ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Báo cáo Seminar

USER AUTHENTICATION

Môn học: Mã hóa ứng dụng

CSC15003_22CNTThuc

Sinh viên:
Lê Hoàng Đạt
Nguyễn Hồ Đăng Duy
Phạm Quang Duy

Giảng viên hướng dẫn:
Trương Toàn Thịnh
Lương Vĩ Minh
Mai Anh Tuấn

Mục lục

1 Thông tin sinh viên **4**

2 Giới thiệu **4**

3 Quy Trình **5**
3.1 Gửi thông tin đăng nhập . . . 5
3.2 Xác thực thông tin đăng nhập . . . 5
3.2.1 Kiểm tra dữ liệu đầu vào . . . 5
3.2.2 So sánh . . . 5
3.2.3 Tạo phiên làm việc (session) hoặc token . . . 5
3.3 Phản hồi . . . 6
3.4 Duy trì phiên . . . 6

4 Các phương pháp phổ biến **7**
4.1 Something You Know (Những gì bạn biết) . . . 7
4.2 Something You Have (Những gì bạn có) . . . 8
4.3 Something You Are (Những gì bạn là) . . . 9
4.4 Multi-Factor Authentication (Xác thực nhiều yếu tố) . . . 10
4.5 Behavioral Authentication (Hành vi) . . . 11
4.6 Token (Mã token) . . . 12
4.7 Xác Thực Không Mật Khẩu (Passwordless Authentication) . . . 13
4.8 Xác Thực Bên Thứ Ba (Third-Party Authentication) . . . 14

5 Remote User-Authentication **15**
5.1 Nguyên tắc xác thực từ xa . . . 15
5.1.1 NIST Model for Electronic User Authentication. . . . 15
5.1.2 Các yếu tố xác thực: mật khẩu, token, sinh trắc học. . . . 16
5.2 Các phương pháp áp dụng trong Remote User-Authentication . . . 17
5.2.1 Xác thực hai chiều (Mutual Authentication). . . . 17
5.2.2 Xác thực một chiều (One-Way Authentication). . . . 17
5.3 Sử dụng mã hóa đối xứng trong xác thực từ xa . . . 18
5.3.1 Xác thực hai chiều bằng mã hóa đối xứng. . . . 18
5.3.2 Xác thực một chiều bằng mã hóa đối xứng. . . . 19
5.4 Sử dụng mã hóa bất đối xứng trong xác thực từ xa . . . 19
5.4.1 Xác thực hai chiều bằng mã hóa bất đối xứng . . . 19
5.4.2 Xác thực một chiều bằng mã hóa bất đối xứng. . . . 20
5.5 Federated Identity Management . . . 20
5.5.1 Identity Management (Quản lý danh tính). . . . 20
5.5.2 Identity Federation (Liên kết danh tính). . . . 20
5.6 Personal Identity Verification (PIV) . . . 21
5.6.1 PIV System Model . . . 21
5.6.2 PIV Documentation . . . 21
5.6.3 PIV Credentials and Keys . . . 21
5.6.4 Authentication (Quá trình xác thực). . . . 22

6 Các giao thức và chuẩn hỗ trợ **23**
6.1 Giao thức . . . 23
6.1.1 OAuth 2.0 . . . 23
6.1.2 OpenID Connect (OIDC) . . . 23
6.1.3 Kerberos . . . 24
6.1.4 RADIUS (Remote Authentication Dial-In User Service) . . . 25
6.1.5 LDAP (Lightweight Directory Access Protocol) . . . 25
6.2 Chuẩn hỗ trợ . . . 27
6.2.1 SAML (Security Assertion Markup Language) . . . 27
6.2.2 FIDO (Fast Identity Online) . . . 27
6.2.3 X.509 (Digital Certificates) . . . 28
6.2.4 NIST SP 800-63 (Digital Identity Guidelines) . . . 29

7 Các vấn đề về bảo mật **31**
7.1 Chống tấn công brute-force không hiệu quả . . . 31
7.2 Thông tin đăng nhập yếu . . . 31
7.3 Lộ thông tin tên đăng nhập (Username Enumeration) . . . 31
7.4 HTTP Basic Authentication không an toàn . . . 31
7.5 Quản lý phiên làm việc (Session Management) kém . . . 32
7.6 Chức năng "Ghi nhớ đăng nhập" kém bảo mật . . . 32
7.7 Tấn công SQL Injection . . . 32
7.8 Quy trình đổi và khôi phục mật khẩu không an toàn . . . 32
7.9 Xác thực hai yếu tố (2FA) kém an toàn . . . 32
7.10 Lỗi trong logic xác thực . . . 32

8 Nâng cấp bảo mật **33**
8.1 Nâng cấp chính sách mật khẩu mạnh (Strong Password Policies) . . . 33
8.2 Kích hoạt xác thực đa yếu tố (Multi-Factor Authentication - MFA) . . . 33
8.3 Áp dụng xác thực sinh trắc học (Biometric Authentication) . . . 33
8.4 Cập nhật và vá lỗi hệ thống xác thực thường xuyên . . . 33
8.5 Đào tạo người dùng về an ninh . . . 33
8.6 Sử dụng xác thực thích ứng (Adaptive Authentication) . . . 33
8.7 Theo dõi và phân tích các sự kiện xác thực . . . 34

9 Ứng dụng thực tiễn **35**
9.1 Ứng dụng trong dịch vụ trực tuyến . . . 35
9.2 Thương mại điện tử và thanh toán trực tuyến . . . 35
9.3 Ứng dụng trong doanh nghiệp . . . 35
9.4 Ứng dụng trong giáo dục . . . 35
9.5 Ứng dụng trong các thiết bị thông minh . . . 35
9.6 Ứng dụng trong chăm sóc sức khỏe . . . 36
9.7 Hệ thống giao thông và dịch vụ công cộng . . . 36
9.8 Ứng dụng trong an ninh . . . 36

10 Kerberos Authentication: Kiến Trúc, Hoạt Động & Ứng Dụng **37**
10.1 Giới Thiệu về Kerberos . . . 37
10.1.1 Tổng quan về Kerberos . . . 37
10.1.2 Tại sao lại là Kerberos . . . 37
10.2 Kiến Trúc & Thành Phần Của Kerberos . . . 38
10.2.1 Nguyên lý hoạt động của Kerberos . . . 38
10.2.2 Các thành phần chính . . . 38
10.2.3 Các loại vé trong kerberos . . . 39
10.3 Quy Trình Xác Thực Của Kerberos . . . 41
10.3.1 Bước 1: Xác thực ban đầu (Client → Authentication Server - AS) . . . 41
10.3.2 Bước 2: Nhận vé dịch vụ (Client → Ticket Granting Server - TGS) . . . 41
10.3.3 Bước 3: Truy cập dịch vụ (Client → Service Server - SS) . . . 42
10.3.4 Lưu đồ mô tả quy trình xác thực . . . 42
10.4 Cơ Chế Mã Hóa & Bảo Mật . . . 43
10.4.1 Mã hóa đối xứng trong Kerberos . . . 43
10.4.2 Cách Kerberos ngăn chặn tấn công . . . 43
10.5 Ứng Dụng Thực Tế . . . 44
10.5.1 Kerberos trong Windows Active Directory . . . 44
10.5.2 Kerberos trong UNIX/Linux . . . 44
10.6 So Sánh Kerberos Với Các Cơ Chế Xác Thực Khác . . . 45
10.6.1 Kerberos vs OAuth . . . 45
10.6.2 Kerberos vs NTLM . . . 45
10.6.3 Kerberos vs SAML . . . 45
10.7 Hạn Chế và Hướng Phát Triển Của Kerberos . . . 46
10.7.1 Hạn chế của Kerberos . . . 46
10.7.2 Hướng phát triển của Kerberos . . . 46

11 Kế hoạch thực thi mô phỏng Kerberos **47**
11.1 Mô tả tổng quan . . . 47
11.2 Phân công chi tiết . . . 48
11.2.1 Seminar 1 . . . 48
11.2.2 Seminar 2 . . . 48
11.2.3 Seminar 3 . . . 49

12 Thực thi mô phỏng **51**
12.1 Cài đặt môi trường . . . 51
12.1.1 Cài Đặt OpenSSL, Boost bằng Vcpkg . . . 51
12.1.2 Cấu hình Visual Studio . . . 51
12.2 Cấu trúc dữ liệu . . . 53
12.3 Cấu trúc code . . . 55
12.4 Luồng chương trình . . . 57
12.5 Mã giả - Pseudo code . . . 60
12.6 Mã nguồn dự án . . . 62

Tài liệu tham khảo **63**

1 Thông tin sinh viên

Nhóm gồm có 3 thành viên:

- 22127060 - Lê Hoàng Đạt - 22127060@student.hcmus.edu.vn
- 22127085 - Nguyễn Hồ Đăng Duy - 22127085@student.hcmus.edu.vn
- 22127088 - Phạm Quang Duy - 22127088@student.hcmus.edu.vn

2 Giới thiệu

Mục tiêu: Xác thực người dùng là một quy trình bảo mật quan trọng nhằm xác minh danh tính của cá nhân hoặc thực thể đang cố gắng truy cập vào một hệ thống, ứng dụng hoặc mạng. Quy trình này đóng vai trò như một cánh cổng, đảm bảo rằng chỉ những người dùng được ủy quyền mới có thể truy cập vào thông tin và chức năng nhạy cảm. Điều này rất cần thiết để bảo vệ tính toàn vẹn của dữ liệu và ngăn chặn truy cập trái phép.

Tầm quan trọng:

- **Bảo Vệ Dữ Liệu Nhạy Cảm:** Ngăn chặn truy cập trái phép vào thông tin bí mật.
- **Đảm Bảo Tuân Thủ Quy Định:** Nhiều quy định yêu cầu các biện pháp bảo vệ dữ liệu nghiêm ngặt, mà xác thực người dùng giúp đáp ứng.
- **Cá Nhân Hóa Trải Nghiệm Người Dùng:** Bằng cách nhận diện người dùng, hệ thống có thể điều chỉnh dịch vụ và nội dung theo sở thích cá nhân.

3 Quy Trình

3.1 Gửi thông tin đăng nhập

Thông tin đăng nhập:

- **Tên người dùng (User name)**: Được sử dụng để nhận diện người dùng trong hệ thống.
- **Mật khẩu (Password)**: Một chuỗi ký tự bí mật mà chỉ người dùng biết.
- **Dữ liệu sinh trắc học**: Như vân tay hoặc nhận diện khuôn mặt, cung cấp thêm một lớp bảo mật.
- ...

Truyền tải dữ liệu đến server:

- Gửi thông tin qua phương thức **POST** để tránh hiển thị thông tin nhạy cảm trên URL.
- Bảo vệ dữ liệu bằng giao thức **HTTPS** (mã hóa dữ liệu giữa client và server).

3.2 Xác thực thông tin đăng nhập

3.2.1 Kiểm tra dữ liệu đầu vào

- Kiểm tra các trường input username và password có trống không.
- Mã hóa mật khẩu (Không lưu trữ mật khẩu thô trong CSDL): Sử dụng thuật toán hash (như **BCrypt, Argon2**) để lưu trữ mật khẩu.

3.2.2 So sánh

- Tìm kiếm username trong cơ sở dữ liệu.
- Hệ thống so sánh thông tin đăng nhập đã gửi với thông tin lưu trữ trong cơ sở dữ liệu:
  - Đối với các hệ thống dựa trên mật khẩu, điều này liên quan đến việc băm mật khẩu nhập vào và so sánh với giá trị băm đã lưu.
  - Đối với các hệ thống sinh trắc học, nó liên quan đến việc so khớp dữ liệu sinh trắc học với các mẫu đã lưu.

3.2.3 Tạo phiên làm việc (session) hoặc token

- **Session:** Lưu trạng thái đăng nhập trên server (thường dùng với ứng dụng web truyền thống).
- **Token:** Một chuỗi mã hóa chứa thông tin người dùng (JWT phổ biến trong REST API).

3.3 Phản hồi

- **Cấp quyền truy cập**: Nếu thông tin đăng nhập khớp, người dùng sẽ được xác thực và quyền truy cập sẽ được cấp cho các tài nguyên hoặc chức năng yêu cầu.
- **Từ chối quyền truy cập**: Nếu không khớp, quyền truy cập sẽ bị từ chối và người dùng có thể được yêu cầu nhập lại thông tin đăng nhập. Sau nhiều lần thử không thành công, các biện pháp bảo mật bổ sung có thể được kích hoạt, chẳng hạn như tạm thời khóa tài khoản hoặc yêu cầu khôi phục mật khẩu.

3.4 Duy trì phiên

- Trong các lần yêu cầu tiếp theo, trình duyệt sẽ gửi mã phiên để xác thực mà không cần nhập lại thông tin đăng nhập.
- **Session-based:**
  - Server tạo **session ID**, lưu trong bộ nhớ hoặc cơ sở dữ liệu.
  - ID được gửi về client qua cookie, sử dụng để xác minh trong các yêu cầu tiếp theo.
  - **Ưu điểm:** Bảo mật cao vì lưu trên server.
  - **Nhược điểm:** Tốn tài nguyên với ứng dụng lớn.
- **Token-based:**
  - Server tạo **token** (thường dùng JWT), mã hóa thông tin đăng nhập và gửi về client.
  - Token được lưu trong **localStorage, sessionStorage** hoặc **cookie**.
  - **Ưu điểm:** Phù hợp với ứng dụng quy mô lớn, không cần lưu trên server.
  - **Nhược điểm:** Dễ bị lộ nếu bảo mật client kém.

4 Các phương pháp phổ biến

4.1 Something You Know (Những gì bạn biết)

Định nghĩa: Yêu cầu người dùng nhớ và nhập thông tin.
Ví dụ:

- **Mật khẩu** (Phương pháp sử dụng phổ biến nhất): Một chuỗi ký tự bí mật người dùng tạo ra để bảo mật tài khoản.
- **PIN - Personal Identification Numbers (Mã nhận dạng cá nhân):** Mã số ngắn, thường với thẻ hoặc thiết bị.
- **Câu hỏi bảo mật:** Trả lời các câu hỏi cá nhân đã định trước.

Mức độ bảo mật:

- Thường được coi là mức độ bảo mật trung bình. Mặc dù phương pháp này được sử dụng rộng rãi, nhưng nó cũng dễ bị tổn thương trước nhiều loại tấn công khác nhau như tấn công lừa đảo (phishing), tấn công brute-force và kỹ thuật xã hội (social engineering).

Trải nghiệm người dùng:

- Cung cấp trải nghiệm người dùng cao vì người dùng đã quen với việc nhập mật khẩu và mã PIN. Tuy nhiên, nếu người dùng quên thông tin xác thực của họ, điều đó có thể dẫn đến sự cố truy cập.

Thực hành:

- Chính Sách Mật Khẩu Mạnh:Khuyến khích người dùng tạo mật khẩu phức tạp, bao gồm sự kết hợp giữa chữ hoa, chữ thường, số và ký tự đặc biệt.
- Cập Nhật Định Kỳ: Người dùng nên được nhắc thay đổi mật khẩu định kỳ để giảm thiểu rủi ro truy cập trái phép.
- Cơ Chế Khóa Tài Khoản: Triển khai chính sách khóa tài khoản sau một số lần đăng nhập không thành công nhất định để ngăn chặn các cuộc tấn công brute-force.
- Sử Dụng Trình Quản Lý Mật Khẩu: Đề xuất người dùng sử dụng trình quản lý mật khẩu để giúp họ tạo và lưu trữ mật khẩu mạnh một cách an toàn.
- Bổ Sung Bằng Xác Thực Nhiều Yếu Tố (MFA): Kết hợp "Something You Know" với các yếu tố khác (như "Something You Have" hoặc "Something You Are") để tăng cường bảo mật.

