Digital Asset

Digital Asset Security Posture

Secure Products, Secure Data

Digital Asset

July 2023

Executive Summary

Digital Asset understands and appreciates the importance of security to our clients, which is reflected in how we architect, design, develop, build and distribute our products, and in how we protect our staff, our locations, our and client's confidential data, our services, and our infrastructure. This position paper describes how we view security and the implementation of controls to ensure this is enforced.

Table of Contents

1 Digital Asset Security Initiatives and Programs
- 1.1 The Digital Asset Information Security Program
- 1.2 Security Program and Governance
- 1.3 Digital Asset Security Team
- 1.4 Digital Asset Security Attestations and Certifications
  - 1.4.1 ISO 27001
  - 1.4.2 SOC 2
  - 1.4.3 Privacy, GDPR and UK PECR
  - 1.4.4 Cloud Security Alliance | Star Registry Participant
  - 1.4.5 Responsible Disclosure of security vulnerabilities

2 Security of the Company
- 2.1 Risk Management and Governance
  - 2.1.1 Risk Management Policies and Procedures
- 2.2 Data Governance and Oversight
- 2.3 Supply Chain Risks and Vendor Management
- 2.4 Email & Web Security
- 2.5 Endpoint Security
- 2.6 Network Infrastructure
- 2.7 SaaS, Cloud Services and Infrastructure
- 2.8 Personnel Security
  - 2.8.1 Staff Background Checks
  - 2.8.2 Staff Onboarding and Offboarding
  - 2.8.3 Security Awareness Training
- 2.9 Physical Security

3 Secure SDLC
- 3.1 Goals of Secure SDLC
- 3.2 High-Level CI/CD Pipeline
- 3.3 Pipeline Security
  - 3.3.1 Infrastructure
  - 3.3.2 Build Pipeline Compliance and Security
- 3.4 Source Code Management (SCM)
  - 3.4.1 Code Reviews
  - 3.4.2 Source Code Analysis
- 3.5 Vulnerability Management, Software Composition and Supply Chain Risk
  - 3.5.1 Containers and Docker Security
- 3.6 Release Management & Artifact Signing

4 Security Capabilities of Digital Asset Products
- 4.1 Daml Enterprise Ecosystem
- 4.2 Daml

4.2.1 Daml Language and Packages
4.2.2 Daml Studio and Linter
4.2.3 Daml Script and Test Scenarios
4.2.4 Daml Runtime
4.3 Daml Applications - Common Deployment Pattern
4.4 Canton Distributed Ledger Technology
4.4.1 Daml “Virtual Shared Ledger”
4.4.2 Canton
4.4.3 Canton Protocol
4.4.4 Key Management and Connection Security
4.4.5 GDPR, Locality and Right to Forget
4.4.6 Multiple Sync Domains and Composability
4.4.7 Centralised vs Distributed Trust for Canton Domains
4.4.8 High Availability and Horizontal Scaling
4.5 Daml / Canton Threat Model and Attack Vectors
4.5.1 Threat Model
4.5.2 Attacks Against the Ledger API
4.5.4 Attacks via Malicious Daml Packages Files
Attacks against Daml Business Logic and Daml Application components
4.5.5 Attacks against Canton
4.6 Ongoing Security Testing of Daml and Canton
4.6.1 Threat Modeling and Code Security Reviews
4.6.2 Ongoing Security Testing and Evidence as part of the Build
4.6.3 Third Party Security Audits and Penetration Testing
4.7 Reporting Issues Discovered in the Daml or Canton
5 Security of Digital Asset Services (Daml Hub, Canton Network)
5.1 Daml Hub & Security
5.2 Canton Network

1 Digital Asset Security Initiatives and Programs

1.1 The Digital Asset Information Security Program

The goals of the Digital Asset Security Program include:

- Protect the information and privacy of our clients and partners.
- Protect Digital Asset staff, locations, data, services, and infrastructure from compromise or misuse.
- Ensure the appropriate security (Confidentiality, Integrity, and Availability) and privacy capabilities are built into Digital Asset products & services.
- Implement a secure Software Development Life Cycle (SDLC) process (plan, design, develop, package, distribute) to produce hardened, well-tested, high-quality products commensurate with their expected applications.
- Ensure Digital Asset clients, partners, and operations teams can deploy and run the product securely.

