Complete Guide to AI Agents, Identity, and Security

From Basics to Secure Implementation

Table of Contents

Lesson 1: Introduction to AI Agents

What is an LLM (Large Language Model)?

A Large Language Model (LLM) is a program that can understand and generate human-like text. Examples include ChatGPT, Claude, and other conversational AI systems. These models are trained on vast amounts of text data, enabling them to:

- Answer questions
- Write code
- Have natural conversations
- Explain complex topics

Limitations of Basic LLMs

LLMs have significant constraints:

- **Knowledge cutoff**: They only know information up to their training date
- **No real-world actions**: They can only discuss topics, not interact with systems
- **No tool access**: Cannot check email, create events, or use applications

What is an AI Agent?

An AI agent is an LLM enhanced with the ability to **take real-world actions** on your behalf. Think of it as upgrading from an advisor to an executive assistant.

Examples of AI Agent capabilities:

- Read and send emails
- Create tasks in project management tools
- Schedule calendar meetings
- Post messages in Slack
- Access and process company documents
- Make purchases or transactions
- Update databases

Key Distinction

Feature	Basic LLM	AI Agent
Conversation	✓	✓
Information retrieval	✓	✓
Take actions	✗	✓
Access external tools	✗	✓
Act on user's behalf	✗	✓

Lesson 2: Why Identity Matters for AI Agents

The Trust Problem

When an AI agent can take real actions, critical questions arise:

- How does Gmail know this AI is acting on YOUR behalf?
- What if the AI makes a mistake with sensitive data?
- If something goes wrong, how do you prove what the AI did?
- How do you ensure the AI only does what you authorize?

What is Identity in This Context?

Identity management for AI agents encompasses:

1. **Agent identification**: Which specific agent is this?

2. **User representation**: Who does the agent represent?
3. **Permission scope**: What actions is the agent authorized to perform?
4. **Audit trail**: Can we track and review all actions taken?

The Critical Principle

> **"Without identity, there is no accountability"**

Without proper identity management, you cannot:

- Trace problems back to their source
- Limit access to sensitive resources
- Hold parties responsible for mistakes
- Build trust in AI-powered systems

Real-World Analogy: Power of Attorney

Consider legal power of attorney:

- Requires legal identity documents
- Specifies exact authorized actions
- Creates a paper trail of all activities
- Provides accountability mechanisms for abuse

AI agents require the same framework—digital identity with accountability.

Risks Without Identity Management

- **Data breaches**: Unlimited access increases attack surface
- **Unauthorized actions**: No way to prevent or limit mistakes
- **No audit trail**: Cannot investigate incidents
- **Compliance violations**: Regulatory requirements unmet
- **Loss of trust**: Users won't adopt systems they can't trust

---

Lesson 3: How AI Agents Actually Work (Tool Calling)

The Evolution of AI Capabilities

Stage 1: Basic LLM

- Responds based solely on training data
- Cannot access real-time information
- Example: "What's the weather?" → "I don't have current weather data"

Stage 2: LLM + RAG (Retrieval Augmented Generation)

- Pulls real-time data from documents and databases
- Provides context-aware responses
- Example: "What's in our company handbook?" → Searches and answers
- Still cannot take actions

Stage 3: LLM + Tool Calling (True AI Agent)

- Can execute functions and interact with services
- Example: "Schedule a meeting with John tomorrow at 2pm" → Creates the event
- Full action-taking capability

Understanding Tool Calling

A **tool** is a function the AI agent can execute. Tools represent actions or verbs.

Common tool examples:

```
send_email(to, subject, body)
create_task(title, due_date, priority)
post_slack_message(channel, text)
search_github(repository, query)
charge_credit_card(amount, customer_id)
get_weather(location)
search_database(table, criteria)
```

How Tool Calling Works: Step-by-Step

User request: "Send an email to Sarah saying the report is ready"

Step 1: AI Analysis

- Identifies need to send email
- Recognizes available `send_email` tool
- Extracts required parameters

Step 2: Tool Invocation

```javascript
send_email(
  to: "sarah@company.com",
  subject: "Report Ready",
  body: "The report is ready for your review."
)
```

Step 3: Execution

- Tool connects to Gmail API
- Email is sent using user's credentials
- Result is returned to agent

Step 4: Confirmation

- Agent reports: "Done! I've sent the email to Sarah."