4.2 Something You Have (Những gì bạn có)

Định nghĩa: Phụ thuộc vào các vật dụng vật lý hoặc mã mà người dùng sở hữu.
Ví dụ:

- **OTP - One Time Password (Mật khẩu dùng một lần):** Gửi qua SMS, email hoặc ứng dụng xác thực.
- **Thẻ thông minh:** thẻ vật lý để dùng trong các nơi làm việc an toàn.
- **Mã thông báo phần cứng:** Các thiết bị vật lý tạo mã truy cập.
- **Key Fobs:** Thiết bị nhỏ để cung cấp truy cập và thiết bị, khu vực.

Mức độ bảo mật:

- Thường được coi là có tính bảo mật cao vì nó yêu cầu sở hữu một vật phẩm vật lý khó sao chép hoặc đánh cắp mà người dùng không hề hay biết.

Trải nghiệm người dùng:

- Cung cấp trải nghiệm người dùng ở mức trung bình vì người dùng phải có sẵn vật phẩm thực tế để xác thực. Điều này có thể bất tiện nếu vật phẩm bị mất hoặc không có sẵn.

Thực hành:

- Cập Nhật Định Kỳ Các Mã Bảo Mật: Đảm bảo rằng bất kỳ mã bảo mật hoặc thẻ nào cũng được cập nhật thường xuyên để ngăn chặn truy cập trái phép.
- Đào Tạo Người Dùng: Huấn luyện người dùng cách quản lý thiết bị của họ một cách an toàn và nhận diện các cuộc tấn công lừa đảo có thể làm tổn hại đến các phương pháp xác thực của họ.
- Tùy Chọn Sao Lưu: Cung cấp các phương pháp thay thế cho người dùng trong trường hợp họ mất thiết bị xác thực chính, chẳng hạn như mã sao lưu hoặc thiết bị phụ.
- Bổ Sung Bằng Xác Thực Nhiều Yếu Tố (MFA): Kết hợp "Something You Know" với các yếu tố khác (như "Something You Have" hoặc "Something You Are") để tăng cường bảo mật.

4.3 Something You Are (Những gì bạn là)

Định nghĩa: Sử dụng các đặc điểm sinh trắc học đặc biệt của từng cá nhân. Phương pháp này dựa vào các đặc điểm sinh lý hoặc hành vi độc đáo của người dùng, làm cho nó trở thành một trong những hình thức xác thực an toàn nhất.

Ví dụ:

- **Nhận Diện Vân Tay:** Quét vân tay của người dùng để xác thực danh tính.
- **Nhận Diện Khuôn Mặt:** Sử dụng các đặc điểm khuôn mặt để xác minh danh tính, thường thông qua camera trên điện thoại thông minh hoặc hệ thống an ninh.
- **Nhận Diện Mống Mắt:** Quét các mẫu độc đáo trong mống mắt của một người để xác định danh tính.
- **Nhận Diện Giọng Nói:** Phân tích các mẫu và đặc điểm giọng nói để xác thực người dùng.

Mức độ bảo mật:

- Thường được coi là rất cao, vì các đặc điểm sinh trắc học rất khó để sao chép hoặc giả mạo. Tuy nhiên, chúng vẫn có thể bị tổn thương trước một số loại tấn công (ví dụ: giả mạo bằng hình ảnh hoặc khuôn mẫu chất lượng cao).

Trải nghiệm người dùng:

- Cung cấp một trải nghiệm người dùng trung bình, vì người dùng thường thấy các phương pháp sinh trắc học thuận tiện và nhanh chóng. Tuy nhiên, có thể gặp vấn đề về độ chính xác, đặc biệt trong điều kiện ánh sáng kém hoặc với những người có sự khác biệt về thể chất.

Thực hành:

- Bảo Vệ Dữ Liệu: Đảm bảo rằng dữ liệu sinh trắc học được lưu trữ an toàn và mã hóa để bảo vệ trước các cuộc tấn công.
- Sự Đồng Ý và Quyền Riêng Tư của Người Dùng: Lấy sự đồng ý rõ ràng từ người dùng trước khi thu thập và sử dụng dữ liệu sinh trắc học của họ, và thông báo cho họ về cách dữ liệu sẽ được sử dụng và lưu trữ.
- Tùy Chọn Sao Lưu: Cung cấp các phương pháp xác thực thay thế cho những người có thể gặp khó khăn khi sử dụng hệ thống sinh trắc học (ví dụ: do chấn thương hoặc vấn đề kỹ thuật).

4.4 Multi-Factor Authentication (Xác thực nhiều yếu tố)

Định nghĩa: Là một phương pháp bảo mật yêu cầu người dùng cung cấp ít nhất hai hoặc nhiều yếu tố xác thực khác nhau để chứng minh danh tính của họ trước khi truy cập vào hệ thống hoặc tài khoản.

Ví dụ:

- Mật khẩu + OTP (ví dụ: ngân hàng trực tuyến).
- Quét sinh trắc học + PIN.

Mức độ bảo mật:

- MFA thường được coi là rất cao, vì ngay cả khi một yếu tố (chẳng hạn như mật khẩu) bị lộ, kẻ tấn công vẫn cần vượt qua các yếu tố xác thực bổ sung để truy cập vào tài khoản. Điều này làm giảm đáng kể rủi ro truy cập trái phép.

Trải nghiệm người dùng:

- MFA cung cấp một trải nghiệm người dùng trung bình, vì việc yêu cầu nhiều yếu tố xác thực có thể làm tăng thời gian và công sức cần thiết để đăng nhập. Tuy nhiên, nhiều người dùng đánh giá cao sự an toàn mà MFA mang lại.

Thực hành:

- Giáo Dục Người Dùng: Đào tạo người dùng về tầm quan trọng của MFA và cách sử dụng nó một cách hiệu quả là rất quan trọng. Giao tiếp rõ ràng có thể giúp giảm bớt sự kháng cự và cải thiện tỷ lệ chấp nhận.
- MFA Thích Ứng: Triển khai MFA thích ứng điều chỉnh các biện pháp bảo mật dựa trên hành vi của người dùng, vị trí và thiết bị. Điều này cho phép trải nghiệm mượt mà hơn trong khi vẫn duy trì an ninh.
- Tích Hợp Đăng Nhập Một Lần (SSO): Kết hợp MFA với SSO để đơn giản hóa quy trình đăng nhập, giảm số lượng mật khẩu mà người dùng cần nhớ trong khi đảm bảo quyền truy cập an toàn.
- Đa Dạng Phương Pháp Xác Thực: Cung cấp nhiều tùy chọn xác thực (ví dụ: sinh trắc học, mã thông báo phần cứng, mã SMS) để phù hợp với sở thích của người dùng khác nhau và nâng cao trải nghiệm người dùng.
- Kiểm Tra An Ninh Định Kỳ: Thực hiện các đánh giá định kỳ về việc triển khai MFA của bạn để xác định các lỗ hổng và đảm bảo tính hiệu quả trước các mối đe dọa đang phát triển.

4.5 Behavioral Authentication (Hành vi)

Định nghĩa: Xác thực hành vi là phương pháp xác minh danh tính dựa trên hành vi của người dùng. Phương pháp này sử dụng các mẫu hành vi độc đáo mà mỗi người dùng thể hiện trong quá trình tương tác với thiết bị hoặc ứng dụng, nhằm đảm bảo rằng người dùng là chính họ.

Ví dụ:

- **Động Tác Gõ Phím:** Phân tích mẫu gõ phím của người dùng, bao gồm tốc độ và cách nhấn phím.
- **Chuyển Động Chuột:** Theo dõi cách di chuyển chuột và các tương tác với màn hình cảm ứng để xác định người dùng.

Mức độ bảo mật:

- Xác thực hành vi thường được coi là mức độ bảo mật trung bình, vì nó dựa vào các mẫu hành vi có thể thay đổi theo thời gian. Nếu hành vi của người dùng thay đổi (do căng thẳng, bệnh tật hoặc môi trường khác nhau), khả năng xác thực có thể bị suy giảm.

Trải nghiệm người dùng:

- Cung cấp một trải nghiệm người dùng mượt mà, vì xác thực hành vi diễn ra tự động trong nền mà không cần người dùng phải thực hiện thêm bước nào. Tuy nhiên, có thể gặp vấn đề về độ chính xác, đặc biệt trong điều kiện ánh sáng kém hoặc với những người có sự khác biệt về thể chất.

Thực hành:

- Tích Hợp Sinh Trắc Học Hành Vi: Triển khai sinh trắc học hành vi để phân tích các mẫu hành vi của người dùng, chẳng hạn như tốc độ gõ phím và chuyển động chuột, nhằm xác minh danh tính liên tục trong suốt phiên làm việc. Điều này cung cấp một lớp bảo mật bổ sung mà không làm gián đoạn trải nghiệm người dùng.
- Giám Sát Hành Vi Người Dùng Trong Thời Gian Thực: Liên tục theo dõi và phân tích cách người dùng tương tác với các ứng dụng trong suốt phiên làm việc. Điều này giúp phát hiện các mẫu hành vi bất thường có thể chỉ ra gian lận tiềm ẩn, ngay cả khi một người không được ủy quyền vượt qua yêu cầu đăng nhập ban đầu.
- Xác Thực Dựa Trên Rủi Ro: Sử dụng xác thực dựa trên rủi ro (RBA) điều chỉnh mức độ xác thực cần thiết dựa trên các yếu tố rủi ro đã được đánh giá như hành vi người dùng, vị trí và thiết bị. Điều này cho phép điều chỉnh linh hoạt các biện pháp bảo mật dựa trên phân tích thời gian thực.
- Cân Nhắc Quyền Riêng Tư Dữ Liệu: Đảm bảo rằng dữ liệu hành vi được thu thập và lưu trữ một cách an toàn trong khi vẫn bảo vệ quyền riêng tư của người dùng. Thông báo cho người dùng về cách dữ liệu của họ sẽ được sử dụng và thực hiện các biện pháp bảo vệ thông tin cá nhân của họ.
- Đánh Giá An Ninh Định Kỳ: Thực hiện các đánh giá định kỳ về hệ thống xác thực hành vi của bạn để xác định các lỗ hổng và đảm bảo tính hiệu quả trước các mối đe dọa đang phát triển.

4.6 Token (Mã token)

Định nghĩa: Token-based authentication là một cơ chế xác thực người dùng dựa trên việc tạo ra một token – một chuỗi ký tự thường được mã hóa, mang thông tin xác định người dùng. Token này được máy chủ tạo ra và lưu trữ ở client, cho phép người dùng truy cập tài nguyên mà không cần nhập lại thông tin đăng nhập.

Ví dụ:

- **JSON Web Token (JWT):** Là loại token phổ biến nhất, được sử dụng để xác thực người dùng trong các ứng dụng web hiện đại

Mức độ bảo mật:

- Token-based authentication thường được coi là có tính bảo mật cao vì: Token thường có thời gian sống ngắn, giảm thiểu rủi ro nếu bị đánh cắp cũng như không cần lưu trữ thông tin nhạy cảm trên máy chủ, giúp giảm thiểu nguy cơ bị tấn công. Token có thể được mã hóa và ký để đảm bảo tính toàn vẹn và xác thực.

Trải nghiệm người dùng:

- Cung cấp trải nghiệm người dùng tốt hơn vì: Người dùng không cần phải nhập lại mật khẩu cho mỗi yêu cầu và Token có thể được lưu trữ trong local storage hoặc cookie, giúp truy cập dễ dàng và nhanh chóng.

Thực hành:

- Sử Dụng HTTPS: Đảm bảo rằng tất cả các yêu cầu chứa token đều được truyền qua HTTPS để bảo vệ dữ liệu khỏi bị nghe lén.
- Thời Gian Hết Hạn của Token: Đặt thời gian hết hạn cho token để giảm thiểu rủi ro từ việc sử dụng token đã bị đánh cắp.
- Refresh Tokens: Sử dụng refresh tokens để duy trì phiên làm việc mà không cần yêu cầu người dùng đăng nhập lại thường xuyên.
- Kiểm Tra Token: Thực hiện kiểm tra token trên máy chủ để đảm bảo tính hợp lệ trước khi cấp quyền truy cập vào tài nguyên.
- Xóa Token Khi Đăng Xuất: Đảm bảo rằng token được xóa khỏi client khi người dùng đăng xuất để ngăn chặn truy cập trái phép.

4.7 Xác Thực Không Mật Khẩu (Passwordless Authentication)

Định nghĩa: Là phương pháp xác minh danh tính người dùng mà không yêu cầu nhập mật khẩu. Thay vào đó, người dùng xác thực thông qua các yếu tố khác như sinh trắc học, mã OTP (One-Time Password) gửi qua SMS hoặc email, hoặc thiết bị mà họ sở hữu.

Ví dụ:

- **Mã OTP (One-Time Password):** Gửi qua SMS, email hoặc ứng dụng xác thực, người dùng chỉ cần nhập mã nhận được để đăng nhập.
- **Liên kết đăng nhập qua email:** Người dùng nhận một liên kết đăng nhập duy nhất qua email mà không cần phải nhập mật khẩu.
- **Xác thực sinh trắc học:** Sử dụng các yếu tố như vân tay hoặc nhận diện khuôn mặt.

Mức độ bảo mật:

- Xác thực không mật khẩu thường cung cấp một mức độ bảo mật cao vì nó loại bỏ rủi ro liên quan đến mật khẩu yếu, lặp lại, hoặc bị đánh cắp. Các phương pháp như OTP hoặc xác thực sinh trắc học là các yếu tố độc lập mà kẻ tấn công khó có thể giả mạo.

Trải nghiệm người dùng:

- Trải nghiệm người dùng rất mượt mà và không cần nhớ mật khẩu, giúp giảm sự phức tạp trong việc đăng nhập. Người dùng chỉ cần nhập mã OTP hoặc sử dụng phương thức sinh trắc học như vân tay hoặc nhận diện khuôn mặt.

Thực hành:

- Sử Dụng Các Phương Pháp Xác Thực Đáng Tin Cậy: Kết hợp nhiều phương pháp xác thực như sinh trắc học và mã OTP để tăng cường bảo mật.
- Giáo Dục Người Dùng: Cung cấp thông tin rõ ràng về cách thức hoạt động của xác thực không cần mật khẩu và lợi ích của nó để nâng cao sự chấp nhận của người dùng.
- Đảm Bảo An Ninh Thông Tin Gửi Đi: Sử dụng giao thức an toàn như HTTPS để bảo vệ thông tin xác thực trong quá trình truyền tải.
- Theo Dõi và Phân Tích Hành Vi Người Dùng: Giám sát hoạt động đăng nhập để phát hiện các hành vi bất thường có thể chỉ ra sự cố bảo mật.
- Cập Nhật Thường Xuyên: Đảm bảo rằng các phương pháp xác thực không cần mật khẩu luôn được cập nhật với công nghệ mới nhất và các biện pháp bảo vệ an ninh.

4.8 Xác Thực Bên Thứ Ba (Third-Party Authentication)

Định nghĩa: Xác thực bên thứ ba là phương pháp cho phép người dùng xác minh danh tính của mình thông qua một dịch vụ bên ngoài, thường là các nền tảng lớn như Google, Facebook hoặc Twitter. Điều này giúp người dùng truy cập vào ứng dụng mà không cần tạo tài khoản mới hoặc nhớ mật khẩu.

Ví dụ:

- **Google Sign-In:** Người dùng có thể đăng nhập vào ứng dụng hoặc trang web thông qua tài khoản Google của mình.
- **Facebook Login:** Sử dụng tài khoản Facebook để đăng nhập vào các ứng dụng hoặc dịch vụ khác.
- **OAuth:** Một giao thức cho phép người dùng xác thực bằng tài khoản của dịch vụ khác mà không cần chia sẻ mật khẩu, thường sử dụng trong các ứng dụng web.

