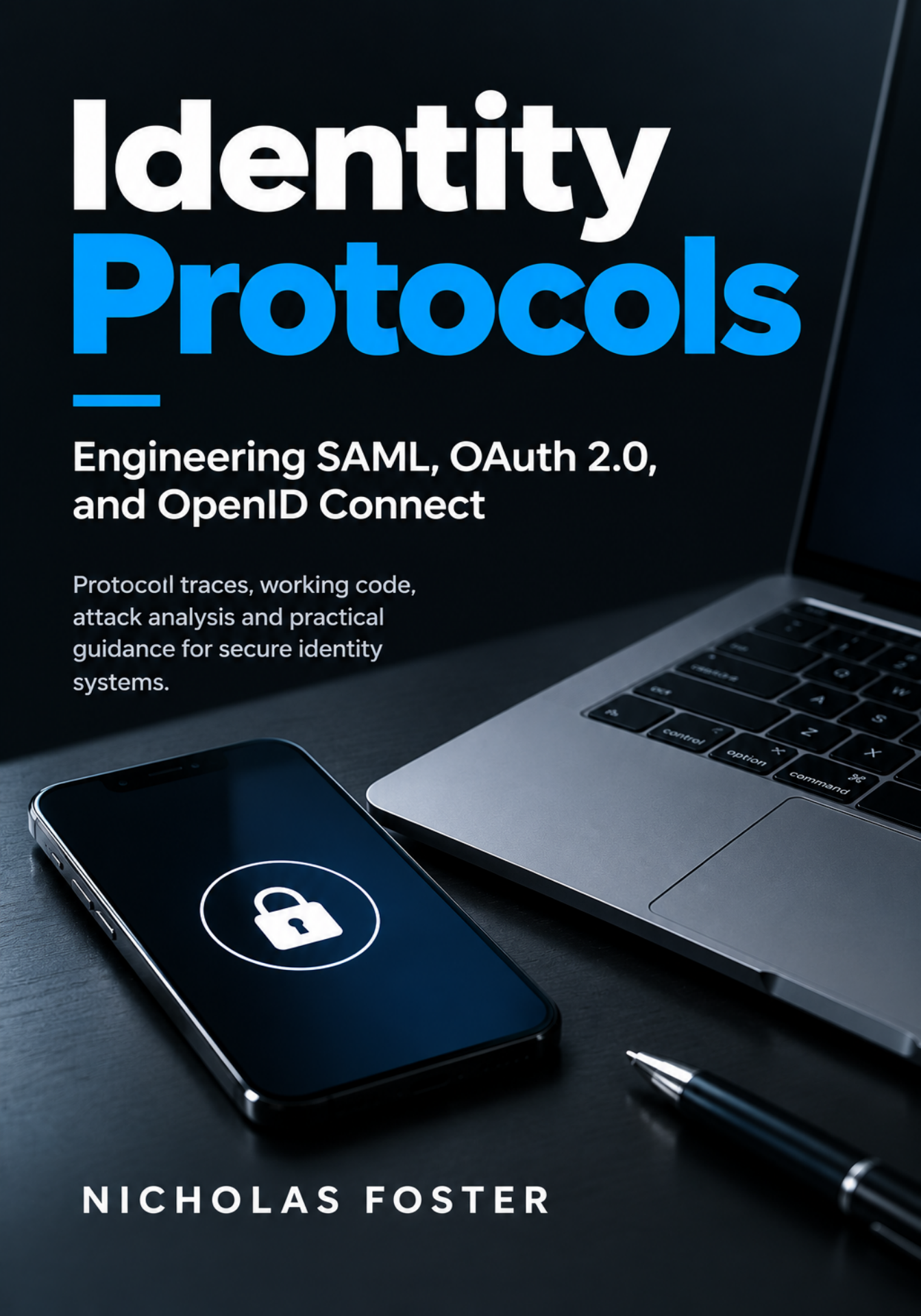
Identity Protocols

Engineering SAML, OAuth 2.0, and OpenID Connect

Protocol traces, working code, attack analysis and practical guidance for secure identity systems.



NICHOLAS FOSTER

Identity Protocols

Engineering SAML, OAuth 2.0, and OpenID Connect

Steve Publications

This book is available at https://leanpub.com/identityprotocols

This version was published on 2026-09-24

Contents

Engineering SAML, OAuth 2.0, and OpenID Connect

About this book

Modern identity protocols are complex distributed trust mechanisms. Their security depends not just on cryptographic correctness but on precise coordination across browsers, servers, and networks. This book takes you from first principles through production operations of SAML 2.0, OAuth 2.0, and OpenID Connect. You will find complete protocol traces showing every HTTP request and response, working code examples you can run and study, rigorous analysis of the most important attacks and vulnerabilities, and architectural guidance for real-world deployments. It is written for experienced engineers who need to design, implement, or audit identity systems.