Multi-Tool Workflows

AI agents can chain multiple tools to complete complex tasks.

Example request: "Prepare for my client meeting tomorrow"

Agent workflow:

1. `check_calendar()` → Identifies meeting attendees
2. `search_emails(client_name)` → Finds recent correspondence
3. `search_documents(client_name)` → Retrieves project files
4. `create_summary(data)` → Generates briefing document
5. `send_slack(team_channel)` → Notifies team of preparation

All executed from a single user request!

The Security Challenge

To use tools, AI agents require **credentials**:

- API keys
- Access tokens
- Passwords
- Authentication certificates

Insecure approach: Give the AI agent all your passwords and full access credentials.

Problems:

- If the agent is compromised, everything is exposed
- No limits on what the agent can do
- No granular control or oversight
- Complete system access with no guardrails

Secure approach: This is what the remainder of this guide addresses—providing safe, limited, auditable access.

Lesson 4: The Security Solution (Tokens & Scoped Access)

The Core Problem

You want your AI agent to send emails without:

- Deleting emails
- Accessing unrelated accounts (banking, personal files)
- Having permanent, unrestricted access
- Operating without oversight

Solution: Implement tokens with scoped access and time limitations.

Solution 1: Tokens Instead of Passwords

What is a token?

A token is a temporary, limited-use credential that provides access to specific resources.

Real-world analogy:

Concept	Password	Token
Analogy	House master key	Hotel room keycard
Duration	Permanent	Expires at checkout
Scope	Opens everything	Opens only your room
Revocation	Requires lock change	Deactivate card instantly

Technical implementation:

```
❌ BAD APPROACH:
agent_credential = "myGmailPassword123"
// Full access, permanent, all services

✓ GOOD APPROACH:
agent_token = "temp_gmail_send_expires_1hr_abc123"
// Limited scope, expires soon, single service
```

Token characteristics:

- **Service-specific**: Only works for designated service (Gmail, not Slack)
- **Permission-limited**: Only specific actions allowed (send, not delete)
- **Time-bound**: Automatically expires after set duration
- **Revocable**: Can be instantly invalidated if compromised

Solution 2: Scoped Access (Least Privilege)

What are scopes?

Scopes define the exact permissions granted by a token.

Gmail scope examples:

```
gmail.send        → Can send emails only
gmail.read        → Can read emails only
gmail.modify      → Can read and send emails
gmail.compose     → Can draft but not send
gmail.full_access → Complete control (avoid!)
```

The Principle of Least Privilege:

> Grant only the minimum permissions necessary to accomplish the task.

Example scenario:

Task: "Check if I have any meetings today"

- ❌ **Bad**: `calendar.full_access` (can delete all meetings!)
- ✓ **Good**: `calendar.read_only` (can only view)

Multi-scope example:

Task: "Summarize my unread emails and create a task list"

Required scopes:

```
gmail.read        → Read unread emails
tasks.create      → Create new tasks
tasks.list.read   → View existing tasks
```

No need for:

```
gmail.send        ✗
gmail.delete      ✗
tasks.delete      ✗
calendar.modify   ✗
```

Solution 3: Short-Lived Credentials

Why tokens expire:

If a token is stolen or intercepted, expiration limits the damage window.

Token lifespan hierarchy:

Access Token:

- **Duration**: 15 minutes to 1 hour
- **Purpose**: Immediate action execution
- **Security**: If stolen, attacker has limited time
- **Usage**: Direct API calls

Refresh Token:

- **Duration**: Days to weeks
- **Purpose**: Obtain new access tokens
- **Security**: Stored more securely, rarely transmitted
- **Usage**: Background renewal process

Token lifecycle example:

1. User authorizes AI agent
2. Agent receives access token (expires in 1 hour)
3. Agent uses token to send email
4. After 1 hour, access token expires
5. Agent uses refresh token to get new access token
6. Process repeats as needed

Security benefit:

Even if an attacker intercepts an access token, it becomes useless within an hour.