Mức độ bảo mật:

- Xác thực qua bên thứ ba có thể cung cấp bảo mật cao nhờ vào các biện pháp bảo mật mạnh mẽ mà các nhà cung cấp dịch vụ lớn (như Google, Facebook) áp dụng. Các công ty này thường sử dụng các biện pháp bảo vệ như xác thực hai yếu tố (2FA) để bảo vệ tài khoản của người dùng.

Trải nghiệm người dùng:

- Trải nghiệm người dùng cực kỳ tiện lợi vì người dùng không cần phải nhớ mật khẩu cho mỗi dịch vụ hoặc ứng dụng. Việc đăng nhập nhanh chóng và dễ dàng thông qua tài khoản của các dịch vụ như Google hoặc Facebook giúp tiết kiệm thời gian đáng kể.

Thực hành:

- Sử Dụng OAuth: Triển khai giao thức OAuth để quản lý quyền truy cập và đảm bảo an toàn cho thông tin người dùng.
- Cung Cấp Tùy Chọn Đăng Nhập Đa Dạng: Cho phép người dùng lựa chọn giữa nhiều dịch vụ bên thứ ba để xác thực.
- Giáo Dục Người Dùng:Thông báo cho người dùng về cách thức hoạt động của xác thực bên thứ ba và lợi ích của nó.
- Theo Dõi và Phân Tích: Giám sát hoạt động đăng nhập để phát hiện các hành vi bất thường có thể chỉ ra sự cố bảo mật.
- Bảo Vệ Thông Tin Cá Nhân: Đảm bảo rằng thông tin cá nhân của người dùng được bảo vệ và chỉ được sử dụng cho mục đích xác thực.

5 Remote User-Authentication

5.1 Nguyên tắc xác thực từ xa

Xác thực từ xa là quá trình xác minh danh tính người dùng khi họ truy cập vào hệ thống qua mạng. Mục tiêu chính là đảm bảo rằng thông tin đăng nhập của người dùng được kiểm tra đúng cách và ngăn chặn các hành vi truy cập trái phép.

Các nguyên tắc cơ bản:

- **Xác minh danh tính:** Đảm bảo rằng chỉ những người dùng hợp lệ mới được truy cập. .
- **Bảo vệ dữ liệu:** Thông tin xác thực (mật khẩu, mã OTP) phải được mã hóa để tránh bị nghe lén.
- **Tính tin cậy:** Hệ thống xác thực phải hoạt động chính xác và ổn định.
- **Ngăn ngừa tấn công:** Bảo vệ khỏi các loại tấn công như phishing, brute force, và man-in-the-middle.

5.1.1 NIST Model for Electronic User Authentication.

Giới thiệu về NIST:

- **NIST (National Institute of Standards and Technology)** là Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ, trực thuộc Bộ Thương mại Hoa Kỳ. NIST đóng vai trò quan trọng trong việc thiết lập các tiêu chuẩn và hướng dẫn liên quan đến bảo mật thông tin, đặc biệt trong lĩnh vực xác thực người dùng và bảo mật hệ thống.

Vai trò trong bảo mật thông tin:

- Đưa ra các tiêu chuẩn và hướng dẫn giúp tổ chức xây dựng hệ thống bảo mật hiệu quả.
- Cung cấp tài liệu và khuyến nghị để giảm thiểu rủi ro từ các cuộc tấn công mạng.
- Tiêu chuẩn phổ biến: NIST SP 800-63 (Digital Identity Guidelines).

Mô hình xác thực của NIST tập trung vào việc đảm bảo **tính toàn vẹn, bảo mật và an toàn** trong việc xác thực người dùng. NIST phân loại các phương pháp xác thực thành ba cấp độ gọi là **Authentication Assurance Levels (AALs):**

- *AAL1 - Mức thấp (Low Assurance)*
  - ***Đặc điểm:***
    - Chỉ yêu cầu một yếu tố xác thực (e.g., mật khẩu).
    - Không yêu cầu mã hóa dữ liệu trong quá trình truyền.
  - ***Ưu điểm:***
    - Dễ triển khai, chi phí thấp.
  - ***Nhược điểm:***
    - Dễ bị tấn công (e.g., phishing, brute force).

  - ***Ứng dụng:***
    * Truy cập vào các hệ thống không nhạy cảm như diễn đàn hoặc trang thông tin công khai.

- *AAL2 - Mức trung bình (Moderate Assurance)*
  - ***Đặc điểm:***
    * Yêu cầu ít nhất hai yếu tố xác thực (e.g., mật khẩu kết hợp mã OTP).
    * Thông tin phải được mã hóa trong quá trình truyền tải.
  - ***Ưu điểm:***
    * Bảo mật tốt hơn, chống được các tấn công đơn giản.
  - ***Nhược điểm:***
    * Phức tạp hơn cho người dùng.
  - ***Ứng dụng:***
    * Tài khoản ngân hàng trực tuyến, hệ thống giao dịch tài chính.

- *AAL3 - Mức cao (High Assurance)*
  - ***Đặc điểm:***
    * Sử dụng xác thực mạnh mẽ như sinh trắc học hoặc thiết bị bảo mật (e.g., YubiKey).
    * Dữ liệu được mã hóa đầu cuối.
    * Đảm bảo các thiết bị xác thực không bị giả mạo.
  - ***Ưu điểm:***
    * Độ tin cậy cao, chống được các cuộc tấn công tinh vi.
  - ***Nhược điểm:***
    * Chi phí triển khai cao.
    * Yêu cầu thiết bị chuyên dụng.
  - ***Ứng dụng:***
    * Hệ thống quốc phòng, cơ quan chính phủ, hoặc các hệ thống yêu cầu bảo mật cao.

5.1.2 Các yếu tố xác thực: mật khẩu, token, sinh trắc học.

NIST xác định ba loại yếu tố xác thực chính: **Something You Know** (password, PIN,...), **Something You Have** (Mã OTP, thẻ thông minh, thiết bị phần cứng,...), **Something You Are** (Sinh trắc học)

5.2 Các phương pháp áp dụng trong Remote User-Authentication

5.2.1 Xác thực hai chiều (Mutual Authentication).

- Phương pháp này đảm bảo cả người dùng và máy chủ đều xác thực danh tính lẫn nhau.
- **Quy trình:**
  - Người dùng gửi thông tin xác thực tới máy chủ.
  - Máy chủ xác thực thông tin của người dùng.
  - Máy chủ gửi lại thông tin xác thực của mình để người dùng xác minh.
  - Nếu cả hai bên đều xác minh thành công, truy cập sẽ được cấp.
- **Ưu điểm:**
  - Cung cấp độ bảo mật cao hơn, ngăn chặn các tấn công giả mạo máy chủ.
  - Tăng tính tin cậy trong môi trường giao dịch từ xa.
- **Nhược điểm:**
  - Phức tạp hơn trong triển khai.
  - Yêu cầu thêm tài nguyên tính toán và giao thức bảo mật (e.g., SSL/TLS).
- **Ứng dụng:**
  - Giao dịch ngân hàng trực tuyến, dịch vụ thanh toán điện tử, VPN.

5.2.2 Xác thực một chiều (One-Way Authentication).

- Đây là phương pháp phổ biến, trong đó máy chủ xác thực danh tính của người dùng, nhưng người dùng không xác thực ngược lại máy chủ.
- **Quy trình:**
  - Người dùng gửi thông tin xác thực (e.g., tên đăng nhập, mật khẩu).
  - Máy chủ kiểm tra thông tin với dữ liệu lưu trữ.
  - Nếu thông tin hợp lệ, truy cập sẽ được cấp.
- **Ưu điểm:**
  - Đơn giản và dễ triển khai.
  - Phù hợp cho các hệ thống có mức độ bảo mật không quá cao.
- **Nhược điểm:**
  - Không đảm bảo rằng máy chủ là hợp lệ, dễ bị tấn công man-in-the-middle (MITM).
  - Bảo mật phụ thuộc vào độ mạnh của thông tin xác thực (e.g., mật khẩu).
- **Ứng dụng:**
  - Các dịch vụ web cơ bản như đăng nhập vào tài khoản mạng xã hội, diễn đàn.

5.3 Sử dụng mã hóa đối xứng trong xác thực từ xa

Mã hóa đối xứng sử dụng cùng một khóa bí mật (secret key) để mã hóa và giải mã dữ liệu. Phương pháp này đảm bảo tính bảo mật và tính toàn vẹn của thông tin khi được truyền qua mạng. Dưới đây là chi tiết từng bước cho hai phương pháp xác thực trong bối cảnh sử dụng mã hóa đối xứng.

5.3.1 Xác thực hai chiều bằng mã hóa đối xứng.

- **Mô hình hoạt động:** Cả client (người yêu cầu xác thực) và server (bên cung cấp dịch vụ) đều cần xác minh danh tính của nhau trước khi tiến hành giao tiếp.
- **Quy trình chi tiết:**
  - *Client khởi tạo yêu cầu xác thực:* Client gửi yêu cầu xác thực đến server, kèm theo thông tin định danh (ID của client).
  - *Server phản hồi với giá trị nonce (giá trị ngẫu nhiên):* Server tạo một giá trị ngẫu nhiên (nonce) để đảm bảo yêu cầu là duy nhất và không bị phát lại. Nonce được gửi đến client để xác minh khả năng mã hóa của client.
  - *Client mã hóa nonce của server:* Client sử dụng khóa đối xứng chung (pre-shared key) để mã hóa giá trị nonce và gửi lại cho server.
  - *Server giải mã và xác minh:* Server giải mã giá trị nhận được bằng khóa đối xứng. Nếu giá trị khớp với nonce ban đầu, client được xác minh là hợp lệ.
  - *Server gửi nonce thứ hai (Nonce_Server2):* Server gửi một giá trị nonce thứ hai hoặc một thông điệp được mã hóa đến client để client xác minh server.
  - *Client giải mã và xác minh:* Client giải mã thông điệp từ server bằng khóa đối xứng. Nếu giá trị hợp lệ, server được xác nhận danh tính.
- **Ưu điểm:**
  - Bảo mật cao, giảm nguy cơ giả mạo từ bên thứ ba.
  - Ngăn chặn các cuộc tấn công replay (phát lại yêu cầu).
- **Nhược điểm:**
  - Yêu cầu quản lý khóa phức tạp, đặc biệt trong hệ thống lớn.
  - Nếu khóa bị lộ, toàn bộ hệ thống có thể bị xâm phạm.
- **Ứng dụng:**
  - Ngân hàng trực tuyến, giao dịch tài chính, hệ thống IoT bảo mật.

5.3.2 Xác thực một chiều bằng mã hóa đối xứng.

- **Mô hình hoạt động:** Chỉ một bên, thường là client, cần xác minh danh tính của server.
- **Quy trình chi tiết:**
  - *Client gửi yêu cầu đến server:* Client gửi thông tin định danh hoặc yêu cầu kết nối đến server.
  - *Server phản hồi với giá trị mã hóa:* Server tạo một giá trị ngẫu nhiên (nonce) hoặc một thông điệp xác thực, mã hóa nó bằng khóa đối xứng, và gửi cho client.
  - *Client giải mã thông điệp từ server:* Client sử dụng khóa đối xứng để giải mã giá trị nhận được. Nếu giải mã thành công và giá trị khớp, client xác minh server là hợp lệ.
- **Ưu điểm:**
  - Đơn giản hơn so với xác thực hai chiều.
  - Dễ dàng triển khai trong các hệ thống không yêu cầu xác thực client.
- **Nhược điểm:**
  - Chỉ đảm bảo server là hợp lệ, không xác minh client.
  - Không ngăn chặn được các cuộc tấn công giả danh client.
- **Ứng dụng:**
  - Dịch vụ API, ứng dụng web cơ bản, hệ thống thông báo, dịch vụ tải xuống

5.4 Sử dụng mã hóa bất đối xứng trong xác thực từ xa

5.4.1 Xác thực hai chiều bằng mã hóa bất đối xứng

- **Mục tiêu:**
  - Xác thực danh tính của một bên giao tiếp (thường là server) với bên còn lại (thường là client).
  - Bảo đảm rằng dữ liệu nhận được thực sự đến từ nguồn hợp lệ và không bị sửa đổi.
- **Quy trình chi tiết:**
  1. *Client gửi yêu cầu xác thực:* Client tạo một nonce (N1 - số ngẫu nhiên) và gửi đến server kèm theo thông tin nhận dạng của mình.
  2. *Server phản hồi với thông tin được mã hóa:* Server tạo một nonce mới (N2) và mã hóa cùng với N1 bằng khóa bí mật chung (K) mà cả client và server đều biết trước. Server gửi lại thông điệp đã mã hóa.
  3. *Client giải mã và xác thực server:* Client sử dụng khóa K để giải mã thông điệp từ server. Nếu N1 hợp lệ, client biết rằng server là hợp lệ. Sau đó, client mã hóa lại N2 bằng khóa K và gửi lại server.

4. *Server giải mã và xác thực client:* Server giải mã thông điệp từ client để kiểm tra N2. Nếu N2 chính xác, server biết rằng client là hợp lệ
5. *Thiết lập session key:* Nếu cả hai bước xác thực đều thành công, một session key (Ks) có thể được tạo ra để mã hóa dữ liệu trao đổi trong phiên làm việc.

5.4.2 Xác thực một chiều bằng mã hóa bất đối xứng.

- **Mục tiêu:**
  - Xác thực danh tính của một bên giao tiếp (thường là server) với bên còn lại (thường là client).
  - Bảo đảm rằng dữ liệu nhận được thực sự đến từ nguồn hợp lệ và không bị sửa đổi.
- **Quy trình chi tiết:**
  1. *Client gửi yêu cầu đến server:* Client gửi một yêu cầu xác thực hoặc truy cập dịch vụ đến server.
  2. *Server gửi chứng chỉ:* Server phản hồi bằng cách gửi **chứng chỉ số (digital certificate)**, bao gồm: **Khóa công khai (public key)** của server và Chữ ký số của **Certificate Authority (CA)** chứng thực cho khóa công khai đó.
  3. *Client xác thực server:* Client kiểm tra tính hợp lệ của chứng chỉ bằng cách:
     - Xác minh chữ ký số của CA bằng khóa công khai của CA.
     - Đảm bảo rằng chứng chỉ không bị hết hạn hoặc bị thu hồi.
     - Đối chiếu tên miền hoặc thông tin của server trong chứng chỉ.
  4. *Thiết lập khóa phiên (nếu cần):*
     - Client tạo một **khóa phiên (session key)** ngẫu nhiên và mã hóa bằng **khóa công khai** của server, sau đó gửi đến server.
     - Server giải mã khóa phiên bằng **khóa riêng (private key)** của mình.
     - Khóa phiên này được sử dụng để mã hóa dữ liệu trao đổi trong phiên làm việc.

5.5 Federated Identity Management

5.5.1 Identity Management (Quản lý danh tính).

Mục tiêu: **Identity Management (Quản lý danh tính)** là một hệ thống tập trung dùng để quản lý và điều phối danh tính của người dùng, giúp kiểm soát truy cập vào các tài nguyên mạng một cách hiệu quả và an toàn.

5.5.2 Identity Federation (Liên kết danh tính).

Mục tiêu: **Identity Federation** là một cơ chế cho phép chia sẻ danh tính giữa nhiều tổ chức hoặc hệ thống khác nhau (gọi là các **miền bảo mật**, security domains). Đây là một phần quan trọng của **Federated Identity Management (FIM)**.

5.6 Personal Identity Verification (PIV)

5.6.1 PIV System Model

PIV (Personal Identity Verification) System Model là một hệ thống được thiết kế để xác thực danh tính cá nhân trong các tổ chức, thường được sử dụng trong môi trường yêu cầu mức độ bảo mật cao, như cơ quan chính phủ.