Introduction

You are building a system that authenticates users, authorizes their actions, and coordinates trust across organizational boundaries. This is not straightforward. The protocols you will use were designed by committees over decades, refined through repeated exploitation, and deployed in environments far more hostile than any textbook model assumes. They must survive the quirks of HTTP, the constraints of browsers, the failures of networks, and the ingenuity of attackers.

Consider what happens when a user clicks "Log in with Google" on your application. Before that user is logged in, dozens of things must happen: your application must construct an HTTP redirect with precise parameters, the browser must follow that redirect while preserving state, Google's servers must verify a cryptographic proof that the request originated from your application, the user must authenticate, Google must issue a signed token, your application must validate that signature using a key it fetched over HTTPS, and only then can you trust that you know who this user is. At any step along the way, a misconfigured redirect URI, a missing state parameter, a confused key lookup, or a leaked token can result in complete authentication bypass.

This book is not about which identity provider to choose. It is about how the protocols themselves work, why they were designed as they were, what security properties they provide, what assumptions they make, and where those assumptions break down. By the time you finish reading it, you will be able to read the relevant specifications and understand them, trace a complete authentication flow through HTTP headers and cookies, spot the difference between a normative requirement and a deployment convention, and design identity systems that are secure by construction rather than hope.

The book is organized into a logical progression. We begin with the foundational concepts of identity, authentication, and authorization, then move through the historical context that led to modern protocols. We then treat SAML 2.0, OAuth 2.0, and OpenID Connect in depth, one protocol family at a time, with complete protocol traces, concrete examples, and implementation code. After the protocol chapters we cover the attack surface systematically, showing how real implementations fail and how to fix them. The final part

of the book deals with architecture and operations: how to design identity at scale, how to manage keys and certificates, how to migrate between protocols, and how to respond when things go wrong.

Every chapter is written with a single assumption: you are a skilled engineer who needs technical precision, not a sales pitch. If a section feels dense, it is because the subject deserves the detail. You will not find hand-waving about "state-of-the-art solutions" or vague promises about security. You will find HTTP requests, XML documents, JSON payloads, JWTs with their actual byte-level structure, and code you can run to see what happens when things are done right and when they are done wrong.

Chapter 1: Identity, Authentication, and Authorization: The Foundations

Before we examine any protocol, we must establish precise definitions. The terms authentication, authorization, identity, and delegation are used interchangeably in casual conversation and marketing copy. In security engineering they are distinct concepts, and confusing them leads to design errors. This chapter sets the terminology and mental models for everything that follows.

Principals, Identities, and Accounts

A principal is any entity that can act. In computer systems this includes human users, automated processes, services, scripts, and even containers or functions. Any system that mediates access must be able to distinguish between principals.

An identity is a name or representation for a principal. When Alice logs into your system, her identity might be "alice@example.com", a UUID, or some other identifier. The crucial point is that an identity is not the same thing as a principal. The identity is what the system calls the principal; the principal is the thing acting in the world. This distinction matters because a single principal may have multiple identities across different systems, and a single identity may sometimes be assumed by the wrong principal.

An account is a system-specific representation of an identity, typically including credentials, settings, and associated data. Your organization's Active Directory might have an account for Alice with her corporate email, password hash, and group memberships. Your SaaS application might have a separate account for Alice with its own user profile and permissions. These are different accounts representing the same underlying principal, using the same or different identities.

In federated systems, we often distinguish between a local identity (one managed directly by the system) and a federated identity (one derived from or asserted by an external authority). When Alice logs in via her corporate Single

Sign-On provider, your application typically does not store her credentials and may not even choose her identity. It accepts an assertion from the identity provider that says "this user is Alice" and maps that to an account in its own database.

Authentication versus Verification versus Identification

Authentication is the process of confirming that a principal is who it claims to be. When Alice proves she is Alice by presenting something she knows (a password), something she has (a cryptographic key or hardware token), or something she is (a biometric characteristic), the system authenticates her.

Authentication is not identification. Identification is the act of asserting an identity. When Alice types her email address into a login form, she is identifying herself. The system does not yet know whether she is actually Alice. When the system then confirms her password matches the one on record, it has authenticated her claim.