1.2 Security Program and Governance

The Digital Asset Information Security Program covers four primary views of information security:

- Corporate Security: the security of Digital Asset people, locations, data, systems, and services
- Secure SDLC: how we develop, test, and deliver our products
- Product & Service Security: the security of Digital Asset products and services
- Customer Security: enabling our customers to use our products and services in a secure manner

The Digital Asset Information Security Framework is diagrammed in Figure 1.

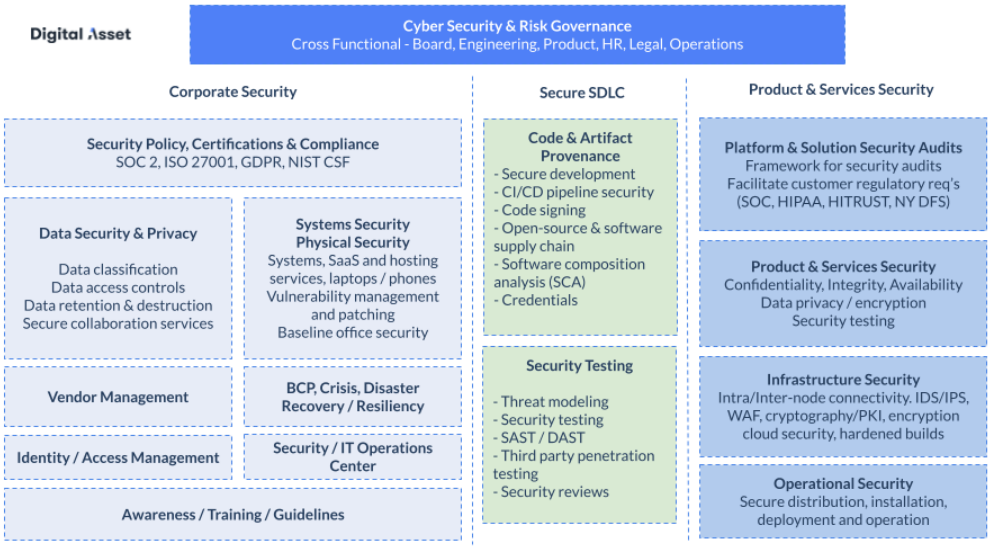

Figure 1: Digital Asset Information Security Framework

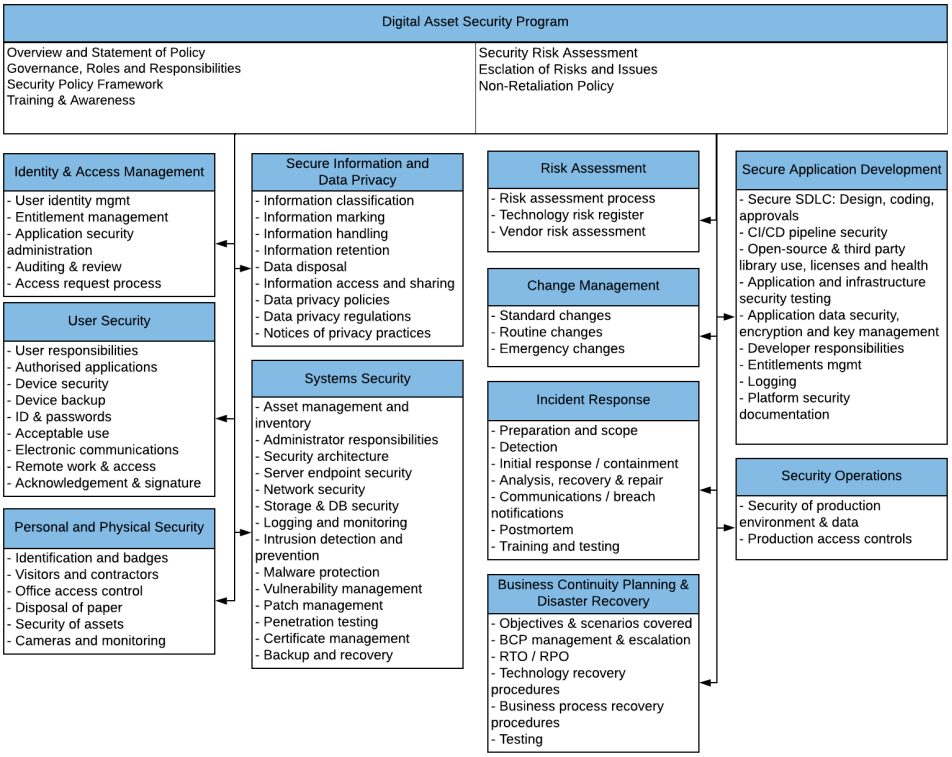

Figure 2: Details of the Digital Asset Information Security Framework

1.3 Digital Asset Security Team

The Security Team consists of staff who have made careers working for the largest financial services organizations. The team holds over 60 industry security certifications — CISSP, CCSP, CIPP, CIPM, ISC-CERT, CISM, CCSK, ITIL V3, and others —and has a breadth of experience across a wide variety of technology platforms and services.