Cách triển khai:

- **Đăng ký:** Cá nhân đăng ký danh tính của mình với tổ chức. Thông tin này được xác minh và lưu trữ trong hệ thống.
- **Cấp phát thẻ:** PIV card được cấp với các thông tin mã hóa và chứng chỉ số.
- **Xác thực:** Người dùng sử dụng PIV card để đăng nhập vào hệ thống, thường đi kèm với mã PIN hoặc sinh trắc học.
- **Quản lý quyền truy cập:** Dựa trên danh tính được xác thực, hệ thống sẽ cấp quyền truy cập tương ứng.

5.6.2 PIV Documentation

PIV Documentation bao gồm các thông tin và tài liệu liên quan đến việc quản lý, triển khai, và sử dụng hệ thống xác thực danh tính cá nhân (PIV) trong các tổ chức, đặc biệt là trong các môi trường yêu cầu bảo mật cao.

5.6.3 PIV Credentials and Keys

PIV Credentials:

- **Chứng chỉ số (Digital Certificates):** Được lưu trữ trên thẻ PIV và dùng để xác thực danh tính người dùng. Các chứng chỉ này thường được cấp bởi một **Certificate Authority (CA)** đáng tin cậy.
- **Thông tin sinh trắc học (Biometric Information):** Bao gồm vân tay, ảnh khuôn mặt, hoặc dữ liệu khác để xác thực người dùng ở mức độ bảo mật cao.
- **Mã PIN (Personal Identification Number):** Được yêu cầu để kích hoạt thẻ trước khi sử dụng các chức năng khác như ký số hoặc mã hóa.
- **Tích hợp nhiều yếu tố:** Thẻ PIV có thể lưu trữ cả thông tin chứng thực vật lý và logic để hỗ trợ các phương thức xác thực đa yếu tố (MFA).

PIV keys: Gồm các loại khóa:

- **Authentication Key (Khóa xác thực):** Sử dụng để xác thực người dùng.
- **Digital Signature Key (Khóa ký số):** Dùng để ký số các tài liệu và giao dịch.
- **Encryption Key (Khóa mã hóa):** Bảo vệ dữ liệu nhạy cảm khi truyền tải hoặc lưu trữ.
- **Key Management Key (Khóa quản lý):** Quản lý giao tiếp an toàn giữa thẻ và hệ thống.

5.6.4 Authentication (Quá trình xác thực).

PIV hỗ trợ nhiều yếu tố xác thực để tăng cường bảo mật, bao gồm:

1. **Xác thực dựa trên mã PIN (PIN Authentication):**
   - Người dùng phải nhập mã PIN để kích hoạt thẻ PIV.
   - Đây là phương pháp xác thực cơ bản, yêu cầu người dùng nhớ mã PIN cá nhân.
2. **Xác thực sinh trắc học (Biometric Authentication):**
   - Sử dụng thông tin sinh trắc học như vân tay hoặc nhận dạng khuôn mặt được lưu trữ trên thẻ PIV.
   - Yêu cầu người dùng cung cấp dữ liệu sinh trắc học để so sánh với dữ liệu đã được lưu trữ.
3. **Xác thực bằng khóa mật mã (Cryptographic Authentication):**
   - Sử dụng các khóa mật mã và chứng chỉ số lưu trên thẻ để thực hiện các tác vụ như ký số hoặc mã hóa.
   - Phổ biến nhất là sử dụng khóa Authentication Key để xác thực với hệ thống thông qua giao thức mã hóa bất đối xứng.
4. **Xác thực đa yếu tố (Multi-Factor Authentication - MFA):**
   - Yêu cầu cả mã PIN và thông tin sinh trắc học.
   - Hoặc mã PIN kết hợp với chứng chỉ số.

6 Các giao thức và chuẩn hỗ trợ

6.1 Giao thức

6.1.1 OAuth 2.0

- OAuth 2.0 là một giao thức ủy quyền tiêu chuẩn mở, cho phép các ứng dụng truy cập vào tài nguyên của người dùng mà không cần chia sẻ thông tin đăng nhập như tên người dùng và mật khẩu.
- Cách hoạt động:
  - **Yêu cầu ủy quyền:** Client yêu cầu quyền truy cập từ Resource Owner thông qua Authorization Server.
  - **Cấp phép:** Nếu Resource Owner đồng ý, Authorization Server sẽ cấp một mã ủy quyền (authorization code) cho Client.
  - **Nhận token:** Client sử dụng mã ủy quyền để yêu cầu Access Token từ Authorization Server.
  - **Truy cập tài nguyên:** Client sử dụng Access Token để truy cập tài nguyên trên Resource Server.
- Lợi ích:
  - **Bảo mật cao hơn:** Người dùng không cần chia sẻ mật khẩu với các ứng dụng bên thứ ba.
  - **Dễ dàng tích hợp:** Các ứng dụng có thể dễ dàng tích hợp với nhau thông qua API mà không cần quản lý thông tin đăng nhập phức tạp.
  - **Hỗ trợ đa dạng nền tảng:** OAuth 2.0 có thể được sử dụng trên nhiều loại ứng dụng khác nhau, từ web đến di động.
- Ứng dụng:
  - Được sử dụng trong các ứng dụng web, di động.
  - Hỗ trợ đa nền tảng, dễ dàng tích hợp với API như Google, Facebook, GitHub.

6.1.2 OpenID Connect (OIDC)

- OpenID Connect (OIDC) là một giao thức xác thực dựa trên OAuth 2.0, cho phép các ứng dụng xác minh danh tính người dùng và lấy thông tin hồ sơ cơ bản của họ. OIDC được thiết kế để đơn giản hóa quy trình xác thực, cung cấp khả năng đăng nhập một lần (Single Sign-On - SSO) và cho phép người dùng sử dụng một bộ thông tin đăng nhập để truy cập nhiều dịch vụ khác nhau.
- Cách hoạt động:
  - **Người dùng truy cập** vào ứng dụng (RP) và chọn tùy chọn đăng nhập.
  - **RP gửi yêu cầu** đến OP để xác thực người dùng.

  - **OP xác thực người dùng** và nếu thành công, sẽ gửi lại một ID Token (thường là JSON Web Token - JWT) cùng với một Access Token cho RP.
  - **RP sử dụng Access Token** để yêu cầu thông tin bổ sung từ UserInfo Endpoint, nơi chứa các thông tin về người dùng như tên, email, v.v.

- Lợi ích:
  - **Xác thực an toàn:** Người dùng không cần chia sẻ mật khẩu với các ứng dụng bên thứ ba.
  - **Đăng nhập một lần (SSO):** Cho phép người dùng đăng nhập vào nhiều dịch vụ mà không cần phải nhập lại thông tin đăng nhập.
  - **Thông tin phong phú:** Cung cấp thông tin chi tiết về người dùng thông qua ID Token và UserInfo Endpoint.

- Ứng dụng:
  - Phổ biến trong các ứng dụng cần xác minh danh tính người dùng.
  - Hỗ trợ bảo mật mạnh mẽ nhờ JSON Web Token (JWT).

6.1.3 Kerberos

- Kerberos là một giao thức xác thực mạng được phát triển bởi MIT, nhằm cung cấp một phương pháp bảo mật cho việc xác thực các yêu cầu dịch vụ giữa các máy chủ đáng tin cậy trên một mạng không an toàn. Kerberos đã trở thành nền tảng cho nhiều hệ thống bảo mật mạng hiện đại và được sử dụng rộng rãi trong các môi trường yêu cầu xác thực mạnh mẽ.

- Cách hoạt động:
  - **Yêu cầu xác thực:** Client gửi yêu cầu đến AS của KDC với thông tin người dùng (User Principal Name - UPN). Yêu cầu này không chứa mật khẩu mà chỉ được mã hóa bằng hash của mật khẩu người dùng.
  - **Cấp phát TGT:** Nếu thông tin hợp lệ, AS sẽ cấp phát một TGT và một session key, cả hai đều được mã hóa và gửi lại cho client.
  - **Yêu cầu ticket dịch vụ:** Client sử dụng TGT để yêu cầu ticket dịch vụ từ TGS. Yêu cầu này bao gồm TGT đã mã hóa và thông tin về dịch vụ mà client muốn truy cập.
  - **Truy cập dịch vụ:** TGS kiểm tra tính hợp lệ của TGT và nếu hợp lệ, cấp phát một service ticket cho client. Client sau đó gửi ticket này đến SS để được truy cập.

- Lợi ích:
  - **Bảo mật cao:** Kerberos sử dụng mã hóa khóa đối xứng và không bao giờ truyền mật khẩu qua mạng, giúp bảo vệ thông tin đăng nhập khỏi bị đánh cắp.
  - **Xác thực lẫn nhau:** Không chỉ client mà cả server cũng phải xác thực danh tính của nhau, giảm thiểu nguy cơ tấn công man-in-the-middle.
  - **Hỗ trợ Single Sign-On (SSO):** Người dùng chỉ cần đăng nhập một lần để có thể truy cập nhiều dịch vụ khác nhau mà không cần phải nhập lại mật khẩu.

- Ứng dụng:
  - Được sử dụng trong hệ thống nội bộ như Active Directory.
  - Hỗ trợ xác thực mạnh mẽ với mã hóa symmetric (DES, AES).

6.1.4 RADIUS (Remote Authentication Dial-In User Service)

- RADIUS là một giao thức xác thực, ủy quyền và kế toán (AAA) được thiết kế để cung cấp dịch vụ xác thực cho người dùng truy cập vào mạng. Giao thức này sử dụng mô hình client-server, cho phép quản lý tập trung các thông tin xác thực và quyền truy cập của người dùng.
- Cách hoạt động:
  - **Yêu cầu truy cập:** Người dùng gửi thông tin đăng nhập (username và password) đến RADIUS Client.
  - **Gửi yêu cầu đến RADIUS Server:** RADIUS Client gửi một gói tin Access-Request đến RADIUS Server chứa thông tin đăng nhập.
  - Xác thực: RADIUS Server kiểm tra thông tin đăng nhập bằng cách so sánh với cơ sở dữ liệu người dùng. Nếu thông tin hợp lệ, server sẽ gửi lại gói tin Access-Accept; nếu không, nó sẽ gửi gói tin Access-Reject.
  - **Kế toán:** Sau khi xác thực thành công, NAS sẽ ghi lại thông tin về phiên làm việc của người dùng và gửi thông tin kế toán đến RADIUS Server.
- Lợi ích:
  - **Quản lý tập trung:** RADIUS cho phép quản lý thông tin đăng nhập và quyền truy cập từ một vị trí duy nhất, giúp giảm thiểu rủi ro bảo mật.
  - **Bảo mật cao:** Thông tin đăng nhập được mã hóa khi truyền tải qua mạng, bảo vệ khỏi các cuộc tấn công đánh cắp thông tin.
  - **Hỗ trợ đa dạng dịch vụ:** RADIUS có thể được sử dụng cho nhiều loại dịch vụ khác nhau như VPN, Wi-Fi, và truy cập từ xa.
- Ứng dụng:
  - Thích hợp cho mạng doanh nghiệp lớn.
  - Hỗ trợ nhiều phương pháp xác thực như username-password, OTP.

6.1.5 LDAP (Lightweight Directory Access Protocol)

- LDAP, viết tắt của Lightweight Directory Access Protocol, là một giao thức mạng được sử dụng để truy cập và quản lý thông tin trong một thư mục. Giao thức này được phát triển dựa trên tiêu chuẩn X.500 và được thiết kế để hoạt động hiệu quả trên nền tảng TCP/IP, mang lại khả năng truy cập nhanh chóng và nhẹ nhàng hơn so với các giao thức trước đó như DAP (Directory Access Protocol).
- Cách hoạt động:

  - **Gửi yêu cầu:** Client tạo ra một thông điệp LDAP chứa yêu cầu (như tìm kiếm, thêm mới, sửa đổi hoặc xóa) và gửi đến LDAP Server.
  - **Xử lý yêu cầu:** LDAP Server nhận yêu cầu, xử lý thông tin và truy xuất dữ liệu từ cơ sở dữ liệu.
  - **Gửi phản hồi:** Server gửi lại kết quả cho client dưới dạng thông điệp LDAP.

- Lợi ích:
  - **Quản lý tập trung:** LDAP cho phép lưu trữ thông tin người dùng, thiết bị và tài nguyên trong một vị trí trung tâm, giúp dễ dàng quản lý và truy cập.
  - **Bảo mật:** Giao thức hỗ trợ nhiều phương thức xác thực và ủy quyền, giúp bảo vệ thông tin nhạy cảm.
  - **Hỗ trợ cho Single Sign-On (SSO):** LDAP có thể tích hợp với các hệ thống SSO, cho phép người dùng truy cập nhiều ứng dụng chỉ với một lần đăng nhập .

- Ứng dụng:
  - Được sử dụng trong quản lý danh tính và truy cập (IAM).
  - Hỗ trợ tích hợp với các hệ thống xác thực khác như Kerberos hoặc Active Directory.

6.2 Chuẩn hỗ trợ

6.2.1 SAML (Security Assertion Markup Language)

- SAML, viết tắt của Security Assertion Markup Language, là một tiêu chuẩn mở dựa trên XML được thiết kế để trao đổi dữ liệu xác thực và ủy quyền giữa các bên, chủ yếu giữa nhà cung cấp danh tính (Identity Provider - IdP) và nhà cung cấp dịch vụ (Service Provider - SP). SAML giúp đơn giản hóa quy trình xác thực người dùng, cho phép họ truy cập nhiều ứng dụng với một bộ thông tin đăng nhập duy nhất.

- Cách hoạt động:

  - **Yêu cầu truy cập:** Khi người dùng cố gắng truy cập vào một dịch vụ, SP sẽ tạo ra một yêu cầu xác thực SAML và chuyển hướng người dùng đến IdP.
  - **Xác thực:** IdP xác thực thông tin đăng nhập của người dùng. Nếu thành công, IdP sẽ tạo ra một tài liệu SAML chứa thông tin xác thực và gửi lại cho SP.
  - **Cấp quyền truy cập:** SP nhận tài liệu SAML, kiểm tra tính hợp lệ của nó và cấp quyền truy cập cho người dùng nếu thông tin xác thực là hợp lệ.

- Lợi ích:

  - **Đăng nhập Một lần (SSO):** Người dùng chỉ cần đăng nhập một lần để truy cập nhiều ứng dụng khác nhau, giảm thiểu sự phiền phức trong việc nhớ nhiều mật khẩu.
  - **Bảo mật cao:** SAML sử dụng mã hóa để bảo vệ thông tin nhạy cảm trong quá trình truyền tải.
  - **Quản lý tập trung:** Giúp tổ chức quản lý danh tính người dùng và quyền truy cập từ một vị trí trung tâm.

- Hỗ trợ:

  - SAML Assertion chứa các thông tin như danh tính người dùng, quyền hạn và xác nhận danh tính.
  - Tương thích với các hệ thống doanh nghiệp, giáo dục, và chính phủ.

- Ứng dụng:

  - Đăng nhập một lần trên các nền tảng như Google Workspace, Microsoft 365.

6.2.2 FIDO (Fast Identity Online)

- FIDO, viết tắt của Fast Identity Online, là một tập hợp các tiêu chuẩn xác thực mở được phát triển bởi FIDO Alliance. Mục tiêu chính của FIDO là loại bỏ sự phụ thuộc vào mật khẩu bằng cách cung cấp các phương pháp xác thực an toàn và tiện lợi hơn. Giao thức này sử dụng công nghệ mã hóa khóa công khai để bảo vệ thông tin xác thực và giảm thiểu rủi ro bị đánh cắp thông tin.