Verification is a broader term. We verify signatures to confirm that a message was created by the holder of a private key. We verify tokens to confirm they were issued by a trusted authority and have not been tampered with. We verify certificates to confirm they were issued by a trusted certificate authority. All of these are forms of authentication at the cryptographic level.

The distinction between identification and authentication explains why login flows have two phases. First, the user provides an identifier. Second, the system challenges the user to prove ownership of that identifier. In modern protocols, these phases are often combined into a single redirect, but conceptually they remain distinct.

Authentication is always relative to a context. Authenticating to a password database is not the same thing as authenticating to a hardware token or authenticating via a federated assertion. Different methods provide different levels of assurance, which we will discuss in detail when we cover authentication assertions and security levels.

Authorization and the Confused Deputy Problem

Authorization is the process of determining whether an authenticated principal is permitted to perform a specific action on a specific resource. If Alice is

authenticated, authorization determines whether she can read a document, delete a record, or call an API endpoint.

Authorization is often confused with authentication because they occur together in practice. You typically need to know who someone is before you can decide what they are allowed to do. But they are conceptually independent. A system can authorize actions without authenticating identities (public APIs with anonymous access) and can authenticate identities without making authorization decisions (a login portal that only issues tokens).

Authorization decisions depend on three things: who is asking, what they are asking for, and under what conditions. A simple access control list answers the first two questions: user A can read resource B. More sophisticated authorization systems also consider context: is the request coming from an internal network? What time is it? Has the user performed multi-factor authentication? What other actions has the user taken recently?

The confused deputy problem is central to authorization design and appears repeatedly in identity protocols. It occurs when a principal (the deputy) with access to a resource is tricked into using that access on behalf of another principal (the confuser) in a way that would not be allowed if the other principal made the request directly.

Consider a classic example. Alice has read access to a file that Bob does not. An application runs as Alice and provides a service that reads files on her behalf. If Bob can tell that application to read a file and return the contents, he has successfully exploited the confused deputy. The application is acting as Alice when reading the file, but Bob is the one who benefits from the action.

In the context of OAuth 2.0, the confused deputy problem manifests when an access token is issued that allows a client to act on behalf of a user, but the resource server does not verify that the token was intended for that specific client. If Client A can use a token intended for Client B, it has confused the resource server about who should be allowed to access the resource. Token binding mechanisms, audience claims, and proper scope validation are all designed to prevent this confusion.

Modern authorization architectures sometimes separate authorization into its own service. The authorization policy is evaluated by a central engine rather than embedded in each resource server. This pattern, often called policy-based or attribute-based access control, is increasingly common in large-scale systems.

Sessions and Their Invariants

A session is a logical association between a principal and a system that persists across multiple requests. When Alice logs in, the system creates a session that tracks her authentication state so she does not need to re-authenticate on every request. Sessions introduce state into an otherwise stateless protocol like HTTP, and that state is the source of many security vulnerabilities.

Sessions have several invariants that must be maintained for them to be secure:

- **Uniqueness:** Each session must be uniquely identifiable. Two different principals should never share a session. This is typically enforced by generating a high-entropy session identifier.
- **Binding:** A session must be bound to the principal who established it. If an attacker obtains a session identifier, they should not be able to use it unless they also have access to the principal's communication channel or device. This is why session cookies are scoped to specific domains and why secure session management uses HttpOnly and Secure attributes.
- **Expiry:** Sessions must have a finite lifetime. Perpetual sessions increase the window of opportunity for session hijacking and make credential rotation ineffective.
- **Revocability:** A session must be terminatable before its natural expiry. This is essential for logout, for revoking access after credential changes, and for responding to compromise.
- **Integrity:** Session data must not be tamperable by the client. If the client controls the session state, they can potentially escalate privileges by modifying their own role or permissions.

Sessions can be implemented in several ways. Server-side sessions store the session state in the server's memory or database and send the client only an identifier. Client-side sessions embed all the necessary state in a token sent by the client, typically signed or encrypted so the server can verify its authenticity without storing state. JWTs used for session management are client-side sessions.

The choice between server-side and client-side sessions involves trade-offs. Server-side sessions provide better revocation control and allow the server to modify session attributes without client involvement. Client-side

sessions are easier to scale horizontally and work naturally in distributed environments, but revocation requires additional mechanisms and tokens tend to grow larger.