Solution 4: Token Vault (Secure Storage)

The problem:

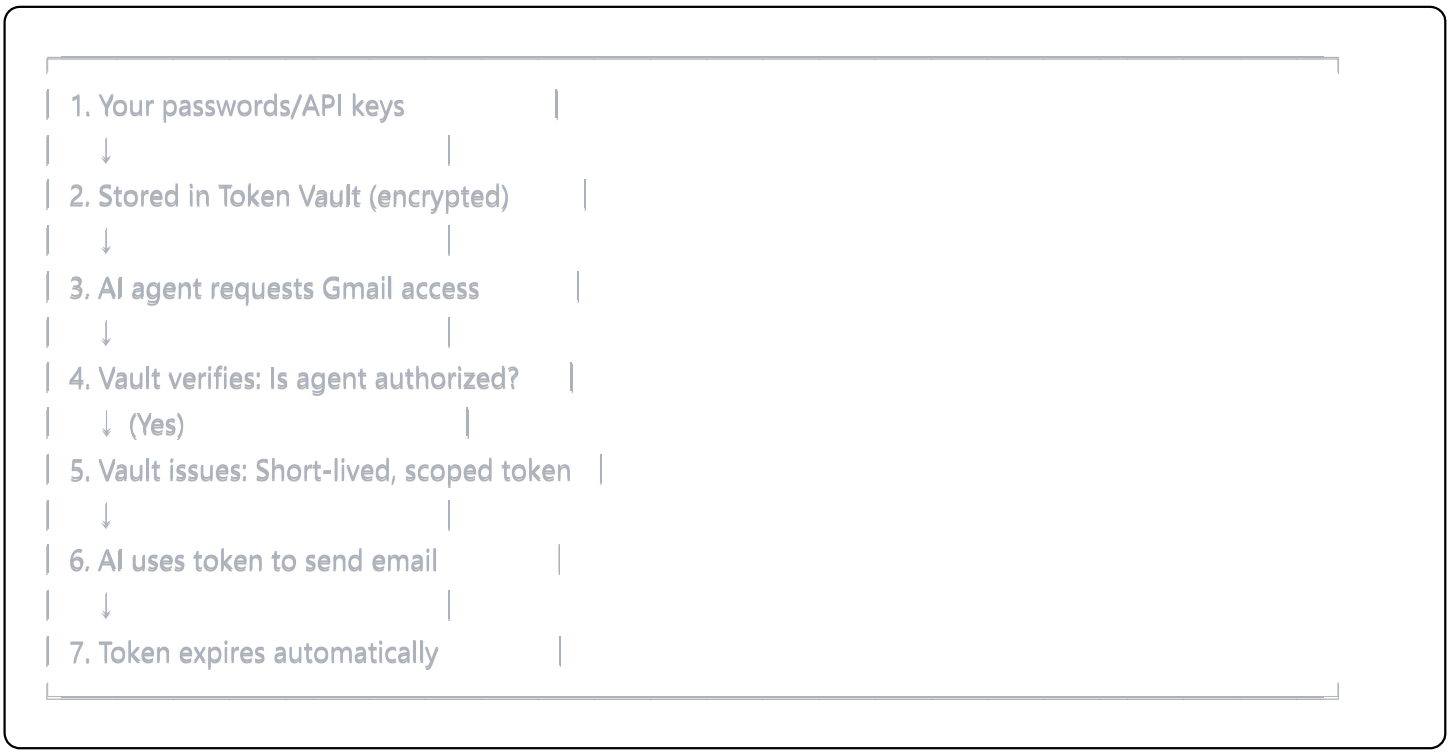
Tokens are sensitive credentials. If compromised, attackers can impersonate users.

The Token Vault solution:

A token vault is a secure, centralized credential management system that:

- Stores all sensitive credentials encrypted
- Issues tokens only when authorized and needed
- Never exposes underlying passwords
- Logs every token issuance and usage
- Enables instant access revocation

How it works:



Benefits:

- **Credential isolation**: Real passwords never leave the vault
- **Temporary access**: AI only gets time-limited tokens
- **Audit logging**: Complete visibility into access patterns
- **Centralized control**: Single point for revoking all access
- **Reduced attack surface**: Compromising agent doesn't expose vault

Hotel analogy revisited:

Component	Hotel Equivalent
Password	Master key (kept secure, never shared)
Token Vault	Front desk (manages access)
Access Token	Room keycard (limited, expires)
Scopes	Keycard permissions (room, gym, pool)
Expiration	Checkout time (key deactivates)

Lesson 5: The Four Pillars of Offzero

Offzero provides a comprehensive identity and security platform for AI agents, built on four foundational pillars.

Pillar 1: User Authentication

What is authentication?

Authentication is the process of verifying identity—proving you are who you claim to be.