- Cách hoạt động:

  - **Đăng ký:**

    * Người dùng đăng ký thiết bị của mình với dịch vụ trực tuyến bằng cách cung cấp thông tin như tên người dùng và mật khẩu (chỉ trong giai đoạn này).
    * Một cặp khóa (khóa riêng và khóa công khai) được tạo ra. Khóa riêng được lưu trữ trên thiết bị của người dùng, trong khi khóa công khai được gửi đến dịch vụ trực tuyến.
  - **Xác thực:**
    * Khi người dùng muốn đăng nhập, họ sẽ sử dụng phương pháp xác thực đã chọn (ví dụ: quét vân tay, nhận diện khuôn mặt hoặc nhập PIN).
    * Thiết bị sẽ chứng minh quyền sở hữu khóa riêng bằng cách ký một thách thức từ dịch vụ trực tuyến mà không cần gửi mật khẩu qua mạng.

- Lợi ích:
  - **Bảo mật cao:** Thông tin nhạy cảm như khóa riêng và dữ liệu sinh trắc học không bao giờ rời khỏi thiết bị của người dùng, làm giảm khả năng bị đánh cắp.
  - **Trải nghiệm người dùng tốt hơn:** Người dùng có thể đăng nhập nhanh chóng mà không cần nhớ mật khẩu, chỉ cần sử dụng phương pháp xác thực đã chọn.
  - **Hỗ trợ đa dạng:** FIDO cho phép sử dụng nhiều phương pháp xác thực khác nhau, từ sinh trắc học đến thiết bị vật lý như USB token.

- Hỗ trợ:
  - FIDO2 hỗ trợ đăng nhập không cần mật khẩu.
  - Hỗ trợ các thiết bị sinh trắc học như vân tay hoặc nhận diện khuôn mặt.

- Ứng dụng:
  - Xác thực trong ngân hàng, ứng dụng di động, và các nền tảng web không cần mật khẩu.

6.2.3 X.509 (Digital Certificates)

- X.509 là một tiêu chuẩn quốc tế được phát triển bởi Liên minh Viễn thông Quốc tế (ITU), định nghĩa định dạng của các chứng chỉ khóa công khai. Chứng chỉ X.509 được sử dụng rộng rãi trong nhiều giao thức Internet, bao gồm TLS/SSL, là nền tảng cho HTTPS, giúp bảo mật việc duyệt web.

- Cách hoạt động:
  - **Yêu cầu cấp chứng chỉ:** Một tổ chức hoặc cá nhân tạo một cặp khóa (khóa riêng và khóa công khai) và gửi yêu cầu cấp chứng chỉ (Certificate Signing Request - CSR) tới một cơ quan cấp chứng chỉ (CA).
  - **Xác thực và cấp phát:** CA xác thực thông tin trong CSR và cấp phát một chứng chỉ X.509, liên kết khóa công khai với danh tính đã xác thực.
  - **Sử dụng chứng chỉ:** Chứng chỉ này có thể được sử dụng để thiết lập kết nối an toàn, xác thực danh tính và ký tài liệu số.

- Lợi ích:

  - **Bảo mật cao:** Chứng chỉ X.509 giúp xác thực danh tính và mã hóa dữ liệu, bảo vệ thông tin nhạy cảm khỏi bị đánh cắp.
  - **Xác thực lẫn nhau:** Cung cấp khả năng xác thực giữa các bên tham gia giao dịch, đảm bảo rằng cả hai bên đều là những người mà họ tuyên bố.
  - **Quản lý tập trung:** Hệ thống PKI dựa trên X.509 cho phép quản lý tập trung các chứng chỉ, giúp dễ dàng kiểm soát và theo dõi. .

- Hỗ trợ:
  - Được dùng để bảo mật giao tiếp trong HTTPS.
  - Dùng trong chứng thực danh tính qua các ứng dụng như email, VPN, hoặc ký số tài liệu.

- Ứng dụng:
  - Cơ sở hạ tầng mã hóa SSL/TLS, các giao dịch ngân hàng trực tuyến.

6.2.4 NIST SP 800-63 (Digital Identity Guidelines)

- NIST SP 800-63, được phát hành bởi Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST), là bộ hướng dẫn về quản lý danh tính số, cung cấp các yêu cầu cho các cơ quan liên bang trong việc triển khai dịch vụ danh tính số.

- Cách hoạt động:
  - **Định nghĩa ba cấp độ xác thực** (IAL, AAL, FAL) dựa trên độ tin cậy của danh tính, phương thức xác thực, và liên kết phiên làm việc.
  - **Mức Độ Đảm Bảo Danh Tính (IAL):**
    * IAL1: Không yêu cầu liên kết người đăng ký với danh tính thực tế cụ thể.
    * IAL2: Cần có bằng chứng hỗ trợ sự tồn tại của danh tính đã được tuyên bố.
    * IAL3: Yêu cầu sự hiện diện vật lý để xác minh danh tính.
  - **Mức Độ Đảm Bảo Xác Thực (AAL):**
    * AAL1: Cung cấp yêu cầu tối thiểu cho xác thực.
    * AAL2: Yêu cầu sử dụng một phương thức xác thực mạnh hơn.
    * AAL3: Yêu cầu các phương thức xác thực mạnh nhất, thường bao gồm nhiều yếu tố.

- Lợi ích:
  - **Bảo mật cao hơn:** Các hướng dẫn này giúp các cơ quan xây dựng hệ thống bảo mật mạnh mẽ hơn thông qua việc xác minh danh tính và kiểm soát truy cập.
  - **Khả năng tương tác:** Hỗ trợ việc triển khai các giải pháp danh tính linh hoạt và có thể tích hợp với nhiều hệ thống khác nhau.
  - **Tăng cường quyền riêng tư:** Khuyến khích việc giảm thiểu thông tin cá nhân được thu thập và chia sẻ trong quá trình xác thực.

- Hỗ trợ:

  - Xác định các phương pháp bảo mật cần thiết cho từng cấp độ bảo mật.
  - Áp dụng mật khẩu, sinh trắc học, hoặc mã hóa token.

- Ứng dụng:
  - Cung cấp tiêu chuẩn xác thực cho các cơ quan chính phủ và doanh nghiệp.

7 Các vấn đề về bảo mật

Những lỗ hổng sau nhấn mạnh tầm quan trọng của việc triển khai và giám sát các phương pháp xác thực an toàn, đồng thời nâng cao nhận thức về bảo mật cho người dùng:

7.1 Chống tấn công brute-force không hiệu quả

- Tấn công brute-force, chẳng hạn như dictionary attack, là việc cố gắng truy cập trái phép vào hệ thống hoặc tài khoản của người dùng bằng cách thử hàng loạt tổ hợp tên đăng nhập và mật khẩu, cho đến khi tìm ra tổ hợp đúng.
- Nếu hệ thống bảo vệ brute-force bị lỗi, như lỗ hổng trong logic xác thực, tường lửa hoặc giao thức SSH, kẻ tấn công có thể chiếm quyền đăng nhập và gây ra rủi ro nghiêm trọng cho thông tin người dùng.

7.2 Thông tin đăng nhập yếu

- Khi người dùng tạo tài khoản, họ thường phải chọn tên đăng nhập và mật khẩu. Tuy nhiên, nếu mật khẩu dễ đoán, điều này làm tăng nguy cơ bị tấn công.
- Kẻ xâm nhập thường không cần sử dụng toàn bộ brute-force mà chỉ thử các mật khẩu phổ biến như **"admin," "password1"** để khai thác. Nếu không có chính sách hạn chế mật khẩu yếu, các trang web dù có bảo vệ brute-force cũng có thể bị xâm phạm.

7.3 Lộ thông tin tên đăng nhập (Username Enumeration)

Mặc dù không phải lỗ hổng trực tiếp, nhưng nó giúp kẻ tấn công dễ dàng thực hiện các cuộc tấn công khác như brute-force.

Ví dụ:

- Một thông báo lỗi cho biết "mật khẩu không chính xác với tên người dùng Sally" xác nhận rằng tên đăng nhập hợp lệ.
- Ngược lại, thông báo "tên đăng nhập không tồn tại" cho thấy tên đó không hợp lệ.

Kẻ tấn công có thể dựa vào các thông báo này để xác định tài khoản hợp lệ, sau đó sử dụng kỹ thuật brute-force để tấn công hiệu quả hơn.

7.4 HTTP Basic Authentication không an toàn

Giao thức xác thực HTTP Basic Authentication đơn giản nhưng dễ gặp rủi ro nếu không sử dụng mã hóa TLS cho toàn bộ kết nối.

- Thông tin đăng nhập (username và password) có thể bị gửi dưới dạng văn bản thuần, dễ bị đánh cắp.
- Trình duyệt hiện đại thường lưu trữ thông tin này vô thời hạn, tạo điều kiện cho kẻ xâm nhập sử dụng lại thông tin.

7.5 Quản lý phiên làm việc (Session Management) kém

- Các lỗ hổng như không giới hạn thời gian phiên, lộ ID phiên trong URL, hoặc cookie không có cờ Http-Only dễ dẫn đến việc chiếm đoạt phiên.
- Kẻ tấn công có thể giả mạo danh tính của người dùng đã xác thực và truy cập trái phép vào hệ thống.

7.6 Chức năng "Ghi nhớ đăng nhập" kém bảo mật

Tính năng "Remember Me" tạo cookie để bỏ qua việc đăng nhập. Nếu cookie này dễ đoán hoặc không được bảo vệ tốt, kẻ xâm nhập có thể sử dụng tấn công brute-force hoặc XSS để đánh cắp cookie và truy cập tài khoản.

7.7 Tấn công SQL Injection

SQL Injection sử dụng mã độc SQL để xâm nhập cơ sở dữ liệu, đánh cắp mật khẩu hoặc bỏ qua quy trình xác thực. Lỗ hổng này thường xuất hiện khi hệ thống không kiểm tra chặt chẽ đầu vào từ người dùng.

7.8 Quy trình đổi và khôi phục mật khẩu không an toàn

Nếu quy trình khôi phục mật khẩu dựa vào câu hỏi bảo mật dễ đoán, không có CAPTCHA, hoặc liên kết khôi phục không có thời hạn, kẻ xâm nhập có thể sử dụng brute-force để chiếm quyền truy cập.

7.9 Xác thực hai yếu tố (2FA) kém an toàn

- 2FA có thể trở nên kém hiệu quả nếu mã xác thực (verification code) dễ bị khai thác, như tấn công SIM Swap khi mã được gửi qua SMS.
- Ngoài ra, nếu không có bảo vệ brute-force, tài khoản vẫn dễ bị xâm nhập.

7.10 Lỗi trong logic xác thực

Các lỗ hổng logic xảy ra khi mã hoặc thiết kế của ứng dụng không đầy đủ. Ví dụ, ứng dụng có thể hỏi câu hỏi bảo mật dễ đoán như ngày sinh hoặc tên mẹ của người dùng, tạo cơ hội cho kẻ tấn công vượt qua xác thực.

8 Nâng cấp bảo mật

8.1 Nâng cấp chính sách mật khẩu mạnh (Strong Password Policies)

- Khuyến khích người dùng tạo mật khẩu an toàn bằng cách áp dụng các quy tắc nhằm ngăn chặn mật khẩu yếu hoặc dễ đoán, đồng thời yêu cầu họ thay đổi mật khẩu định kỳ.
- Việc hướng dẫn người dùng thay thế mật khẩu mặc định và sử dụng mật khẩu riêng cho từng tài khoản sẽ giúp giảm thiểu rủi ro bị lộ thông tin.

8.2 Kích hoạt xác thực đa yếu tố (Multi-Factor Authentication - MFA)

- Thêm một lớp bảo mật bên cạnh mật khẩu thông qua các yếu tố như quét vân tay, mã token hoặc mã SMS.
- MFA kết hợp giữa thông tin người dùng biết (như mật khẩu) với những gì họ có hoặc là (ví dụ thiết bị hoặc sinh trắc học), giúp bảo mật tốt hơn và giảm thiểu nguy cơ bị xâm nhập.

8.3 Áp dụng xác thực sinh trắc học (Biometric Authentication)

- Triển khai các công nghệ xác thực dựa trên sinh trắc học như quét vân tay hoặc nhận diện khuôn mặt.
- Với đặc tính duy nhất của mỗi cá nhân, sinh trắc học là phương thức đáng tin cậy để đảm bảo danh tính, khó bị giả mạo hoặc đánh cắp hơn mật khẩu thông thường.

8.4 Cập nhật và vá lỗi hệ thống xác thực thường xuyên

- Thực hiện cập nhật định kỳ cho hệ thống xác thực nhằm sửa chữa các lỗ hổng và bảo vệ trước các mối đe dọa mới phát sinh.
- Điều này đảm bảo rằng các phương pháp xác thực luôn an toàn và đáp ứng tốt các yêu cầu bảo mật hiện hành.

8.5 Đào tạo người dùng về an ninh

- Nâng cao nhận thức của người dùng về các rủi ro như giả mạo tài khoản hoặc các hình thức lừa đảo khác. Thông qua các chương trình đào tạo liên tục, người dùng sẽ hiểu rõ hơn về tầm quan trọng của việc bảo vệ thông tin đăng nhập và biết cách ứng phó với các cuộc tấn công xã hội.

8.6 Sử dụng xác thực thích ứng (Adaptive Authentication)

- Tích hợp các biện pháp xác thực dựa trên ngữ cảnh, như hành vi người dùng, thiết bị sử dụng hoặc tần suất truy cập. Phương pháp này giúp đánh giá mức độ rủi ro và áp dụng các biện pháp bảo mật phù hợp, đảm bảo hiệu quả bảo vệ.

8.7 Theo dõi và phân tích các sự kiện xác thực

- Triển khai các công cụ giám sát và ghi nhật ký để theo dõi các sự kiện xác thực theo thời gian thực, giúp phát hiện các hoạt động bất thường hoặc nghi ngờ.
- Phân tích nhật ký xác thực giúp nhận diện sớm các mối đe dọa và ngăn chặn kịp thời, từ đó tăng cường khả năng phòng vệ chủ động.

9 Ứng dụng thực tiễn

User authentication có rất nhiều ứng dụng thực tiễn trong các lĩnh vực khác nhau, từ công nghệ đến đời sống hàng ngày. Dưới đây là một số ví dụ điển hình:

9.1 Ứng dụng trong dịch vụ trực tuyến

- **Mạng xã hội (Facebook, Instagram, Twitter):** Xác minh danh tính người dùng để bảo vệ tài khoản cá nhân và nội dung riêng tư.
- **Email (Gmail, Outlook):** Đảm bảo rằng chỉ chủ sở hữu mới có thể truy cập vào hộp thư điện tử.
- **Dịch vụ giải trí (Netflix, Spotify):** Kiểm soát quyền truy cập vào nội dung được cấp phép.

9.2 Thương mại điện tử và thanh toán trực tuyến

- **Sàn thương mại điện tử (Amazon, Shopee):** Xác nhận danh tính để thực hiện giao dịch an toàn.
- **Ví điện tử (PayPal, Momo):** Bảo vệ thông tin tài khoản và giao dịch tài chính.
- **Xác thực hai yếu tố (2FA):** Tăng cường bảo mật khi mua sắm trực tuyến.

9.3 Ứng dụng trong doanh nghiệp

- **Hệ thống quản lý nhân sự:** Quản lý truy cập vào các tài liệu và dữ liệu nội bộ, đảm bảo rằng chỉ nhân viên có quyền mới được phép truy cập.
- **Làm việc từ xa (VPN, Microsoft Teams, Zoom):** Xác thực nhân viên khi truy cập hệ thống từ xa để bảo mật dữ liệu công ty.

9.4 Ứng dụng trong giáo dục

- **Hệ thống học trực tuyến (Coursera, Udemy):** Đảm bảo người học đăng nhập đúng tài khoản để truy cập vào các khóa học đã đăng ký.
- **Hệ thống quản lý trường học:** Xác thực giáo viên, học sinh và phụ huynh để truy cập thông tin cá nhân và điểm số.

