:
   a. What threats have been considered, what are their risk levels, if/how have they been remediated and if the controls have been verified (code review, penetration testing)

Let’s threat model https://demo.testfire.net/login.jsp



Category	Description	Controls
Spoofing	Can I impersonate another user/entity	Strong Authentication
Tampering	Can I modify sensitive data	Encryption, Hashing
Repudiation	Can I repudiate my actions	Authentication, Authorization, Logging (along with other controls)
Information Disclosure	Can I get unauthroized access to sensitive data	Encryption
Denial of Service	Can I DoS the system	Rate Limiting, Throttling, WAFs etc.
Elevation of Privileges	Can I elevate my privileges	Strong Authorization

Threats (not exhaustive)

- **Unauthorized addition of users (web)** -
  - CSRF on add user Spoofing, Repudiation
  - Authentication bypass on admin Spoofing
  - IDOR by not-privileged user Elevation of Privilege, Repudiation
  - Session fixation on admin Spoofing
- **Tampering new user details on network** - Tampering
  - Browser to App,
  - App to DB
- **Steal admin cookies** - XSS Spoofing
- **Admin account takeover** - Spoofing
  - Password brute force,
  - Password Spraying/guessing,
  - SQL injection
- **Compromise DB credentials on network (App to DB)** - Packet capture on network Information Disclosure
- **Denial of service** - Login brute force, DB exhaust via user add function abuse Denial of Service
- **Unauthorized addition of uses (DB)** - Repudiation, Spoofing
  - No auth to DB on internal network
  - Weak DB creds on internal network

Controls

- **Unauthorized addition of users (web)** -
  - CSRF on add user - CSRF token
  - Authentication bypass on admin - protect against injection attacks, enforce authentication on all functions/pages
  - IDOR by not-privileged user - authorization check on add user
  - Session fixation on admin - assign new session tokens after login
- **Tampering new user details on network** -
  - Browser to App - TLS connection, Strict transport security header
  - App to DB - TLS connection, Message Integrity Checks
- **Steal admin cookies** - XSS - Input validation & output encoding
- **Admin account takeover** -
  - Password brute force - Account lockout after a few failed attempts (4-6)
  - Password Spraying/guessing - 2FA
  - SQL injection - Input validation, Parameterized queries, secure frameworks for you language etc.
- **Compromise DB credentials on network (App to DB)** - Packet capture on network - TLS connection
- **Denial of service** - Login brute force, DB exhaust via user add function abuse - CAPTCHA on login, Throttling DB writes etc.
- **Unauthorized addition of uses (DB)** -
  - No auth to DB on internal network - TLS 2 Way Auth between App Server to DB, Implement Auth for DB access
  - Weak DB creds on internal network - TLS 2 Way Auth between App Server to DB, Implement Auth for DB access, if not possible, then secure passwords managed by Password Vault solutions

Validate Controls

- Code Review
    - SQLi
        - Files: https://github.com/hclproducts/AltoroJ/blob/AltoroJ-3.2/src/com/ibm/security/appscan/altoromutual/servlet/LoginServlet.java https://github.com/hclproducts/AltoroJ/blob/AltoroJ-3.2/src/com/ibm/security/appscan/altoromutual/util/DBUtil.java
        - Code Snippet

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
                    //log in
                    // Create session if there isn't one:
                    HttpSession session = request.getSession(true);


                    String username = null;

                    try {
                              username = request.getParameter("uid");
                              if (username != null)
                                        username = username.trim().toLowerCase();

                              String password = request.getParameter("passw");
                              password = password.trim().toLowerCase(); //in real life the password usually is case sensitive and this cast would not be
done