1.4 Digital Asset Security Attestations and Certifications

Digital Asset monitors a variety of initiatives and regulations to ensure that it meets all regulatory, contractual, and compliance requirements. Our attestations and certifications provide confidence and external validation that Digital Asset's software and services are properly secured and protected against unauthorized modification or disclosure.

1.4.1 ISO 27001

Digital Asset is certified to ISO 27001:2013. The certificate is available on our Trust Center (https://digitalasset.com/trust-center). Digital Asset expects to certify against ISO27001:2022 in Oct 2024.

1.4.2 SOC 2

Since 2019, Digital Asset has successfully completed the SOC 2 Type 2 independent audit with an unqualified opinion from our auditor. The scope of the assessment is products and services with respect to the Common Controls and Security Trust Principle.

As the scope and range of services changes, we will also consider assessment against a broader scope of services and other SOC2 Trust Principles.

1.4.3 Privacy, GDPR and UK PECR

Digital Asset is compliant with the European Union General Data Protection Regulation (GDPR) and the UK GDPR and PECR regulations. Digital Asset monitors international, national and local privacy regulations to ensure we meet any requirements. We take the security and privacy of personal data seriously. Our privacy policy is published on our public website:

https://digitalasset.com/privacy

Questions relating to GDPR or our Data Privacy Policies can be directed to privacy@digitalasset.com

1.4.4 Cloud Security Alliance | Star Registry Participant

Digital Asset has completed and registered responses to the CSA CAIQ v4.0.2 in the Cloud Security Alliance STAR registry. A version of that same response including all detailed comments is available under NDA from Digital Asset directly.

1.4.5 Responsible Disclosure of Security Vulnerabilities

Digital Asset maintains a Responsible Disclosure policy for reporting security vulnerabilities and concerns. We do not currently operate a bug bounty program. The policy is detailed on our public website:

https://www.digitalasset.com/responsible-disclosure

The Digital Asset Security Team can be contacted at security@digitalasset.com

2 Security of the Company

2.1 Risk Management and Governance

Governance of the Risk Management program is executed through the Risk Management Committee, with a membership consisting of the COO, CFO, CISO, CTO, CPO, Head of Marketing and our General Counsel. The committee meets regularly to review identified risks and prioritize mitigation efforts. The committee reports to the CEO and, ultimately, to the Board on the critical concerns of the firm.

2.1.1 Risk Management Policies and Procedures

Digital Asset maintains a set of industry standard policy and procedures, including:

- Information Security Program & Policy Framework
- Data Classification, Protection, Retention and Disposal
- Change & Incident Management
- DR, BCP and Crisis Management
- Vulnerability and Patch Management
- Vendor and Supply Chain Risk Management

These are reviewed annually, and staff are required to acknowledge them annually.

2.2 Data Governance and Oversight

All Digital Asset staff are required to review and acknowledge the Digital Asset Secure Information Policy and Data Privacy. This defines data classification tiers for all data and defines the appropriate handling, backup, retention, and destruction requirements.

Digital Asset requires appropriate risk assessment of all data services used for storage and processing of Digital Asset data, including data shared from partners and clients. Reviews are particularly focused on Personally Identifiable and Personal Health information, but also on the broader compliance requirements for GDPR and related data privacy regulations.

2.3 Supply Chain Risks and Vendor Management

As a supplier to many financial services and financial market infrastructure providers, Digital Asset is aware of its position in the overall supply chain. The company works to ensure that our customers can trust us with their data, trust our products in their environments, and trust applications built on our technology.

Digital Asset Vendor Risk Reviews are performed for all vendors/business associates. The use case, scope, and data class in use drive the risk assessment process. Overall risk rating and general risk maturity are created specific to the vendor and the data class being handled. Critical vendors are reassessed annually or on major changes of use.