9.5 Ứng dụng trong các thiết bị thông minh

- **Điện thoại di động và máy tính bảng:** Sử dụng mật khẩu, vân tay, hoặc nhận diện khuôn mặt để mở khóa.
- **Nhà thông minh:** Xác thực chủ sở hữu để kiểm soát các thiết bị (khóa cửa, camera an ninh, điều hòa nhiệt độ).

9.6 Ứng dụng trong chăm sóc sức khỏe

- **Hồ sơ y tế điện tử:** Xác thực bệnh nhân và bác sĩ để truy cập thông tin y tế nhạy cảm.
- **Ứng dụng y tế (Telemedicine):** Đảm bảo chỉ bệnh nhân và bác sĩ mới có thể tham gia vào cuộc tư vấn.

9.7 Hệ thống giao thông và dịch vụ công cộng

- **Dịch vụ chia sẻ xe (Grab, Uber):** Xác thực tài khoản tài xế và hành khách để đảm bảo an toàn.
- **Hệ thống vé điện tử:** Xác minh quyền sử dụng vé trong giao thông công cộng (tàu, xe buýt, máy bay).

9.8 Ứng dụng trong an ninh

- **Hệ thống camera giám sát:** Hạn chế quyền truy cập vào dữ liệu camera.
- **Bảo vệ cơ sở hạ tầng quan trọng:** Xác thực nhân viên để quản lý các thiết bị và hệ thống quan trọng như nhà máy điện, ngân hàng.

10 Kerberos Authentication: Kiến Trúc, Hoạt Động & Ứng Dụng

10.1 Giới Thiệu về Kerberos

10.1.1 Tổng quan về Kerberos

- **Lịch sử**: Vào đầu những năm 1980, MIT khởi xướng dự án Athena, một sáng kiến nhằm cung cấp một môi trường máy tính phân tán cho sinh viên và giảng viên. Dự án này yêu cầu một cơ chế xác thực mạnh mẽ để bảo vệ tài nguyên mạng. Với sự gia tăng của các mạng máy tính và các mối đe dọa bảo mật, MIT nhận ra rằng cần có một giao thức xác thực an toàn hơn so với các phương pháp truyền thống như truyền mật khẩu dạng văn bản thuần túy. Từ động lực trên hai nhà nghiên cứu tại MIT Kerberos được phát triển bởi Steve Miller và Clifford Neuman.

- **Bối cảnh hiện nay**: Giao thức Kerberos được ứng dụng rộng rãi trong nhiều dự án thực tế, đặc biệt là trong các môi trường doanh nghiệp và mạng nội bộ. Hệ điều hành Windows, Linux, macOS: Kerberos được tích hợp sẵn trong các hệ điều hành này để cung cấp xác thực và bảo mật cho người dùng khi truy cập vào các tài nguyên mạng. Ứng dụng máy chủ như Apache và MySQL: Các ứng dụng này sử dụng Kerberos để xác thực người dùng và bảo mật thông tin, giúp tăng cường bảo mật cho các dịch vụ web và cơ sở dữ liệu

10.1.2 Tại sao lại là Kerberos

- **Single Sign-On**: Single Sign-On (SSO) là một trong những ưu điểm nổi bật nhất của Kerberos, giúp người dùng chỉ cần đăng nhập một lần để truy cập nhiều dịch vụ và tài nguyên mạng mà không cần phải đăng nhập lại.

- **Không truyền mật khẩu qua mạng**: Trong các hệ thống xác thực truyền thống, mật khẩu thường được truyền qua mạng dưới dạng văn bản thuần túy (plaintext) hoặc được mã hóa yếu, dễ bị đánh cắp bởi các công cụ nghe lén. Kerberos không truyền mật khẩu qua mạng, thay vào đó sử dụng các ticket được mã hóa mạnh mẽ, giúp ngăn chặn các cuộc tấn công nghe lén.

10.2 Kiến Trúc & Thành Phần Của Kerberos

10.2.1 Nguyên lý hoạt động của Kerberos

- Trong Kerberos, ticket là trung tâm của cơ chế xác thực. Ticket đóng vai trò như một "giấy thông hành" được mã hóa, giúp người dùng và dịch vụ xác thực lẫn nhau mà không cần truyền mật khẩu qua mạng.
- Các ticket được mã hóa bởi một key chỉ được biết bởi các server với nhau và sử dụng các giao thức mã hóa đối xứng để mã hóa các ticket của nhau

10.2.2 Các thành phần chính

- Client
  - Client là máy tính hoặc thiết bị của người dùng, yêu cầu truy cập vào các dịch vụ hoặc tài nguyên mạng.
  - Vai trò:
    * Gửi yêu cầu xác thực đến Authentication Server (AS) để nhận Ticket-Granting Ticket (TGT)
    * Sử dụng TGT để yêu cầu Service Ticket (ST) từ Ticket-Granting Server (TGS)
    * Gửi ST đến Server để truy cập dịch vụ
- Server
  - Server là máy chủ cung cấp các dịch vụ hoặc tài nguyên mà client muốn truy cập
  - Vai trò:
    * Nhận Service Ticket (ST) từ client
    * Giải mã ST bằng khóa bí mật của mình để xác minh danh tính của client
    * Cấp quyền truy cập cho client nếu ST hợp lệ
- Authentication Server (AS)
  - Vai trò:
    * Xác thực người dùng khi họ đăng nhập vào hệ thống
    * Cấp Ticket-Granting Ticket (TGT) và session key cho client sau khi xác thực thành công
    * Mã hóa TGT bằng khóa bí mật của TGS (Ktgs) và session key bằng khóa bí mật của client (Kc).
  - Quy trình:
    * Client gửi yêu cầu xác thực đến AS
    * AS xác minh danh tính của client và tạo TGT cùng session key
    * AS gửi TGT và session key (được mã hóa) về cho client
- Ticket-Granting Server (TGS)

- Vai trò:
  - Cấp Service Ticket (ST) cho client để truy cập vào các dịch vụ cụ thể
  - Xác minh TGT và authenticator từ client để đảm bảo tính hợp lệ của yêu cầu
  - Mã hóa ST bằng khóa bí mật của dịch vụ (Kv) và session key bằng khóa bí mật của client (Kc,tgs)
- Quy trình:
  - Client gửi TGT và authenticator đến TGS
  - TGS giải mã TGT và authenticator để xác minh danh tính của client
  - TGS tạo ST và gửi về cho client

10.2.3 Các loại vé trong kerberos

- Ticket-Granting Ticket (TGT)
  - Định nghĩa
    - TGT là một loại ticket được cấp bởi Authentication Server (AS) sau khi người dùng đăng nhập thành công
    - TGT cho phép client yêu cầu các Service Ticket (ST) từ Ticket-Granting Server (TGS) mà không cần phải đăng nhập lại
  - Thông tin trong TGT
    - Tên người dùng (client principal): Xác định người dùng mà TGT được cấp.
    - Địa chỉ mạng của client: Đảm bảo rằng TGT chỉ có thể được sử dụng từ địa chỉ mạng cụ thể
    - Thời gian hiệu lực (timestamp và thời gian sống): Đảm bảo TGT chỉ có giá trị trong một khoảng thời gian nhất định
    - Session key (Kc,tgs): Khóa phiên dùng để giao tiếp giữa client và TGS
  - Cách TGT được sử dụng
    - Client đăng nhập:
      1. Client gửi yêu cầu xác thực đến AS
      2. AS xác minh danh tính của client và tạo TGT cùng session key (Kc,tgs)
      3. AS gửi TGT (được mã hóa bằng khóa bí mật của TGS - Ktgs) và session key (được mã hóa bằng khóa bí mật của client - Kc) về cho client
    - Client yêu cầu Service Ticket:
      1. Client gửi TGT và một authenticator (được mã hóa bằng session key - Kc,tgs) đến TGS
      2. TGS giải mã TGT bằng khóa bí mật của mình (Ktgs) và xác minh thông tin trong TGT và authenticator
      3. Nếu hợp lệ, TGS cấp Service Ticket (ST) cho client
  - Vai trò của TGT
    - Hỗ trợ Single Sign-On (SSO): TGT cho phép người dùng đăng nhập một lần và sử dụng nhiều dịch vụ mà không cần đăng nhập lại

    - Bảo mật: TGT được mã hóa bằng khóa bí mật của TGS, đảm bảo rằng chỉ TGS mới có thể giải mã và đọc thông tin

- Service Ticket (ST)
  - Định nghĩa
    - ST là một loại ticket được cấp bởi Ticket-Granting Server (TGS) để client truy cập vào một dịch vụ cụ thể
    - ST cho phép client xác thực với Server và truy cập vào tài nguyên mà server cung cấp
  - Thông tin trong ST
    - Tên người dùng (client principal): Xác định người dùng mà ST được cấp
    - Địa chỉ mạng của client: Đảm bảo rằng ST chỉ có thể được sử dụng từ địa chỉ mạng cụ thể
    - Thời gian hiệu lực (timestamp và thời gian sống): Đảm bảo ST chỉ có giá trị trong một khoảng thời gian nhất định
    - Session key (Kc,v): Khóa phiên dùng để giao tiếp giữa client và server
  - Cách ST được sử dụng
    - Client yêu cầu ST:
      1. Client gửi TGT và authenticator đến TGS để yêu cầu ST
      2. TGS xác minh TGT và authenticator, sau đó tạo ST cùng session key (Kc,v)
      3. TGS gửi ST (được mã hóa bằng khóa bí mật của server - Kv) và session key (được mã hóa bằng session key của client và TGS - Kc,tgs) về cho client
    - Client truy cập dịch vụ:
      1. Client gửi ST và một authenticator (được mã hóa bằng session key - Kc,v) đến server
      2. Server giải mã ST bằng khóa bí mật của mình (Kv) và xác minh thông tin trong ST và authenticator
      3. Nếu hợp lệ, server cấp quyền truy cập cho client
  - Vai trò của ST
    - Truy cập dịch vụ: ST cho phép client truy cập vào các dịch vụ cụ thể mà server cung cấp
    - Bảo mật: ST được mã hóa bằng khóa bí mật của server, đảm bảo rằng chỉ server mới có thể giải mã và đọc thông tin

10.3 Quy Trình Xác Thực Của Kerberos

10.3.1 Bước 1: Xác thực ban đầu (Client → Authentication Server - AS)

- **Quy trình:**
  - Người dùng nhập **tên đăng nhập** và **mật khẩu** vào **Client**.
  - Client tạo một yêu cầu xác thực và gửi đến **Authentication Server (AS)**.
  - **AS kiểm tra danh tính** của Client trong **cơ sở dữ liệu người dùng**.
  - Nếu hợp lệ, AS sẽ tạo một **Ticket Granting Ticket (TGT)**.
  - TGT này được mã hóa bằng **khóa bí mật của AS** và gửi về Client.
- **TGT gồm:**
  - ID người dùng.
  - Thời gian hết hạn vé.
  - Khóa phiên (Session Key) dùng để giao tiếp với **Ticket Granting Server (TGS)**.
- **Mục đích:**
  - Giúp Client xác thực một lần duy nhất mà không cần gửi lại mật khẩu trong mỗi lần truy cập dịch vụ.

10.3.2 Bước 2: Nhận vé dịch vụ (Client → Ticket Granting Server - TGS)

- **Quy trình:**
  - Khi cần truy cập một dịch vụ, Client sẽ gửi **TGT** đến **TGS**, kèm theo yêu cầu cấp vé dịch vụ.
  - TGS kiểm tra tính hợp lệ của TGT:
    - Xác minh chữ ký của AS.
    - Kiểm tra thời gian hiệu lực.
  - Nếu hợp lệ, TGS **cấp Service Ticket (vé dịch vụ) cho Client**.
- **Service Ticket gồm:**
  - ID người dùng.
  - Tên dịch vụ cần truy cập.
  - Thời gian hiệu lực.
  - Khóa phiên giữa Client và Service Server.
- **Mục đích:**
  - Giúp Client có thể truy cập dịch vụ mà không cần cung cấp lại thông tin đăng nhập.

10.3.3 Bước 3: Truy cập dịch vụ (Client → Service Server - SS)

- **Quy trình:**
  - Client gửi **Service Ticket** đến **Service Server (SS)** để yêu cầu truy cập.
  - SS kiểm tra vé:
    - Xác minh chữ ký từ **TGS**.
    - Kiểm tra thời gian hiệu lực của vé.
  - Nếu vé hợp lệ, SS cho phép Client truy cập dịch vụ.
- **Cách đảm bảo bảo mật:**
  - Dữ liệu trong vé được mã hóa bằng khóa bí mật của TGS.
  - Client không thể giả mạo Service Ticket vì không có khóa bí mật này.
- **Mục đích:**
  - Bảo đảm rằng chỉ người dùng hợp lệ mới được truy cập dịch vụ.

10.3.4 Lưu đồ mô tả quy trình xác thực

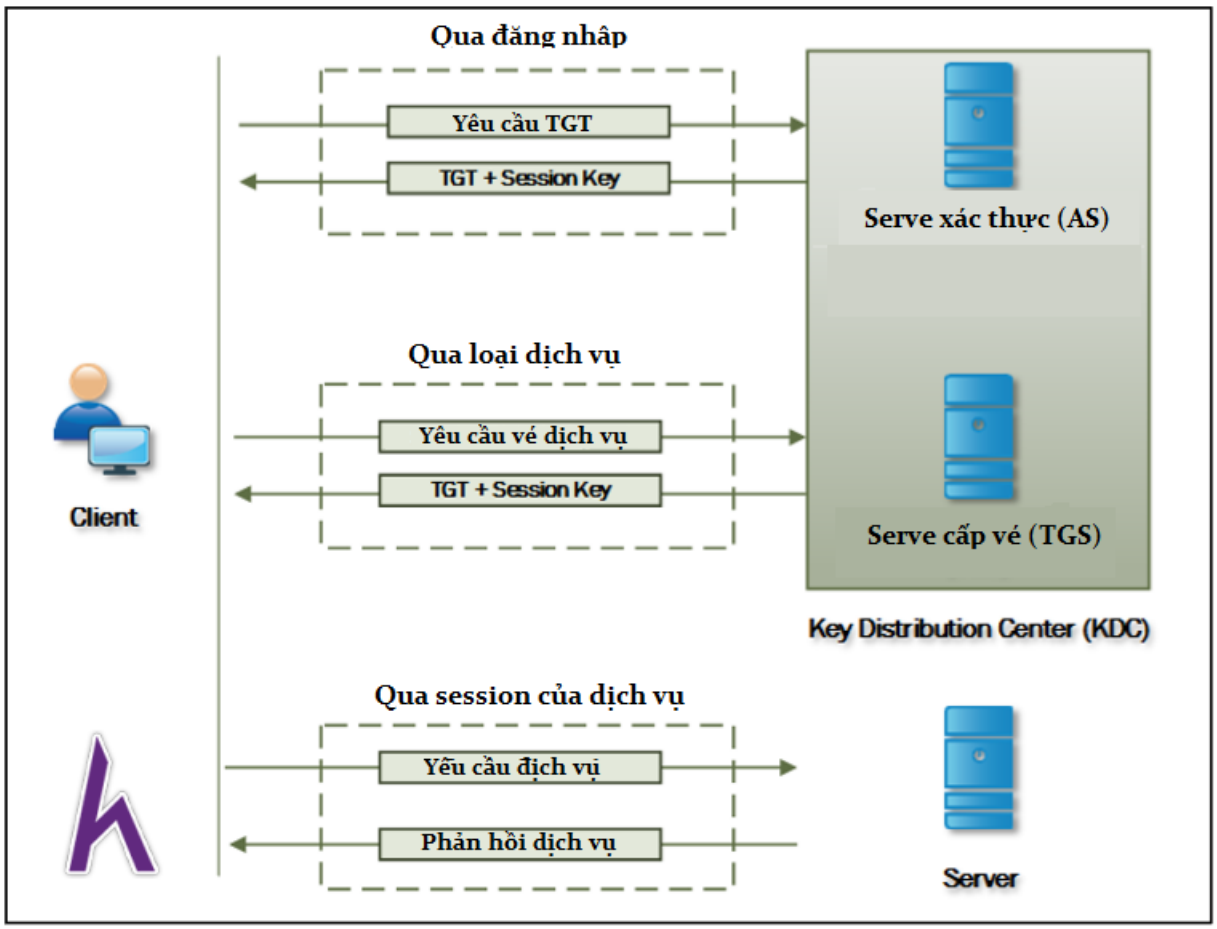


Figure 6-15 Kerberos Authentication Process

10.4 Cơ Chế Mã Hóa & Bảo Mật

10.4.1 Mã hóa đối xứng trong Kerberos

- Kerberos **sử dụng mã hóa đối xứng** để bảo vệ thông tin trao đổi giữa các thành phần.
- **Các thuật toán phổ biến:**
  - **AES (Advanced Encryption Standard)** - thuật toán mạnh, hiệu suất cao.
  - **DES (Data Encryption Standard)** - thuật toán cũ, kém an toàn hơn AES.
- **Cách chia sẻ khóa trong Kerberos**:
  - **Session Key**: được cấp phát cho mỗi phiên làm việc, đảm bảo mỗi lần xác thực có một khóa khác nhau.
  - **Khóa bí mật (Secret Key)**: lưu trữ trong KDC để mã hóa và xác thực vé.
- Mục đích:
  - Đảm bảo dữ liệu **không bị đánh cắp** trong quá trình xác thực.

