Mastering Identus: A Developer Handbook

Jon Bauer, Roberto Carvajal

2024-04-05

Table of contents

Table of contents

Welcome

Welcome to the book!

Copyright

License

Book Content:

Applications:

Dedication

Preface

Explains who the authors are and why we were motivated to write this book.

Section I

Introduction

This is a developer-centric book about creating and launching Self-Sovereign applications with Identus. We aim to show the reader how to configure, build and deploy a complex idea from scratch.

This is not a book about Self-Sovereign Identity. There are already great resources available on that topic. If you're new to the idea of SSI or Identus, we reccomend the exellent resources listed below as a pre-requisite to this text.

- Self-Sovereign Identity by Alex Preukschat, Drummond Reed, et al. This is the definitive book on SSI and it's ecosystem of topics.
- Atala PRISM Documentation

It should be noted that Identus is still new and at the time of writing this book, there are very few best practices, in fact, very little practices at all. A handfull of advanturous developers have been building on the platform and sharing their experiences, and we hope to share our own learnings in hopes of magnifying that knowledge and helping developers skip the common pitfalls and bring their ideas to market.

We hope this text will be accessible to anyone who's curious about what's

We hope that newcomers will be able to use this text to skip common pitfalls and misunderstandings, and bring their ideas to market faster.

We're glad you could join us :)

SSI Basics

[SSI Roles/Conceptual Diagram (Triangle of Trust)]

Issuer:

The entity that issues a Verifiable Credential to a Holder. This could be a bank, government agency or anyone that accepts responsibility for making credentials. For example a governmental agency can issue a passport to a citizen, or a gym can issue a membership to a member. The type of agency is not important, only that they issue a credential. This role accepts responsibility for having issued that credential and should look to establish trust and reputation with Holders and Verifiers. The Issuer's DID will be listed in the Issuer (iss:) field of a Verifiable Credential and can be inspected by anyone wishing to know the origin of the credential.

Holder:

A Holder is simply any person or wallet that holds a Verifiable Credential. Verifiable Credentials can represent anything, a gym membership, an accomplishment like a University degree, or a permission set, such as a login or authentication role.

Verifier:

A Verifier is any person or wallet that performs a verification on a Verifiable Credential or any of it's referenced entities. A Verifier might perform a check on cryptographic elements of a VC, or make sure that the Issuer DID belongs to the expected entity. Verifiers usually only care to double check that the Verifiable Credential is legitimate and that the included

claims meet their expectations, for example when a bartender checks the age of a patron before serving them alcohol.

Trust Registry:

When Verifiers need to know who a DID belongs to, there needs to be a way to look up that information. A Trust Registry is a mapping between DIDs and the entities they represent. You can think of this like a phone book for DIDs. Trust Registries need to be trusted themselves, and can be as broad or as specific as they need to be. For example, an Real Estate specific Trust Registry that lists real estate agents can be a way to validate that a particular agent is an accepted member of that industry.

Identus Concepts

[Identus Application Architecture Diagram]

Identus is made up of several open source components. Each could be used or forked separately but they are designed to work well together.

PRISM Node:

PRISM Node implements the `did:prism` methods and is an interface to multiple VDR (Verifiable Data Registries). The node can resolve PRISM DIDs and write transactions to a blockchain or database. PRISM Node is expected to be online at all times.

Cloud Agent:

Written in Scala, the Cloud Agent runs on a server and communicates with clients and peers via a REST API. It is a critical component of an Identus application, able to manage identity wallets and their associated operations, as well as issue Verifiable Credentials. The Cloud Agent is expected to be online at all times.

Edge Agent:

Edge Agents give agent capabilities to clients like Websites and mobile apps. They can can never be assumed to be online at any given time, and therefore rely on sending and receiving all communications through an online proxy, the Mediator.

DIDComm:

DIDComm is a private, secure, and interoperable protocol for communication between decentralized identities. Identus supports DIDCommV2 and allows peers to pass messages between each other, proxied by the Mediator. Messages contain a `to:` and `from:` DID

Mediator:

Mediators act as middlemen between Peer DIDs. In order for any agent to send a message to any other agent, it must know the `to` and `from` DIDs of each message. The sender and recipient together make up a cryptographic connection called a `DIDPair`. Mediators maintain queues of messages for each `DIDPair`. If an Edge Agent is offline, the Mediator will hold incoming messages for them until the agent is back online and able to receive them. Mediators can deliver messages when polled, or push via Websockets. Mediators are expected to be online at all times and be highly available.