2.4 Email & Web Security

Digital Asset has implemented mail gateway services for inbound and outbound email security. This filters and protects emails to staff, including spam, malicious emails, attachment and URL security, and malware protection. Phishing awareness training is provided to all staff, and periodic phishing testing is performed to drive further awareness.

Digital Asset has implemented local endpoint agents and browser extensions to protect against website-based attacks and other forms of spam and phishing attacks.

Digital Asset has implemented email security controls (DMARC, DKIM, and SPF) to reduce the opportunity to impersonate the firm or staff.

2.5 Endpoint Security

All users are required to use Digital Asset managed endpoints for work-related activities. All endpoints are managed centrally by MDM solutions. Native capabilities of the device and MDM services are leveraged for full disk encryption, firewall, configuration compliance, security agent enforcement, screensaver locks, and other security requirements.

Digital Asset leverages a number of host-based agents for malware detection and prevention, DLP & data egress restrictions, audit, and oversight, vulnerability scanning.

2.6 Network Infrastructure

Digital Asset corporate network is in place between Digital Asset offices and cloud provider private networks. Network segmentation is used to segregate networks from each other within each cloud environment. Firewalls are used at all office locations to block ingress malicious traffic. Security Groups and firewall rules are used in Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure to block ingress malicious traffic.

The network infrastructure is patched in a timely fashion, with monthly configuration samples taken as part of our audit-control process. Network traffic is sampled and reviewed, and flow traffic is sent to our SIEM for malicious traffic analysis.

Secured, full tunnel VPN is installed and available on all Digital Asset endpoints through a connection icon in the menu bar of each endpoint, should the Digital Asset employee find themselves working on an untrusted network.

2.7 SaaS, Cloud Services and Infrastructure

Digital Asset leverages many SaaS cloud-based services and also uses public cloud (AWS, GCP, and Azure) to host and run our corporate infrastructure.

A mixture of public and private network segments, with firewall rules and access control lists in each cloud provider, are used to ensure that only trusted entities are allowed access to resources within these cloud environments.

IAM roles and policies are in place to control user and administrative access to the cloud environments and administrative consoles. The principle of Most Reasonable Privileged Access is followed when provisioning user and service account access to the cloud resources.

All rights and account access are regularly reviewed and inventoried as part of the Digital Asset audit process.

For data held in SaaS services, Digital Asset has engaged with security vendors to provide guardrails, security oversight, and automated permissions revocation for cloud SaaS, and the associated third-party extension/add-on ecosystems. Application access via third-party extensions or add-ons is reviewed for risky integrations.

Security events from syslogs, network flow logs, and cloud audit logs are forwarded to our SIEM tool for centralized logging, alerting, threat detection, and ticketing for remediation. The security team monitors the logs to identify potential threats and unauthorized activity and follows appropriate steps for remediation.

The vulnerability assessment platform and other Cloud Security Posture Management (CSPM) solutions are used to identify and remediate vulnerabilities and misconfigurations in all environments. Additional open-source security tools and in-house developed solutions are also used to audit and validate security configuration and access.

2.8 Personnel Security

2.8.1 Staff Background Checks

All Digital Asset staff are required to pass background and screening checks. This includes ten-year criminal background checks. These checks are performed at a local level and include checks against the legal systems in the country of residence.

2.8.2 Staff Onboarding and Offboarding

Digital Asset employs comprehensive onboarding and offboarding processes that include the provisioning and deprovisioning of physical resources, as well as user logon accounts, logical access rights, and data access permissions. User access is revoked on last day of employment. Staff access rights are periodically reviewed for alignment to role or function.

2.8.3 Security Awareness Training

Security is considered a key responsibility of each and every member of staff. The company provides onboarding and ongoing awareness training. Training includes:

- Onboarding security training for all new hires

- Annual Security Awareness training and acknowledgement
- Periodic Phishing testing and awareness
- Use-case- or SME-related training, as required
- Annual Policy review and acknowledgement

Employees are provided training for emergency situations with CPR, fire drills, and table-top exercises as available.

2.9 Physical Security

Digital Asset requires physical security controls for all office locations. These may be maintained by Digital Asset directly or utilize building management or office services. Each staff member is required to have an individual access token to access the office spaces. Security cameras or CCTV are required on all ingress/egress points and for any internal secure technology or IT closet. Ingress lists are produced on a monthly basis and reviewed against current employee lists.