10.4.2 Cách Kerberos ngăn chặn tấn công

1. **Chống tấn công đánh cắp mật khẩu (Password Sniffing)**
   - **Vấn đề**: Nếu mật khẩu gửi trực tiếp trên mạng, hacker có thể **chặn gói tin** và đánh cắp thông tin.
   - **Cách Kerberos bảo vệ**:
     - Mật khẩu **không bao giờ được gửi trực tiếp** qua mạng.
     - Thay vào đó, **chỉ có vé (ticket) đã mã hóa** được trao đổi giữa Client và Server.
2. Chống tấn công trung gian (Man-in-the-Middle - MITM)
   - Vấn đề: Hacker có thể giả mạo một Authentication Server giả mạo để đánh lừa Client.
   - Cách Kerberos bảo vệ:
     - **Tất cả vé đều được ký số & mã hóa** bởi KDC.
     - Hacker không thể giả mạo vé vì không có **khóa bí mật của KDC**.
3. Chống tấn công phát lại (Replay Attack)
   - Vấn đề: Hacker có thể **bắt gói tin** chứa vé của người dùng hợp lệ và **phát lại** để truy cập trái phép.
   - Cách Kerberos bảo vệ:
     - Vé chứa **dấu thời gian (timestamp)**, chỉ có hiệu lực trong **một khoảng thời gian nhất định**.
     - Service Server kiểm tra timestamp để phát hiện gói tin đã sử dụng lại.

10.5 Ứng Dụng Thực Tế

10.5.1 Kerberos trong Windows Active Directory

- **Active Directory (AD)** của Microsoft sử dụng Kerberos làm giao thức xác thực mặc định kể từ Windows 2000
- Khi người dùng đăng nhập vào hệ thống, máy tính sẽ gửi yêu cầu đến KDC (Authentication Server - AS) để nhận một **Ticket-Granting Ticket (TGT)**
- Sau khi xác thực thành công, **TGT** có thể được sử dụng để lấy **Service Tickets (ST)** nhằm truy cập các dịch vụ khác như **chia sẻ tệp (SMB), đăng nhập vào máy chủ, truy cập hệ thống nội bộ**

10.5.2 Kerberos trong UNIX/Linux

- Kerberos được sử dụng rộng rãi trong các hệ thống Linux và UNIX để cung cấp **Single Sign-On (SSO)** giữa các dịch vụ
- Các hệ thống như **MIT Kerberos, Heimdal Kerberos** hỗ trợ triển khai Kerberos trên các máy chủ Linux
- Dùng trong **Hadoop, PostgreSQL, OpenSSH** để đảm bảo việc xác thực an toàn

10.6 So Sánh Kerberos Với Các Cơ Chế Xác Thực Khác

10.6.1 Kerberos vs OAuth

Tiêu chí	**Kerberos**	**OAuth**
Mô hình	Xác thực dựa trên vé (Ticket)	Xác thực dựa trên Token
Cách hoạt động	Dựa trên KDC và TGT	Dựa trên Access Token
Ứng dụng	Mạng nội bộ, Windows AD	API, Ứng dụng Web, Mobile
Bảo mật	Mạnh hơn nhờ mã hóa đối xứng	Tốt nhưng phụ thuộc vào HTTPS
Ưu điểm	Hỗ trợ SSO tốt, bảo mật cao	Dễ tích hợp với ứng dụng Web

10.6.2 Kerberos vs NTLM

Tiêu chí	**Kerberos**	**NTLM**
Mô hình	Xác thực dựa trên vé (Ticket)	Hash mật khẩu
Bảo mật	Mạnh hơn, giảm nguy cơ đánh cắp	Dễ bị tấn công relay và brute-force
Ứng dụng	Mạng nội bộ, Windows AD	API, Ứng dụng Web, Mobile
Hỗ trợ	Windows, UNIX, Linux	Windows (chủ yếu)

10.6.3 Kerberos vs SAML

Tiêu chí	**Kerberos**	**SAML**
Mô hình	Xác thực dựa trên vé (Ticket)	Dựa trên XML và SSO
Ứng dụng	Mạng nội bộ, hệ thống Windows AD	Cloud, Web SSO
Ưu điểm	Bảo mật tốt, hỗ trợ mã hóa mạnh	Tích hợp tốt với Web

10.7 Hạn Chế và Hướng Phát Triển Của Kerberos

10.7.1 Hạn chế của Kerberos

- **Khó triển khai và cấu hình**
  - Yêu cầu **đồng bộ thời gian** giữa các máy (chênh lệch quá lớn sẽ làm vé không hợp lệ).
  - Cần **máy chủ KDC** luôn hoạt động, nếu KDC bị lỗi, hệ thống xác thực sẽ bị gián đoạn
- **Rủi ro bảo mật**
  - **Nếu KDC bị tấn công**, toàn bộ hệ thống có thể bị xâm nhập
  - **Tấn công replay** có thể xảy ra nếu vé hết hạn chưa được xử lý đúng cách
- **Không phù hợp với môi trường không đồng nhất**
  - Không hỗ trợ tốt các **ứng dụng web hiện đại** so với OAuth
  - Không phù hợp với **ứng dụng di động**

10.7.2 Hướng phát triển của Kerberos

- **Cải thiện bảo mật**
  - **Tích hợp với xác thực nhiều yếu tố (MFA)**: Kết hợp với OTP, sinh trắc học để tăng độ bảo mật
  - **Cải thiện cơ chế quản lý khóa**: Sử dụng thuật toán mã hóa mạnh hơn như AES-256
- **Hỗ trợ môi trường phi tập trung**
  - Hiện tại, Kerberos chủ yếu dùng cho mạng nội bộ, trong tương lai có thể mở rộng để hỗ trợ **Zero Trust Security** và **Cloud Computing**
- **Hỗ trợ nhiều giao thức hiện đại**
  - **Tích hợp với OAuth, OpenID Connect** để hỗ trợ các ứng dụng Web
  - **Tích hợp với Blockchain** để đảm bảo danh tính người dùng

11 Kế hoạch thực thi mô phỏng Kerberos

11.1 Mô tả tổng quan

Dự án triển khai hệ thống Kerberos với các thành phần chính:

- Authentication Server (AS)
- Ticket Granting Server (TGS)
- Service Server (SS)
- Client Authentication Flow

Công nghệ sử dụng:

- Ngôn ngữ: C++.
- Môi trường: Visual Studio Developer 2022.
- Thư viện: OpenSSL AES-256-ECB, SHA-256
- Cơ sở dữ liệu: MySQL (thông qua MySQL Connector/C++)

11.2 Phân công chi tiết

11.2.1 Seminar 1

Giai đoạn 1: Tìm hiểu và chuẩn bị report		
Thành viên	**Nhiệm vụ**	**Thời hạn**
Lê Hoàng Đạt	Các vấn đề về bảo mật, nâng cấp bảo mật, Remote User Authen (mục 4, 5, 6, 7, 8)	25/01 - 27/01
Nguyễn Hồ Đăng Duy	Các giao thức và chuẩn hỗ trợ, ứng dụng thực tiễn, Remote User Authen (mục 3, 4, 5)	25/01 - 27/01
Phạm Quang Duy	Giới thiệu, quy trình, các phương pháp phổ biến, Remote User Authen (mục 1, 2, 3)	25/01 - 27/01

11.2.2 Seminar 2

Giai đoạn 2: Hoàn thành report & Code kerberos server		
Thành viên	**Nhiệm vụ**	**Thời hạn**
Lê Hoàng Đạt	Chuẩn bị doc "Tổng quát và kiến trúc", Chi tiết luồng thực thi giao thức	29/01 - 09/03
Lê Hoàng Đạt	Code Authentication Server (AS), Thiết kế kiến trúc dữ liệu	29/01 - 09/03
Nguyễn Hồ Đăng Duy	"Chuẩn bị doc ""Ứng dụng thực tế và so sánh"", Chuẩn bị kế hoạch, milestone cho implementation	29/01 - 09/03
Nguyễn Hồ Đăng Duy	Code Ticket Granting Server (TGS), Thiết kế kiến trúc code, giao tiếp giữa các thành phần	29/01 - 09/03
Phạm Quang Duy	Chuẩn bị doc "Cơ chế và bảo mật", Pseudo cho giao thức	29/01 - 09/03
Phạm Quang Duy	Hướng dẫn cài đặt môi trường, Code Service server / Client	29/01 - 09/03

11.2.3 Seminar 3

Giai đoạn 3: Xây dựng hoàn chỉnh Kerberos & Xây dựng outline Slide		
Thành viên	**Nhiệm vụ**	**Thời hạn**
Lê Hoàng Đạt	**AS:** Thêm/Xóa/Lưu người dùng vào database	09/03 - 12/03
Lê Hoàng Đạt	**AS:** Hash mật khẩu trước khi lưu xuống Database	09/03 - 12/03
Lê Hoàng Đạt	**AS:** Xác thực người dùng bằng mật khẩu đã hash	09/03 - 12/03
Lê Hoàng Đạt	**AS:** Generate TGT/session key ngẫu nhiên, có thời gian hết hạn	09/03 - 12/03
Lê Hoàng Đạt	Viết hàm main để chạy chương trình	13/03 - 15/03
Nguyễn Hồ Đăng Duy	**TGS:** Generate ST/session key ngẫu nhiên, có thời gian hết hạn	09/03 - 12/03
Nguyễn Hồ Đăng Duy	**TGS:** Lưu log các service ticket đã được cấp (người dùng, dịch vụ, thời gian tạo, hết hạn)	09/03 - 12/03
Nguyễn Hồ Đăng Duy	**TGS:** Thu hồi service ticket	09/03 - 12/03
Nguyễn Hồ Đăng Duy	Viết hàm Encrypt / Decrypt bằng OpenSSL	13/03 - 15/03
Phạm Quang Duy	**SS:** Thêm một danh sách các dịch vụ hợp lệ (unordered_map và lưu trong database)	09/03 - 12/03
Phạm Quang Duy	**SS:** Khi user yêu cầu truy cập, kiểm tra xem service đó có tồn tại không	09/03 - 12/03
Phạm Quang Duy	**SS:** Lưu vào database lịch sử truy cập	09/03 - 12/03
Phạm Quang Duy	Slide thuyết trình - outline	13/03 - 15/03

Giai đoạn 4: Merge các branch & Test & Chuẩn bị nội dung slide		
Thành viên	**Nhiệm vụ**	**Thời hạn**
Lê Hoàng Đạt	**AS:** Debug lại các chức năng & Đảm bảo liên kết tốt với Database	16/03 - 19/03
Lê Hoàng Đạt	**Slide:** Nội dung: Kerberos authentication	19/03 - 23/03
Nguyễn Hồ Đăng Duy	**TGS:** Debug lại các chức năng & Đảm bảo liên kết tốt với Database	16/03 - 19/03
Nguyễn Hồ Đăng Duy	**Slide:** Nội dung: Kerberos simulationL	19/03 - 23/03
Phạm Quang Duy	**SS:** Debug lại các chức năng & Đảm bảo liên kết tốt với Database	16/03 - 19/03
Phạm Quang Duy	Clean up code	19/03 - 23/03
Phạm Quang Duy	Slide: Nội dung: User authentication & Kerberos with others	19/03 - 23/03

Giai đoạn 5: Báo cáo seminar & Nộp bài		
Thành viên	**Nhiệm vụ**	**Thời hạn**
Phạm Quang Duy	**Soạn script:** Mô phỏng	23/03 - 02/04
Nguyễn Hồ Đăng Duy	**Soạn script:** KBR	23/03 - 02/04
Lê Hoàng Đạt	**Soạn script:** User authen	23/03 - 02/04
Nguyễn Hồ Đăng Duy	**Report:** Sửa kế hoạch + thêm simulation	23/03 - 09/04
Lê Hoàng Đạt	Vẽ flowchart mô phỏng Kerberos	23/03 - 02/04
Phạm Quang Duy	Demo dự án	23/03 - 02/04
Phạm Quang Duy	Hoàn thiện slide	23/03 - 02/04

12 Thực thi mô phỏng

12.1 Cài đặt môi trường

12.1.1 Cài Đặt OpenSSL, Boost bằng Vcpkg

Chương trình sử dụng OpenSSL, Boost để thực hiện mã hóa dữ liệu. Nếu bạn chưa cài đặt Vcpkg, hãy làm theo các bước sau:

Tải về và cài đặt Vcpkg

```
git clone https://github.com/microsoft/vcpkg.git
cd vcpkg
./bootstrap-vcpkg.bat  # Windows
./bootstrap-vcpkg.sh   # Linux/macOS
```

Cài đặt OpenSSL

Trên Windows:

```
vcpkg install openssl:x64-windows
```

Trên Linux/macOS:

```
vcpkg install openssl
```

Cài đặt Boost

```
vcpkg install boost
```

Cài đặt mysqlcppconnector

Truy cập trang tải xuống của MySQL: https://dev.mysql.com/downloads/connector/cpp/ để tải .zip (giải nén) hoặc .msi (cài đặt tự động)

12.1.2 Cấu hình Visual Studio

Tất cả ở chế độ x64, RELEASE

Cấu hình mysqlcppconnector

Thực hiện cấu hình theo video sau.