                              if (!DBUtil.isValidUser(username, password)){
                                        Log4AltoroJ.getInstance().logError("Login failed >>> User: " +username + " >>> Password: " + password);
                                        throw new Exception("Login Failed: We're sorry, but this username or password was not found in our system.
Please try again.");
                              }
                    } catch (Exception ex) {
                              request.getSession(true).setAttribute("loginError", ex.getLocalizedMessage());
                              response.sendRedirect("login.jsp");
                              return;
                    }
```

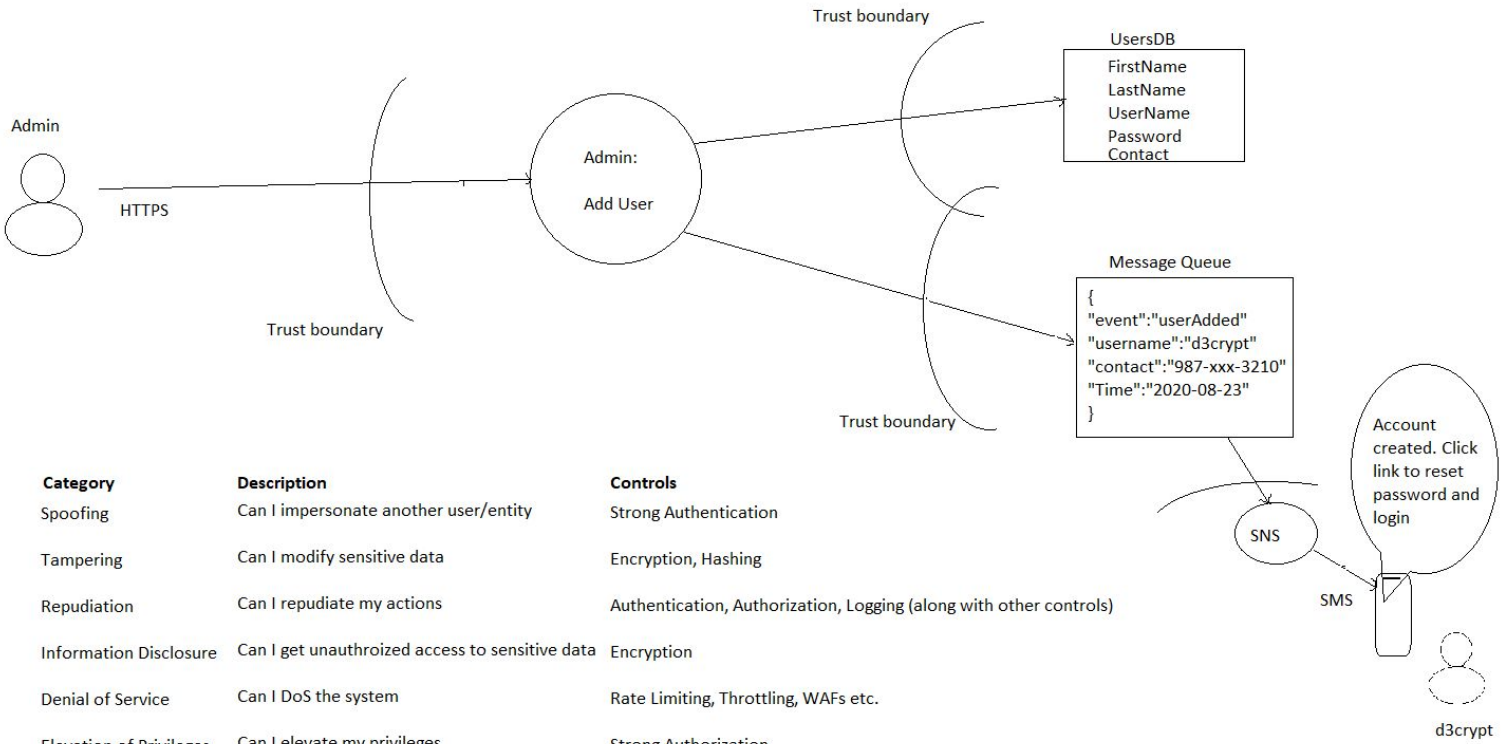
Validate Controls (continued)

- Code Review
    - SQLi
        - Code Snippet

```
          public static boolean isValidUser(String user, String password) throws SQLException{
                    if (user == null || password == null || user.trim().length() == 0 || password.trim().length() == 0)
                              return false;

                    Connection connection = getConnection();
                    Statement statement = connection.createStatement();

                    ResultSet resultSet =statement.executeQuery("SELECT COUNT(*)FROM PEOPLE WHERE USER_ID = '"+ user +"' AND PASSWORD='" + password + "'"); /* BAD
- user input should always be sanitized */

                    if (resultSet.next()){

                                        if (resultSet.getInt(1) > 0)
                                                  return true;
                    }
                    return false;
          }
```

- Penetration Testing (Demo)

Your turn !



```
{
"event":"userAdded"
"username":"d3crypt"
"contact":"987-xxx-3210"
"Time":"2020-08-23"
}
```

Category	Description	Controls
Spoofing	Can I impersonate another user/entity	Strong Authentication
Tampering	Can I modify sensitive data	Encryption, Hashing
Repudiation	Can I repudiate my actions	Authentication, Authorization, Logging (along with other controls)
Information Disclosure	Can I get unauthroized access to sensitive data	Encryption
Denial of Service	Can I DoS the system	Rate Limiting, Throttling, WAFs etc.
Elevation of Privileges	Can I elevate my privileges	Strong Authorization