Digital Asset does not maintain its own data centers, but utilizes premier-tier cloud services providers, including Amazon AWS, Google GCP, and Microsoft Azure. The security of physical access to such services is delegated to the service providers, who can provide the appropriate security certifications and assessments on request.

3 Secure SDLC

Digital Asset runs full CI/CD pipelines for the development, build, packaging, distribution, and deployment of our technologies. The firm has implemented a variety of controls to ensure the security, quality, and provenance of our products and services. The Digital Asset Security Team continually assesses these controls in light of changes in industry best practices, our product features, services, and choice of technology.

3.1 Goals of Secure SDLC

- Security considerations are brought as early in the process as practical.
- All code is tested, reviewed, and approved prior to submission.
- All commercial and open-source dependencies are understood and tracked.
- Licensing of dependencies is understood and approved.
- Vulnerabilities are identified and mitigated during development and during lifetime of artifacts.
- Opportunities for introduction of Harmful Code accidentally or intentionally are removed.
- Enhancing our practices to align with industry best practices as these evolve.

3.2 High-Level CI/CD Pipeline

Digital Asset follows industry best practices in the setup and management of its CI/CD pipeline (Figure 3). Digital Asset has implemented and continues to evolve a set of controls at various gates of the pipeline.

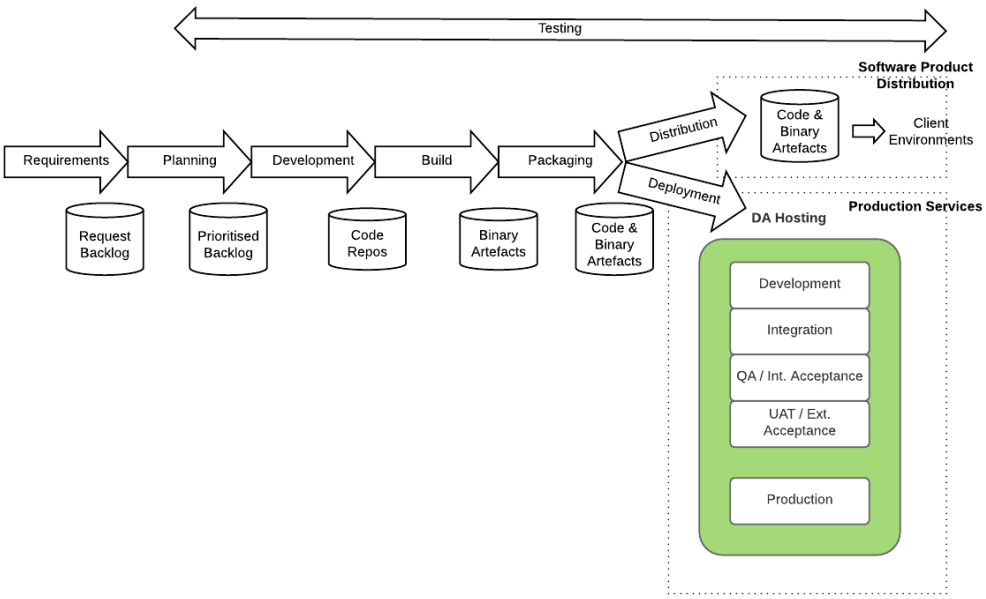

Figure 3: Diagram of SDLC Pipeline

3.3 Pipeline Security

3.3.1 Infrastructure

CircleCI and Microsoft Azure DevOps-based pipelines are used to build Digital Asset products. Administrative access to these systems (build pipelines, configuration, and infrastructure) is restricted to an approved set of Security and DevOps engineers. Access logs are sent to our SIEM for review.

Sensitive credentials for the build pipeline are managed and secured through the pipeline KMS services and are not stored in source code or configuration files. Access to the build infrastructure is also restricted to a core set of staff.

3.3.2 Build Pipeline Compliance and Security

Digital Asset has implemented Pipeline Security Compliance checks. This includes:

- CI/CD and SCM posture checks
- Credential scanning and reviews, including access or API tokens, SSH keys, webhooks
- Third-party extensions and add-ins, and their access permissions
- SCA checks for vulnerability and licensing
- Pipeline access permissions review and behavior monitoring
- Artifact integrity checks
- Artifact repo deploy permissions

3.4 Source Code Management (SCM)

Digital Asset uses GitHub as its source code management system for its proprietary and open source projects. Branch protection rules are enforced for code reviews (see below) and build & test check completion prior to code merge.