Cấu hình OpenSSL

Cách 1: Tự động tích hợp vào Visual Studio: Chạy lệnh sau trong cmd hoặc PowerShell (ở thư mục chứa vcpkg):

```
vcpkg integrate install
```

Cách 2: Cấu hình thủ công nếu cách 1 không hoạt động. Thêm đường dẫn thư viện và include:

1. Mở Visual Studio.
2. Vào Project Properties (Nhấn chuột phải vào Project → Properties).
3. Thêm đường dẫn thư viện:
   - Configuration Properties → VC++ Directories → Library Directories.
   - Thêm đường dẫn thư viện của OpenSSL từ vcpkg, ví dụ:

```
C:/path/to/vcpkg/installed/x64-windows/lib
```

4. Thêm đường dẫn include:
   - Configuration Properties → C/C++ → General → Additional Include Directories.
   - Thêm đường dẫn:

```
C:/path/to/vcpkg/installed/x64-windows/include
```

5. Thêm OpenSSL vào Linker:
   - Configuration Properties → Linker → Input → Additional Dependencies.
   - Thêm các file .lib của OpenSSL:

```
libcrypto.lib
libssl.lib
```

12.2 Cấu trúc dữ liệu

USERS - Lưu thông tin người dùng		
Trường	**Kiểu dữ liệu**	**Mô tả**
id	INT PRIMARY KEY AUTO_INCREMENT	ID duy nhất của user
username	VARCHAR(255), UNIQUE	Tên đăng nhập
password_hash	VARCHAR(255)	Mật khẩu đã hash (SHA-256)
created_at	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Thời gian tạo user

SERVICES - Danh sách dịch vụ trong hệ thống		
Trường	**Kiểu dữ liệu**	**Mô tả**
id	INT PRIMARY KEY AUTO_INCREMENT	ID dịch vụ
service_name	VARCHAR(255), UNIQUE	Tên dịch vụ
service_key	VARCHAR(255)	Khóa bí mật của dịch vụ
created_at	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Thời gian thêm dịch vụ

SERVICES_TICKETS - Lưu thông tin Service Ticket (ST)		
Trường	**Kiểu dữ liệu**	**Mô tả**
id	INT PRIMARY KEY AUTO_INCREMENT	ID ticket
username	VARCHAR(255), FOREIGN KEY	Người dùng yêu cầu
service_name	VARCHAR(255), FOREIGN KEY	Dịch vụ yêu cầu
ticket_data	TEXT	Dữ liệu Service Ticket (đã mã hóa)
session_key	VARCHAR(255)	Session key của phiên làm việc
issued_at	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Thời điểm cấp
expires_at	TIMESTAMP	Thời điểm hết hạn

LOGS - Lịch sử truy cập dịch vụ		
Trường	**Kiểu dữ liệu**	**Mô tả**
id	INT PRIMARY KEY AUTO_INCREMENT	ID log
username	VARCHAR(255), FOREIGN KEY	Người dùng truy cập
service_name	VARCHAR(255), FOREIGN KEY	Dịch vụ truy cập
access_time	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Thời gian truy cập
status	VARCHAR(50)	Trạng thái truy cập (Thành công/Thất bại)

12.3 Cấu trúc code

main.cpp

Chức năng: Entry point để chạy toàn bộ luồng xác thực Kerberos.
Các bước chính:

1. Kết nối CSDL.
2. Khởi tạo các server: AS, TGS, SS.
3. Tạo Client, xác thực, lấy TGT, xin Service Ticket.
4. Truy cập Service.

authentication_server.h/.cpp

Chức năng: Quản lý người dùng và sinh TGT.
Các bước chính:

1. AddUser, RemoveUser, AuthenticateUser.
2. Generate_TGT: Sinh TGT chứa (username, session_key, expiration), mã hóa bằng master key.
3. Mã hóa session_key bằng mật khẩu user → SSK1
4. Ghi log TGT vào bảng tgt_logs

ticket_granting_server.h/.cpp

Chức năng: Xác thực TGT và cấp Service Ticket.
Các bước chính:

1. Validate_TGT: Giải mã TGT, kiểm tra hạn
2. Generate_Service_Ticket: Sinh ST chứa (username, session_key_2, service_name, expiration)
3. Mã hóa ST bằng service_key tương ứng từ DB
4. Ghi log ST vào bảng service_tickets
5. Xóa ST hết hạn (RemoveExpiredTickets)

service_server.h/.cpp

Chức năng: Xác thực ST và cấp quyền truy cập dịch vụ.
Các bước chính:

1. Validate_Service_Ticket: Giải mã ST, kiểm tra hạn và tên dịch vụ
2. Grant_Access: Ghi log Success hoặc Failed vào bảng logs
3. Add_Service, Remove_Service

client.h/.cpp

Chức năng: Mô phỏng người dùng gửi yêu cầu xác thực và truy cập dịch vụ.
Các biến lưu trữ:

1. encrypted_TGT, encrypted_service_ticket
2. session_key_1, session_key_2

kerberos_protocol.h/.cpp

Chức năng: Đóng vai trò **điều phối** toàn bộ luồng giao tiếp giữa Client và các server.

encryption.h/.cpp

Chức năng: Cung cấp các hàm mã hóa/giải mã AES-256-ECB.
Dùng OpenSSL để:

1. Encrypt(string, key).
2. Decrypt(string, key).
3. GenerateRandomKey() → sinh session key 32 bytes.

database.h/.cpp

Chức năng: Giao tiếp với MySQL thông qua thư viện mysql-connector-c++.
Hỗ trợ:

1. executeQuery: câu lệnh không trả dữ liệu (INSERT, DELETE,...)
2. executeSelectQuery: trả về dữ liệu dạng vector<map<string,string> >.

global.h

Chức năng: Khai báo extern string kdc_master_key để dùng ở các nơi cần.

12.4 Luồng chương trình

Hệ thống Kerberos trong chương trình được chia làm 3 bước xác thực chính, tương ứng với giao thức chuẩn:

Bước 1: Client → Authentication Server (AS) – Yêu cầu TGT

- Người dùng nhập **username** và **password**.
- **AS** kiểm tra thông tin xác thực:
  - **password** được băm bằng thuật toán **SHA-256**.
  - So sánh với giá trị hash lưu trong database.
- Nếu hợp lệ, AS sinh ra:
  - **Session key 1 (ssk1)**: 32 bytes ngẫu nhiên.
  - **TGT (Ticket Granting Ticket)**: có cấu trúc như sau:

    username | ssk1 | expiration_time
  - **TGT** được mã hóa bằng **kdc_master_key** sử dụng **AES-256-ECB**.
  - **ssk1** được mã hóa bằng **password** người dùng và trả về cho client.
  - Trả về encrypted_ssk1, encrypted_TGT.

Bước 2: Client → Ticket Granting Server (TGS) – Yêu cầu ST

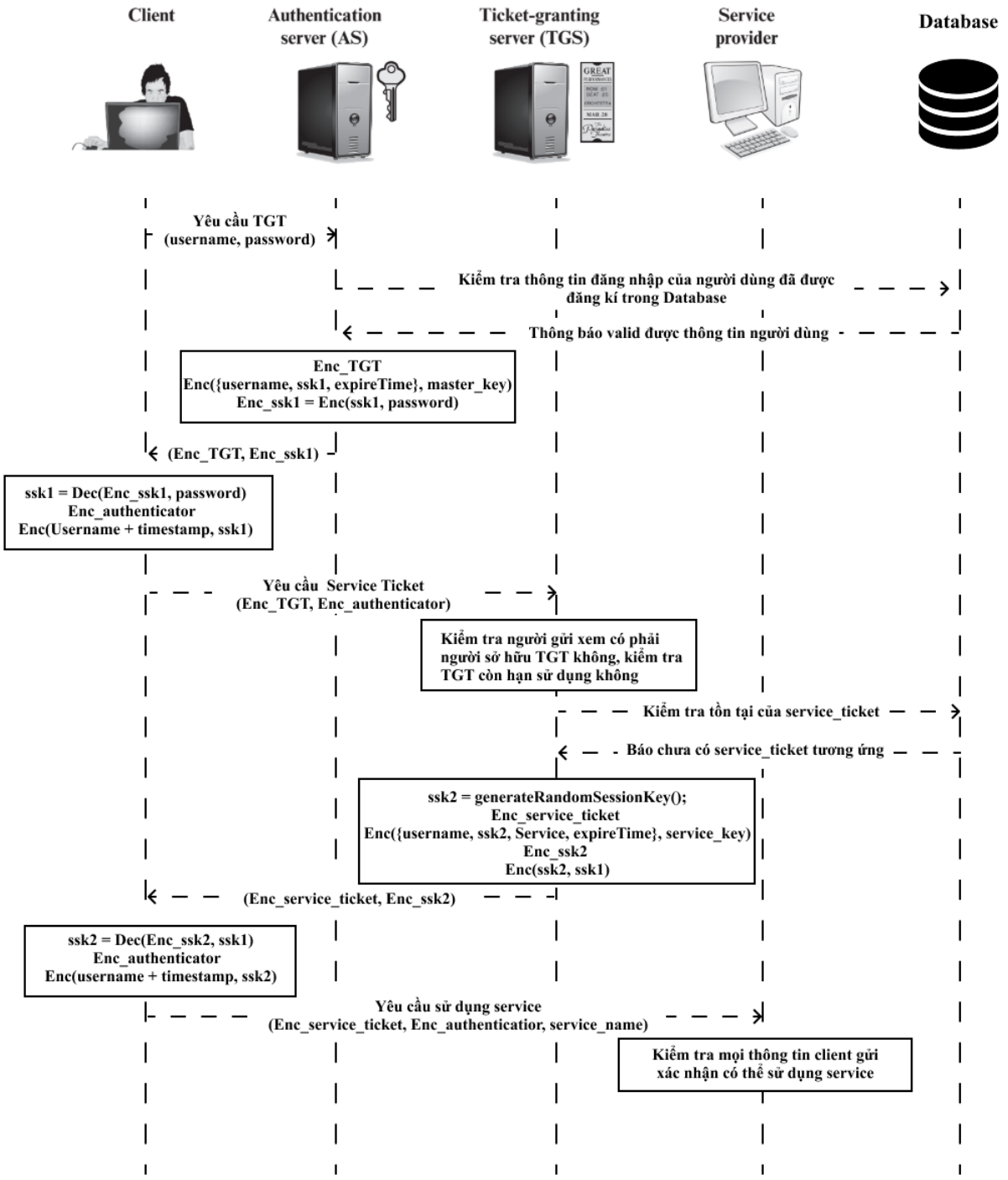
- Client giải mã **ssk1** bằng **password**.
- Tạo **Authenticator = username | timestamp** và mã hóa bằng **ssk1**.
- Gửi **encryptedTGT, encryptedAuthenticator** và **service_name** tới TGS.
- **TGS** kiểm tra:
  - Giải mã TGT bằng **kdc_master_key**.
  - Giải mã Authenticator bằng **ssk1**.
  - Kiểm tra **username**, kiểm tra thời hạn.
- Nếu hợp lệ, TGS sinh:
  - **Session key 2 (ssk2)**: 32 bytes ngẫu nhiên.
  - **Service Ticket (ST)**:

    username | ssk2 | service_name | expiration
  - **ST** được mã hóa bằng **service_secret_key** (của service).
- Mã hóa **ssk2** bằng **ssk1**
- Client nhận lại **encrypted_ssk2** và **encrypted_ST**.
- Ghi log vào **service_tickets** DB.

Bước 3: Client → Service Server – Truy cập dịch vụ

- Client giải mã **ssk2** từ **encrypted_ssk2** bằng **ssk1**.
- Tạo **Authenticator** mới: **Authenticator = username | timestamp** và mã hóa bằng **ssk2**.
- Gửi **encrypted_ST, encrypted_authenticator** đến SS.
- Server thực hiện:
  - Giải mã ST bằng **service_secret_key**.
  - Giải mã Authenticator bằng **ssk2**
  - So khớp **username, service_name**, thời hạn và xác thực database.
- Nếu hợp lệ, client được cấp quyền truy cập dịch vụ và ghi log vào bảng **logs** với trạng thái **Success/Failed**.

Luồng của chương trình được minh họa cụ thể bằng hình sau đây:



Hình 1: Flowchart mô tả luồng xử lý

12.5 Mã giả - Pseudo code

Encrypt a plaintext using AES-256-ECB

```
1: procedure ENCRYPT(plaintext, key)
2:     Pad key to 32 bytes if necessary
3:     ciphertext ← AES256-ECB-Encrypt(plaintext, key)
4:     return HexEncode(ciphertext)
5: end procedure
```

Decrypt a ciphertext using AES-256-ECB

```
1: procedure DECRYPT(hexCiphertext, key)
2:     Pad key to 32 bytes if necessary
3:     ciphertext ← HexDecode(hexCiphertext)
4:     plaintext ← AES256-ECB-Decrypt(ciphertext, key)
5:     return plaintext
6: end procedure
```

Hash a password using SHA-256

```
1: procedure HASHPASSWORD(password)
2:     digest ← SHA256(password)
3:     return HexEncode(digest)
4: end procedure
```

Phase 1: Authenticate with AS and receive TGT

```
1: procedure PHASE1_AS(username, password)
2:     if User not in DB or wrong password then
3:         return Failure
4:     end if
5:     sessionKey_1 ← GenerateRandomKey()
6:     expiration ← current_time + 8 hours
7:     TGT ← Encrypt(username‖sessionKey_1‖expiration, K_AS)
8:     SSK1 ← Encrypt(sessionKey_1, password)
9:     return (SSK1, TGT)
10: end procedure
```

Phase 2: Request Service Ticket from TGS

procedure PHASE2_TGS(*username*, *TGT*, *SSK*1, *serviceName*, *password*)
    $sessionKey_1 \leftarrow$ Decrypt(*SSK*1, *password*)
    $authenticator \leftarrow$ Encrypt($username \| timestamp$, $sessionKey_1$)
    $TGT\_data \leftarrow$ Decrypt(*TGT*, K_{AS})
    Check validity: user match, time valid
    $sessionKey_2 \leftarrow$ GenerateRandomKey()
    expiration ← current_time + 1 hour
    $SSK2 \leftarrow$ Encrypt($sessionKey_2$, $sessionKey_1$)
    ServiceKey ← Lookup from DB(*serviceName*)
    ServiceTicket ← Encrypt($username \| sessionKey_2 \| serviceName \| expiration$, *ServiceKey*)
    Store to DB: LogServiceTicket(*username*, *serviceName*, *ServiceTicket*, *SSK*2, *expiration*)
    return (*SSK*2, *ServiceTicket*)
end procedure

Phase 3: Access service using service ticket

procedure PHASE3_SERVICE(*username*, *SSK*2, *ServiceTicket*, *serviceName*, $sessionKey_1$)
    $sessionKey_2 \leftarrow$ Decrypt(*SSK*2, $sessionKey_1$)
    $authenticator \leftarrow$ Encrypt($username \| timestamp$, $sessionKey_2$)
    ticketData ← Decrypt(*ServiceTicket*, *ServiceKey*)
    Check: username, expiration, serviceName match
    if All valid **then**
        LogAccess(*username*, *serviceName*, "Success")
        return Access Granted
    else
        LogAccess(*username*, *serviceName*, "Failed")
        return Access Denied
    end if
end procedure

Log Service Ticket to Database

procedure LOGSERVICETICKET(*username*, *serviceName*, *ticket*, *ssk*2, *expiration*)
    SQL ← `INSERT INTO service_tickets (username, service_name, ticket_data, session_key, expires_at)`
    Execute SQL with values
end procedure

Log Access to Service

```
procedure LogAccess(username, serviceName, status)
    time ← current timestamp
    SQL ← INSERT INTO logs (username, service_name, access_time, status)
    Execute SQL with values
end procedure
```

Generate Random Session Key (32 bytes)

```
procedure GenerateRandomKey
    key ← random 32-character hexadecimal string
    return key
end procedure
```

12.6 Mã nguồn dự án

Github của dự án: `https://github.com/YuD1405/kerberos_simulation.git`

Tài liệu

[1] Lanchec, S. (2024, June 5). User Authentication: a guide for developers. Forest Admin Blog. https://www.forestadmin.com/blog/user-authentication-a-guide-for-developers/

[2] The five different types of authentication — WorkOS. (n.d.). WorkOS. https://workos.com/blog/the-five-different-types-of-authentication

[3] What is Authentication? Definition and uses - Auth0. (n.d.). Auth0. https://auth0.com/intro-to-iam/what-is-authentication

[4] GeeksforGeeks. (2024, June 6). What is User Authentication, and Why is it Important? GeeksforGeeks. https://www.geeksforgeeks.org/what-is-user-authentication-and-why-is-it-important/

[5] Frontegg. (2025, January 9). Authentication: What It is and How It Works | Frontegg. Frontegg. https://frontegg.com/blog/authenticationAPI_Authentication_Methods

[6] Sakshyam Shah. (2022, February 25). 6 Authentication best practices. Teleport. https://goteleport.com/blog/authentication-best-practices/