Since instantiation of Identus Edge Agents requires a Mediator, there are several publicly available Mediator services which make development simple.

- PRISM Mediator
- RootsID Mediator
- Blocktrust Mediator

While extremely helpful during development, these are not recommended for production Identus deployments as they have no uptime guarantee and will not scale past a small number of concurrent users. We will discuss how to run your own Mediator in Chapter

Building Blocks:

Identus separates the handling of important SSI operations into separate, focused libraries.

Apollo: Apollo is a cryptographic primitives toolbox, which Identus uses to ensure data integrity, authenticity, and confidentiality.

Castor: Castor enables creation, management, and resolution of DIDs.

Pollux: Pollux handles all Verifiable Credential operations.

Mercury: Mercury is an interface to the DIDCommV2 protocol, allowing the sending and receiving of messages between DIDs.

More info on each of the Building Blocks can be found in the Docs

Section II - Getting Started

Installation - Development Environment

Detailed instructions on how to install a development instance of Identus on your local machine.

This will include several tips and tricks to creating a fully functional environment

Section III - Building

Project Overview

This is project overview of the example app, which can issue and verify educational credentials.

Wallets

A deep dive on what an Identus Identity wallet is and represents. We will discuss seed phrases and how they work.

We will talk about creating and recovering Identus wallets.

Tips for using:

- Pluto Encrypted
- Keycloak for OAuth / OpenIDConnect

DIDs

We'll do a deep dive on DIDs and did:prism specifically

- did:prism methods
- canonical vs longform DID

DID Documents

DIDDocuments:

- Resolvers
- Controllers

Connections

We will discuss the concept of Connections, PeerDIDs, and what it means to connect with another Peer, what's happening under the hood and different ways to handle the process in your application.

Also discussed:

- Managing Invites
  - OOB
  - Manual Accept
  - Auto Accept
  - DIDLess Connections (Atala Roadmap)

Verifiable Credentials

Verifiable Credentials

- Overview
- Formats (just mention types, details later in Issuing)
- Schemas
- Publishing your Schema
- Issuing
  - JWT
  - SD-JWT (Atala Roadmap Q2)
  - AnonCreds
- Updating (re-binding)
- Revoking

DIDComm

This chapter will discuss the many ways the DIDComm protocol can be used inside your Identus Application.

- Overview of the DIDComm protocol
- Sending Message
- Sending Files

Verification

Verification

- Presenting Proof
- Presentation Policies / Verification Policies
- Selective Disclosure with AnonCreds

Plugins

This is a placedholer for the topic of Plugins. Identus plugins have not been officially announced yet but are on the IOG roadmap for 2024, Q4. If there is any information by the time we publish we will write about them.

Section IV - Deploy

Installation - Production Environment

In this chapter we will discuss how to prepare a production envioronment for an Identus application.

We will discuss:

- Hardware recommendations
- Production configuration and security
- Lock down Docker / Postgres (default password hole, etc)
- SSL
- Multi Tenancy
- Keycloak
- Testnet / Preprod env
- Connecting to Mainnet
- Set up Cardano Wallet
- Key Management with HashiCorp
- Connecting to Cardano
- Running dbSync

Mediators

Here we will discuss how to set up your own Mediator

We will teach the reader how to

- Install, configure, and Run your own Mediator
- How to manage Websockets
- Performance Tuning

Maintenance

Mastering Identus means maintaing your application once it's launched.

In this chapter we will cover:

- Observability
  - Manage nodes / Memory
  - Performance Testing
  - Analytics with BlockTrust Analytics
- Upgrading Agents
  - How to minimize downtime
- Hashicorp
  - Key Management
  - Key rotation

Section V - Addendum

Trust Registries

Overview of the concept of a Trust Registry

- What role they play in SSI
- Trust Over IP Trust Registry Spec
- Highlight any real world examples if they exist by book completion

Continuing on Your Journey

Information about becoming a Contributor to Identus

- How to contribute to the source code or documentation

Information about how to get involved in the SSI community outside of Identus

- Trust over IP (ToIP)
- Decentralized Identity Foundation (DIF)

Appendices

Errata

Errata goes here

We will list any bugs that may have been “printed” in certain editions of the book.

Glossary

Glossary or Index here

This will reference key concepts and Identus terms and where they are mentioned in the book, allowing someone to look up a term and know what section it is referenced in.

vdr

Verifiable Data Registry (VDR)