The Trust Boundary and What You Can Rely On

The trust boundary is the line between what you can trust and what you cannot. In any system, you must be explicit about which components are trustworthy and which are not.

For web-based identity systems, the trust boundary typically lies at the backend server. The browser is not trusted. The network is not trusted. The user is not fully trusted. Data transmitted over HTTP can be observed and potentially modified. JavaScript executed in the browser can be manipulated. User input can be malicious.

This has direct implications for identity protocol design. Consider token storage. If you store an access token in browser memory or local storage, any JavaScript running on that page can read it. This includes the JavaScript your own application ships, which may contain bugs or may be compromised through a supply chain attack. It also includes any malicious JavaScript injected through a cross-site scripting vulnerability. The browser itself is a hostile environment for secrets.

Consider redirect URIs. An attacker who controls a domain can set up a page that looks like your application's login callback, capture the authorization code or token from the URL parameters, and forward the user to the real application to avoid detection. This is why OAuth 2.0 requires redirect URIs to be pre-registered and validated.

Consider the identity provider's role. When you federate authentication to an external provider, you are trusting that provider to correctly authenticate users and issue accurate assertions. If the provider is compromised, misconfigured, or simply buggy, your system will accept invalid authentication. Federation extends your trust boundary to encompass systems you do not control.

Every component in an identity system has assumptions about what it can trust. SAML assumes XML signatures are validated correctly. OAuth 2.0 assumes the authorization server authenticates clients properly. OpenID Connect assumes ID token signatures are verified and claims are validated. When

any of these assumptions are violated, the security of the entire system is at risk. Understanding these assumptions is more important than understanding the protocol flows themselves, because the attacks almost always target the assumptions rather than the protocols.

From Single Systems to Federation

Early computer systems authenticated users directly against a local directory. Each system maintained its own user database and credential store. This approach is simple and self-contained, but it does not scale. Users must remember different passwords for different systems. Administrators must manage user lifecycles across multiple databases. And when systems need to share identity information, they must build custom integrations.

Federated identity solves these problems by allowing one system to accept authentication evidence from another system it trusts. Instead of managing credentials directly, your application relies on an identity provider to authenticate users and then accepts signed assertions from that provider.

Federation introduces complexity in exchange for centralized management. You must establish trust relationships, configure metadata, manage certificates, and handle failures when the identity provider is unavailable. But the benefits are substantial: single sign-on for users, centralized user lifecycle management, consistent authentication policies, and the ability to share identity data across organizational boundaries.

The protocols we will study (SAML 2.0, OAuth 2.0, and OpenID Connect) are all federation protocols. They define the rules for how trust relationships are established, how authentication evidence is packaged and transmitted, how tokens are issued and validated, and how sessions are coordinated across multiple systems.

Understanding federation requires understanding that trust is transitive but not automatic. When you trust an identity provider to authenticate users, you are implicitly trusting everything in the chain between the user and your system: the browser, the network, the identity provider's authentication mechanisms, the protocol implementation, and the certificate validation process. A failure at any point in that chain breaks the security guarantee.

In the chapters that follow, we will examine each of these protocols in detail, showing you exactly how they establish trust, how they protect against

common attacks, and where their security depends on correct implementation. But the foundational concept is always the same: federation is about extending trust across boundaries, and every boundary is a potential attack surface.

Chapter 2: The Web's Authentication Heritage

Before modern federation protocols existed, web applications authenticated users using simple mechanisms built into HTTP or implemented as custom solutions. Understanding these mechanisms is essential because modern protocols were designed to solve problems that emerged from their limitations. They also remain in use today, often in combination with SAML, OAuth, and OIDC, and their security properties affect everything built on top of them.

HTTP Basic and Digest Authentication

HTTP Basic Authentication, defined in RFC 7617, is one of the simplest authentication mechanisms on the web. The client sends credentials with every request using the Authorization header:

```
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
```

The value after "Basic" is the username and password concatenated with a colon, then Base64-encoded. Base64 is an encoding, not encryption. Anyone who can see the request can decode the credentials in seconds. Basic authentication is only secure when used over TLS, and even then it has limitations.