Traditional authentication:

- Username + Password
- User enters credentials
- System validates match
- Access granted if correct

Modern authentication methods supported:

1. Social Login

- Sign in with Google, Facebook, Microsoft, etc.
- Leverages existing trusted accounts
- No new passwords to create or remember
- OAuth-based secure delegation

2. Passkeys (Most Secure)

- Uses device biometrics (Face ID, fingerprint)
- Cryptographic key pairs (private key never leaves device)
- Phishing-resistant
- No passwords to steal or forget

3. Biometrics

- Fingerprint scanning
- Facial recognition
- Voice authentication
- Iris scanning

4. Single Sign-On (SSO)

- One login for entire organization

- Centralized identity provider
- Common in enterprise environments
- SAML or OpenID Connect protocols

Why this matters for AI agents:

Before an AI agent can act on your behalf, the system must verify YOUR identity with strong authentication. Weak authentication = easy impersonation.

Pillar 2: Token Vault

(Covered in detail in Lesson 4)

Quick summary:

- Securely stores credentials encrypted at rest
- Issues short-lived, scoped tokens to AI agents
- Never exposes underlying passwords or API keys
- Enables instant access revocation
- Provides complete audit logging

Why it's a pillar:

The Token Vault is the foundation of secure credential management, preventing password sharing while enabling controlled access.

Pillar 3: Asynchronous Authorization (Async Off)

The problem:

Some actions are too sensitive for automatic execution:

- Transferring large amounts of money
- Deleting critical data
- Terminating employees
- Signing legal contracts
- Publishing company announcements

The solution: Just-in-time human approval

How Asynchronous Authorization works:

1. AI agent determines it needs to perform sensitive action
2. System PAUSES execution
3. Notification sent to user (phone, email, dashboard)
4. User reviews contextual details
5. User approves or denies
6. AI proceeds based on decision

Example notification:

📱 APPROVAL REQUIRED

Your AI agent wants to:
Transfer $10,000 to Vendor ABC
Account: Business Checking
Reference: Invoice #12345

Context:
- Vendor verified ✓
- Invoice matches PO ✓
- Within budget ✓

[Approve] [Deny] [Review Details]

Why "Asynchronous"?

"Asynchronous" means non-immediate—there's a delay while awaiting human approval. The process doesn't block other operations, but this specific action waits.

Real-world analogy:

Bank dual authorization:

- Teller initiates large wire transfer
- Manager must approve before execution
- Transaction "waits" in pending state
- Provides oversight for high-risk operations

Benefits:

- **Human oversight**: Critical decisions require human judgment
- **Error prevention**: Catch mistakes before execution
- **Audit trail**: Record of who approved what and when
- **Context preservation**: User sees full situation before deciding

- **Flexibility**: User can modify, delay, or cancel

Configuration examples:

```
IF action = "money_transfer" AND amount > $5,000
  THEN require_approval

IF action = "delete_data" AND scope = "production"
  THEN require_approval

IF action = "send_email" AND recipients > 100
  THEN require_approval

IF action = "any" AND time = "after_hours"
  THEN require_approval
```

Pillar 4: Fine-Grained Authorization

What is fine-grained authorization?

Fine-grained authorization provides highly specific, context-aware access control—not just "yes/no" but "yes, if conditions X, Y, Z are met."

Granularity comparison:

```
Coarse-grained: "Can access Gmail"
Fine-grained:   "Can send Gmail to internal recipients,
                 Monday-Friday 9am-5pm,
                 max 50 emails per day,
                 only from office IP address"
```

Types of fine-grained control:

1. Time-based restrictions

```
- Agent operates only during business hours (9am-5pm)
- No weekend access
- Holiday blackout periods
- Emergency access expires after 24 hours
- Specific action windows (backups run 2-4am only)
```

2. Location-based restrictions

```
- Access only from office IP addresses
- Blocks access from foreign countries
- Requires company VPN connection
- Geo-fencing for physical location
- Different permissions for remote vs. on-site
```

3. Context-based restrictions

```
- Can only process orders under $1,000
- Can only access files marked "Public" or "Internal"
- Can only email previous contacts
- Can only modify own department's data
- Resource limits (max 100 API calls per hour)
```

4. Conditional logic

```
IF amount > $5,000 THEN require_approval
IF after_hours THEN read_only_access
IF new_vendor THEN additional_verification
IF data_classification = "confidential" THEN block
IF user_risk_score > threshold THEN deny
```