The server responds with a 401 Unauthorized status and a WWW-Authenticate header to challenge the client:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="Restricted Area"
```

The browser typically displays a modal dialog prompting for credentials. This behavior is built into browsers and cannot be customized, which is why Basic auth is rarely used for user-facing applications.

Despite its simplicity, Basic authentication has legitimate use cases. It is commonly used for machine-to-machine communication where credentials are stored securely in configuration files and every request requires authentication. API documentation tools like Swagger and Postman use Basic auth for API key validation. Some internal services and proxies also rely on it.

HTTP Digest Authentication, defined in RFC 7616, was designed to avoid sending the password in every request. Instead of transmitting the password, the client sends a cryptographic hash of the password combined with a server-provided nonce:

```
Authorization: Digest username="alice", realm="Restricted",
  nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093", uri="/api",
  response="6629fae49393a05397450978507c4ef1",
  ↪  opaque="5ccc069c403ebaf9f0171e9517f40e41",
  qop=auth, nc=00000001, cnonce="0a4f113b"
```

The nonce prevents replay attacks. The qop (quality of protection) parameter specifies that this is authentication quality, and nc tracks the number of requests to detect replays. Digest authentication is more complex than Basic and provides better security for password transmission, but it is rarely implemented today. Most applications that need stronger authentication than Basic use TLS and move to more sophisticated protocols.

Both Basic and Digest authentication have a fundamental limitation: they authenticate every request independently. The server does not maintain any session state. This is efficient and simple, but it means credentials must be sent with every request, and there is no mechanism for single sign-on across multiple services.

Browser Cookies and Session Management

HTTP cookies, introduced by Netscape in 1994, provided a mechanism for maintaining state across requests. A server sets a cookie using the Set-Cookie header:

```
Set-Cookie: session_id=abc123; Path=/; HttpOnly; Secure; SameSite=Lax
```

The browser stores the cookie and automatically includes it in subsequent requests to the same domain using the Cookie header:

```
Cookie: session_id=abc123
```

Cookies became the foundation for web session management. When a user logs in, the server creates a session, stores the session data server-side, and sends a session identifier to the browser in a cookie. Subsequent requests include the cookie, and the server uses the session identifier to look up the user's authentication state.

Cookie attributes control their security properties:

- **Path:** Restricts the cookie to a specific URL path within the domain. A cookie with Path=/admin is only sent for requests to /admin and its subpaths.
- **Domain:** Specifies the domain for which the cookie is valid. If omitted, the cookie is scoped to the exact host that set it. Setting Domain=.example.com makes the cookie valid for all subdomains.
- **Secure:** Ensures the cookie is only sent over HTTPS connections. Without this flag, the cookie is sent over both HTTP and HTTPS, exposing it to network eavesdropping.
- **HttpOnly:** Prevents JavaScript from accessing the cookie via document.cookie. This mitigates cross-site scripting attacks that try to steal session cookies.
- **SameSite:** Controls whether the cookie is sent with cross-site requests. SameSite=Strict prevents the cookie from being sent on any cross-site request. SameSite=Lax allows it on top-level navigation from external sites but not on cross-site POST requests or subresource requests. SameSite=None allows the cookie on all requests, but requires the Secure flag.
- **Expires / Max-Age:** Sets the cookie's lifetime. Without these attributes, the cookie is a session cookie and expires when the browser is closed.

Modern browser security has evolved significantly. The SameSite attribute was introduced in response to cross-site request forgery attacks. Secure by default policies are being adopted. Partitioned cookies and other mechanisms are being developed to limit cross-site tracking while preserving legitimate functionality. These browser-level security mechanisms directly affect how identity protocols operate in browsers.

Cookies are not a panacea. They can be stolen through cross-site scripting, phishing, or network attacks on unencrypted connections. They are limited to

approximately 4KB of data per cookie. They are automatically sent with every request to the domain, which can expose them to logging and profiling. And they only work within a single domain or set of related domains, which limits their usefulness for cross-domain federation.

Same-Origin Policy and Cross-Origin Constraints

The same-origin policy is a browser security model that restricts how resources from different origins can interact. An origin is defined by the combination of protocol, host, and port. Two pages have the same origin only if all three components match.

Under the same-origin policy, a script loaded by one origin cannot access the DOM, cookies, or local storage of a page from a different origin. This prevents a malicious site from reading sensitive data from a legitimate site where the user is logged in. For example, if you are logged into your bank at bank.example.com, a script running on attacker.example.com cannot read your bank's page content or cookies.

The same-origin policy has important implications for identity protocols. When a browser navigates to an identity provider's login page, that page is a different origin from the application requesting authentication. The identity provider cannot read the application's cookies, and the application cannot read the identity provider's cookies. This isolation is both a security feature and an implementation constraint.

Cross-Origin Resource Sharing (CORS) provides a controlled way for servers to relax the same-origin policy. A server can include Access-Control-Allow-Origin headers to permit specific origins to make cross-origin requests:

```
Access-Control-Allow-Origin: https://app.example.com
Access-Control-Allow-Credentials: true
```

When Access-Control-Allow-Credentials is set to true, the browser will include credentials (cookies, authorization headers) in the cross-origin request. This allows a single-page application hosted at app.example.com to make authenticated requests to an API at api.example.com.

CORS configuration is a frequent source of security issues. Overly permissive CORS headers can expose APIs to unauthorized access. The wildcard

origin * cannot be used with credentials, which sometimes leads developers to implement custom origin validation that introduces vulnerabilities. And CORS only applies to browser-initiated requests; server-to-server communication is not constrained by it.

The same-origin policy also affects how tokens are used in browser environments. A token obtained from one origin cannot be directly read by scripts from another origin. This is why single-page applications must receive tokens through redirects, postMessage, or CORS-enabled API responses rather than by directly accessing the identity provider's storage.

Forms-Based Authentication and Its Flaws

Most web applications use forms-based authentication rather than HTTP Basic or Digest. The user enters their credentials into an HTML form and submits it to the server, which validates the credentials and creates a session.

A basic login form looks like this:

```
<form method="POST" action="/login">
  <input type="text" name="username" required>
  <input type="password" name="password" required>
  <button type="submit">Log in</button>
</form>
```

The form submits credentials via HTTP POST, the server validates them, and on success sets a session cookie:

```
HTTP/1.1 302 Found
Location: /dashboard
Set-Cookie: session_id=xyz789; Path=/; HttpOnly; Secure
```

The browser follows the redirect and the user is logged in.

Forms-based authentication is flexible. You can customize the login page, add multi-factor authentication, integrate with external directories, and implement custom security policies. But it has several inherent problems:

Credential management. Each application maintains its own user database and password hashes. Users must create and remember different credentials

for each application. Password resets require dedicated infrastructure. And weak password policies lead to credential reuse across sites, amplifying the impact of any single breach.

No single sign-on. If a user has accounts on five different applications within the same organization, they must log in to each one separately. Each application manages its own sessions independently, so logging out of one does not log out the others.

Limited federation. Integrating with external identity providers requires custom code. If the organization wants to support SSO through Active Directory or a third-party provider, each application must be modified to support the integration.

Security inconsistency. Different applications implement authentication differently. Some use strong password hashing, some use weak or deprecated algorithms. Some implement rate limiting and account lockout, some do not. The overall security posture depends on the weakest implementation.

These problems motivated the development of centralized authentication and federation protocols. Single sign-on allows users to authenticate once and access multiple applications. Federation allows applications to delegate authentication to a trusted provider rather than managing credentials themselves.

The Need for Single Sign-On

Single sign-on is the ability to authenticate once and access multiple services without re-authenticating. The benefits are substantial. Users only need to remember one set of credentials and log in once per session. Organizations can centralize authentication policies, password management, and multi-factor authentication. Applications can focus on their core functionality rather than implementing authentication infrastructure.

SSO introduces complexity, however. Applications must trust the identity provider to correctly authenticate users. Sessions must be coordinated across multiple services. Logging out of one service may need to log out the user from others. And if the identity provider is compromised, all dependent services are affected.

SSO also requires a protocol for communicating authentication state between the identity provider and the service providers. The identity provider

must be able to tell the service provider that a user has been authenticated. The service provider must be able to verify that the authentication claim is genuine and has not been forged. And the communication must survive the constraints of web browsers, which navigate between origins and do not allow arbitrary cross-origin data exchange.

This is exactly the problem that SAML 2.0, OAuth 2.0, and OpenID Connect solve. They define the messages, flows, and security mechanisms that allow authentication state to be communicated across domains in a way that is verifiable and tamper-resistant.

Early Enterprise SSO Attempts (Kerberos, LDAP, RADIUS)

Before web-based federation protocols became mainstream, enterprises used other protocols for centralized authentication and access control. Understanding these protocols helps explain why modern identity protocols are designed as they are and why they sometimes need to integrate with legacy infrastructure.

Kerberos is a network authentication protocol designed by MIT in the 1980s. It uses symmetric-key cryptography and a trusted Key Distribution Center (KDC) to issue tickets that prove a principal's identity to services. Kerberos is widely deployed in Windows Active Directory environments and provides strong mutual authentication. However, it is complex to deploy, requires precise time synchronization, and was designed for internal networks rather than the open internet. Its ticket-granting mechanism influenced later token-based protocols, including concepts found in OAuth and OIDC.

LDAP (Lightweight Directory Access Protocol) is not an authentication protocol but a directory access protocol. Organizations use LDAP directories to store user accounts, groups, and attributes. Applications authenticate users by querying the LDAP directory and verifying credentials. LDAP is the backbone of many enterprise authentication systems. Modern identity providers often use LDAP as a backend for user data while presenting SAML or OIDC interfaces to applications.

RADIUS (Remote Authentication Dial-In User Service) is an authentication protocol primarily used for network access control. When a user connects to a wireless network or VPN, the network access device queries a RADIUS server to

authenticate the user. RADIUS is simple and widely deployed but was designed for a specific use case and does not support the rich claims and federation capabilities needed for modern web applications.

These legacy protocols remain deeply embedded in enterprise infrastructure. Modern identity platforms often bridge between them, presenting LDAP-backed user databases through SAML or OIDC endpoints, or using Kerberos tickets as a form of credential for OAuth authentication. When designing identity systems, you must account for the protocols that already exist in your environment and find ways to integrate them securely.

The evolution from these early systems to modern federation protocols represents a shift from infrastructure-centric to user-centric design. Kerberos, LDAP, and RADIUS were designed for controlled network environments with trusted infrastructure. SAML, OAuth, and OIDC were designed for the open internet, where browsers navigate between untrusted origins, users interact with third-party applications, and security must be maintained without controlling the underlying network.

Chapter 3: Federation and the Trust Model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

What Federation Means and Why It Exists

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Identity Providers and Service Providers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Centralized versus Distributed Trust

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Assertions, Tokens, and the Statement Problem

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Metadata and Dynamic Trust

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Role of Humans: Consent and Delegation

Chapter 4: HTTP, the Browser, and the Attack Surface

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The HTTP Redirect and State Transfer

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Cookie Scoping, SameSite, Secure, and HttpOnly

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Cross-Origin Requests and CORS

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Post Messages and Client-Side Communication

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Browser as a Hostile Environment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Logging, HSTS, and Transport Security

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 5: SAML 2.0: Architecture and Concepts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SAML's Origins in Shibboleth and Liberty Alliance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The SAML Vocabulary: Principals, Assertions, and Artifacts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Identity Providers and Service Providers in SAML

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Bindings: HTTP-Redirect, HTTP-POST, and HTTP-Artifact

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Security Requirements: Signatures, Encryption, and Timestamps

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The SAML Processing Model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 6: SAML 2.0: Protocol Flows and Message Formats

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authentication Request (AuthnRequest)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authentication Response and Asserted Claims

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Complete Protocol Trace: SP-Initiated SSO

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Complete Protocol Trace: IdP-Initiated SSO

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Single Logout (SLO) Flow

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Artifact Binding and Resolution

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 7: XML Security for SAML

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

XML Signature (XMLDSig): Enveloped, Enveloping, Detached

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Canonicalization Algorithms and Their Pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

XML Encryption (XMLenc)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Certificate Formats and Key Transport

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Signing Assertions versus Enveloping Assertions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Key Discovery from Metadata

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 8: SAML Metadata and Trust Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The EntityDescriptor and Its Children

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Endpoints and Bindings in Metadata

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Certificate Publication and Key Roles

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Entity Categories and Attribute Authority Metadata

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Metadata Aggregation and Chaining

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Operational Lifecycle: Updates, Revocation, and Rotation

Chapter 9: OAuth 2.0: Delegated Authorization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authorization Problem and Delegation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

OAuth's Four Roles

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Tokens: Access Tokens, Refresh Tokens, and Their Invariants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Grant Types and What They Represent

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Scopes, Permissions, and Resource Boundaries

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 10: OAuth 2.0: Grant Types and Protocol Flows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authorization Code Grant

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Implicit Grant (and Why It Is Deprecated)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Client Credentials Grant (Machine-to-Machine)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Resource Owner Password Credentials Grant (and Why It Is Deprecated)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Authorization Code Grant with PKCE