Detailed examples:

Example 1: Customer Support AI Agent

Permissions granted:
✓ Read customer support tickets (all)
✓ Reply to tickets (assigned to user)
✓ Auto-refund up to $100 (no approval)
⚠️ Refund $100-$1,000 (requires approval)
✗ Refund over $1,000 (blocked)
✓ Access customer contact info
✗ Access payment methods
✗ Access customer purchase history
✓ Create internal notes
✗ Delete tickets
✗ Reassign tickets to other agents

Time restrictions:
✓ 24/7 read access
✓ Reply 6am-10pm only
✗ Refunds only during business hours

Rate limits:
- Max 50 responses per hour
- Max 10 refunds per day

Example 2: Development AI Agent

Permissions granted:
✓ Read all code repositories
✓ Create feature branches
✓ Submit pull requests
✓ Comment on issues
✗ Merge to main branch
✗ Delete repositories
✗ Modify CI/CD configuration
✓ Run test suites
✗ Deploy to production
⚠️ Deploy to staging (with approval)

Time restrictions:
✓ Development work 8am-10pm
✗ Production access after hours
✓ Code review anytime

Location restrictions:
✓ Require company VPN
✓ Allow from verified IP ranges
✗ Block from blacklisted countries

Context restrictions:
- Max 100 API calls per hour
- Cannot modify files over 10,000 lines without review
- Cannot access secrets or credentials

Example 3: Finance AI Agent

Permissions granted:
✓ Read financial reports
✓ Generate expense summaries
✓ Auto-approve expenses < $500
⚠️ Expenses $500-$5,000 require manager approval
⚠️ Expenses > $5,000 require VP approval
✗ Access bank account credentials
✓ Create payment requests
✗ Execute payments directly
✓ Read invoices
✓ Update invoice status

Time restrictions:
✓ Reports: 24/7 access
✓ Approvals: Business hours only (9am-6pm)
✗ No access on financial close days

Compliance restrictions:
- Audit log all financial data access
- Require multi-factor auth for payment requests
- Block access to records older than retention policy
- Enforce segregation of duties

Why fine-grained authorization matters:

Instead of binary "all or nothing" access, you create **intelligent guardrails** that:

- Adapt to context and risk level
- Balance usability with security
- Enable automation while maintaining control
- Provide defense in depth
- Meet compliance requirements
- Scale across complex organizations

How All Four Pillars Work Together: Complete Example

Scenario: You ask your AI agent to "Send the Q4 financial report to our investors"

Step 1: User Authentication (Pillar 1)

System prompts: Face ID authentication
You authenticate with biometrics
System verifies:
- ✓ Identity confirmed: john.doe@company.com
- ✓ Role: Finance Director
- ✓ Clearance: Authorized for financial data

Step 2: Token Vault (Pillar 2)

AI agent needs access to:
- Google Drive (read financial reports)
- Gmail (send emails)

Token Vault checks authorization and issues tokens:

Token 1:
Service: Google Drive
Scope: drive.files.read
Resource: /finance/reports/*
Expires: 1 hour

Token 2:
Service: Gmail
Scope: gmail.send
Restrictions: External recipients only
Expires: 1 hour

Step 3: Fine-Grained Authorization (Pillar 4)

System evaluates contextual rules:

Location check:
✓ User in office (IP: 203.0.113.45)

Time check:
✓ Current time: 2:15 PM Tuesday (business hours)

Data classification check:
⚠️ File: "Q4_Financial_Report.pdf"
⚠️ Classification: CONFIDENTIAL - INVESTOR RELATIONS
⚠️ Contains: Revenue data, profit margins, forecasts

Risk assessment:
⚠️ External recipients detected
⚠️ Financial data flagged
⚠️ High-risk action triggered → Requires approval

Step 4: Asynchronous Authorization (Pillar 3)

System pauses execution and sends approval request:

📱 APPROVAL REQUIRED

Action: Send financial report
Document: Q4_Financial_Report.pdf (2.3 MB)
Classification: CONFIDENTIAL

Recipients:
- sarah.investor@capitalfund.com
- michael.partner@venturecap.com

Content summary:
- Q4 revenue figures
- Annual profit margins
- 2025 financial projections

Risk factors:
⚠️ Confidential financial data
⚠️ External distribution

Auto-checks passed:
✓ Recipients on approved investor list
✓ Document approved for release
✓ Board disclosure requirements met

[Approve and Send] [Deny] [Review Document] [Edit Recipients]

Step 5: User Decision

You review the request
You verify:
✓ Correct document version
✓ Approved recipient list
✓ Proper timing (after board approval)

You click: [Approve and Send]

Step 6: Execution and Audit

System executes with full logging:

Action: Send email
Agent: finance-assistant-v2
User: john.doe@company.com
Timestamp: 2024-12-14 14:17:33 UTC
Tokens used:
  • gmail.send (token ID: tk_8x9y2z)
  • drive.files.read (token ID: tk_3a4b5c)
Recipients:
  • sarah.investor@capitalfund.com
  • michael.partner@venturecap.com
Attachments:
  • Q4_Financial_Report.pdf (SHA256: abc123...)
Approval:
  • Required: Yes
  • Approved by: john.doe@company.com
  • Approval time: 2024-12-14 14:17:28 UTC
  • Approval method: Mobile app (Face ID)
Result: SUCCESS
Email ID: msg_xyz789

Audit trail stored for 7 years (compliance requirement)

Final Confirmation

AI Agent responds:
"✓ Done! I've sent the Q4 financial report to both investors.

Email delivered at 2:17 PM
Tracking ID: msg_xyz789

Sent to:
- sarah.investor@capitalfund.com
- michael.partner@venturecap.com"

Result: Email sent securely with:

- Strong user authentication
- Temporary, scoped access tokens
- Context-aware authorization rules

- Human approval for sensitive action
- Complete audit trail
- Full accountability

Summary: Key Takeaways

Core Principles

1. **Identity is foundational** — Without identity, there is no accountability, security, or trust in AI agents.
2. **Least privilege access —** Grant only minimum necessary permissions, nothing more.
3. **Defense in depth —** Multiple security layers working together provide robust protection.
4. **Human oversight for critical actions —** Keep humans in the loop for sensitive decisions.
5. **Audit everything —** Complete logging enables investigation, compliance, and trust.

The Four Pillars in Brief

Pillar	Purpose	Key Benefit
Authentication	Verify user identity	Prevents impersonation
Token Vault	Secure credential storage	Protects secrets
Async Authorization	Just-in-time approvals	Human oversight
Fine-Grained Authorization	Context-aware access control	Intelligent guardrails

Security Best Practices

- Use tokens, never share passwords
- Implement short expiration times
- Apply principle of least privilege
- Require approval for sensitive actions
- Log all agent activities
- Use strong authentication methods
- Apply context-aware access rules
- Regularly review and revoke unused access

Why This Matters

As AI agents become more capable and autonomous, robust identity and authorization frameworks are essential to:

- **Enable trust —** Users must trust agents to act on their behalf
- **Ensure compliance —** Meet regulatory requirements
- **Prevent abuse —** Limit damage from compromised agents
- **Maintain accountability —** Track all actions for investigation
- **Scale safely —** Deploy AI agents across organizations securely

---

Next Steps

This guide covered the foundational concepts. To continue learning:

- **Advanced topics**: Highly Regulated Identity (HRI), sender-constrained tokens
- **Implementation**: MCP (Multi-Channel Protocol) for standardized tool calling
- **Hands-on practice**: Build a secure AI agent with Offzero platform
- **Real-world examples**: Case studies and production deployments

---

Glossary

Access Token — Short-lived credential for immediate API access

Agent — AI system that can take actions on behalf of users

Authentication — Verifying identity (who you are)

Authorization — Determining permissions (what you can do)

Audit Trail — Complete log of all actions for investigation and compliance

Fine-Grained Authorization — Highly specific, context-aware access control

Least Privilege — Granting minimum necessary permissions

LLM — Large Language Model (e.g., GPT, Claude)

OAuth — Industry-standard authorization protocol

Passkey — Passwordless authentication using device biometrics

RAG — Retrieval Augmented Generation (adding real-time data to LLM responses)

Refresh Token — Long-lived credential for obtaining new access tokens

Scope — Specific permission granted by a token

SSO — Single Sign-On (one login for multiple systems)

Token — Temporary credential with limited permissions

Token Vault — Secure storage for sensitive credentials

Tool Calling — AI agent executing functions to interact with external services

---

Document prepared for educational purposes
Based on Global Hack Week session by Fred Patton, Offzero

End of Guide