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Endpoint Authentication Methods

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 11: OAuth 2.0 Security Requirements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

TLS Requirements and Certificate Validation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Client Authentication at the Token Endpoint

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Redirect URI Validation and Open Redirect Prevention

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The State Parameter and CSRF Prevention

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Endpoint Security

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Secure Token Handling and Storage

Chapter 12: OpenID Connect: Identity Layer on OAuth

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

What OpenID Connect Adds Over OAuth 2.0

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The ID Token: Structure, Claims, and Validation Rules

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Authorization Code Flow with OpenID Connect

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The userinfo Endpoint and Claim Sets

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

RP-Initiated Logout

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Pushed Authorization Requests (PAR)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 13: JWT, JWS, JWE, and JWK: The Token Formats

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWT Syntax and Claims: Header, Payload, Signature

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWS: Signing Algorithms (RS256, ES256, PS256, etc.)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWE: Encryption and Key Wrapping

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWK and JWKS: Key Representation and Discovery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Algorithm Confusion and the none Attack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Validation: Issuer, Audience, Expiration, and Signature

Chapter 14: OpenID Connect Discovery and Registration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authorization Server Metadata Document

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Client Metadata Registration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Well-Known Discovery URLs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Dynamic Client Registration Protocol

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Front-Channel versus Back-Channel Discovery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Discovering Supported Algorithms and Constraints

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 15: PKCE: Securing Public Clients

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Public Client Problem

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Authorization Code Interception Attack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

PKCE: Protocol Mechanics and Code Challenges

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

PKCE with Authorization Code Flow

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

PKCE with OIDC

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Why PKCE Belongs Everywhere

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 16: Implementing OAuth and OpenID Connect: Authorization Server

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Architecture: Endpoints, Storage, and Dependencies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Client Registry and Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Authorization Endpoint

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Token Endpoint

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWKS and Discovery Endpoints

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Running the Server: Setup and Test

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 17: Implementing Relying Parties and Clients

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Web App with Authorization Code + PKCE (Backend Session)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Single-Page Application with PKCE

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Machine-to-Machine with Client Credentials

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Validation Middleware

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Error Handling and Logging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 18: Implementing SAML Service Providers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SP Metadata Generation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Authentication Request Construction

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Assertion Validation and Signature Verification

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SAML Response Processing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Single Logout Handling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Complete SP Implementation Walkthrough

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 19: Advanced OAuth Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Introspection Endpoint

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Revocation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

OAuth Token Exchange (RFC 8693)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWT Secured Authorization Response Mode (JAR/JARM)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Device Authorization Grant (for TVs and IoT)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Mutual TLS Client Authentication

Chapter 20: OAuth and OIDC for APIs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Reference Tokens versus Self-Contained Tokens

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

The Resource Server as an OAuth Participant

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Validating Tokens in API Gateway and Service Mesh

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Audience Claims and Multi-API Authorization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Fine-Grained Authorization with Scopes and Claims

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

API Security: Rate Limiting, CORS, and Transport

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 21: Sessions, Cookies, and Browser Security

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Session State Models

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Cookie Attributes and Security

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

CSRF Defense Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Session Fixation and Session Hijacking

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Secure Logout and Session Termination

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SameSite: How It Helps and Its Limitations

Chapter 22: Authentication Attack Surface: OAuth and OIDC

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

CSRF and State Parameter Bypass

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Login CSRF (Authorization Server CSRF)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Client Authentication Confusion Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Redirect URI Manipulation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

ID Token Substitution Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Scope Manipulation and Privilege Escalation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Token Leakage Through URLs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

PKCE Downgrade Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 23: SAML Attack Surface

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

XML External Entity (XXE) Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

XML Signature Wrapping Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Alias and Key Confusion Attacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Assertion Replay and Forgery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

NameID and Attribute Manipulation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Insecure XML Parsing

Chapter 24: Identity Architecture Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Backend-for-Frontend (BFF) Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Microservices and Internal Token Propagation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Multi-Tenant Identity and Isolation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Workload Identity for Kubernetes and Cloud

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Enterprise Federation Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SaaS Identity Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 25: Migration: From SAML to Modern Protocols

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Assessing the Current Federation Landscape

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Protocol Bridging and Translation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

User Migration and Credential Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Parallel Operation During Transition

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Breaking Changes and Compatibility

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Post-Migration Security Hardening

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 26: Operational Identity Engineering

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Key and Certificate Lifecycle Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Secrets Management and Rotation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Logging and Auditing Without Leaking Credentials

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Observability and Debugging Identity Flows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Incident Response for Key Compromise

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Availability and Disaster Recovery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Chapter 27: Design Choices and Trade-Offs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

SAML versus OIDC: When Each Makes Sense

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

JWT versus Reference Tokens

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Centralized Identity versus Federated versus Hybrid

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Symmetric versus Asymmetric Token Validation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Short-Lived versus Long-Lived Tokens

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

Vendor Lock-In versus Open Standards

Conclusion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.

References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/identityprotocols.