For open source projects, CLA (Contributor License Agreements) are enforced for external contributions with mandatory Digital Asset staff reviews of contributions.

3.4.1 Code Reviews

All GitHub Pull Requests (PRs) are reviewed prior to being committed to the “main” branch. GitHub branch protections are used to enforce that all PRs must successfully pass all automated tests (unit, integration, functional, non-functional, CLA, etc.) before they can be merged. Additional tests and jobs are scheduled daily and weekly to perform other security scanning activities.

In the case of submissions to our open source projects, Digital Asset staff are required to review all external contributions prior to merging to the main branch. No external staff are authorized to approve such code submissions.

Where code submissions may result in production changes (Infrastructure / Application as Code deployments, External Releases), peer review and approval are required as part of our Change Management process.

Digital Asset continues to improve the efficacy and completeness of its code reviews and testing through new tools and through developer security awareness training.

3.4.2 Source Code Analysis

Digital Asset utilizes a variety of tools to review its code (SAST, Veracode, credential scanners and compiler restrictions). Many Digital Asset staff have backgrounds in security and formal analysis. These reviews cover both functional and non-functional aspects. Digital Asset continues to evaluate new tools in this space to improve the effectiveness and completeness of coverage.

3.5 Vulnerability Management, Software Composition, and Supply Chain Risk

Digital Asset has open sourced significant portions of the Daml code base and is also a heavy user of open source components and libraries. Digital Asset uses commercial and open source tools to discover (Software Composition Analysis, SCA) and validate the set of all open source dependencies. These tools allow Digital Asset to identify improper licensing or vulnerabilities in upstream dependencies that are used in Digital Asset products.

Synopsys BlackDuck is used to identify dependencies in Digital Asset products and to scan for software license issues (toxic and copyleft license types) and vulnerable dependencies. All identified issues are ticketed and assigned to relevant product or component owners for assessment and prioritization. Digital Asset provides a Bill of Materials (BOM) detailing all components in a release and their associated licenses.

Digital Asset also leverages other tools, such as GitHub Repository Vulnerability Analysis and Google Container Registry Analysis to identify dependencies and their associated vulnerabilities in source code and Docker containers.

As this is a rapidly evolving area in the security space, the firm continues to evaluate additional tools and works to understand best practices in this area.

3.5.1 Containers and Docker Security

Digital Asset uses Docker technology during the development, testing, and distribution of our products. We use Blackduck SCA Analysis to scan for vulnerabilities in published images. We continue to evaluate additional tools to enhance our capabilities of vulnerability analysis and runtime access control and protection.

In line with other open source projects, Digital Asset updates the base images of Docker containers as new ones are released, keeping the same minor & patch release version. Changes to functionality or updates to dependencies to the components layer on top of base layers

result in a new patch release version. Companies that use Docker caching repos may need to ensure that updated images are pulled for base layer changes.

Due to the wide variety and quality of such tools, we work with our customers to evaluate any differences they detect through their chosen tools.

3.6 Release Management & Artifact Signing

Digital Asset software releases are taken from the source repo main branch and built through the automated CI/CD tools. The main branch reflects the latest set of peer-reviewed and tested code.

All releases are tested through automated and manual tests prior to publishing and distribution.

Code signing is done when pushing to a variety of external artifact repos, and access to the signing keys is controlled within the CI/CD toolset. Only a very small set of engineers have access to the signing keys.

Details of our public signing key can be found here :

https://docs.daml.com/getting-started/manual-download.html

4 Security Capabilities of Digital Asset Products

Digital Asset works with the world's largest companies to build solutions that synchronize complex multi-party workflows, reduce data reconciliation, lower operational costs, and mitigate risk, while ensuring the privacy of the transactions. We combine deep industry expertise with an intuitive Smart Contract modeling language and an extensive partner network to deliver software that harnesses the benefits of Distributed Ledger Technology (DLT).

The following sections describe the security features of Digital Asset's software including Daml, the privacy-focused smart contract application platform, and the Canton blockchain protocol. Please read this section alongside our comprehensive, user-friendly, product documentation.

https://docs.daml.com/index.html

4.1 Daml Enterprise Ecosystem

The Daml Enterprise ecosystem contains a number of key concepts and technologies:

- The Daml Language, a strongly typed, functional language for modeling business workflows and associated functionality and data access permissions. Digital Asset provides an SDK for the development of Daml models.
- The "virtual shared ledger", a hypothetical distributed ledger that executes transactions modeled in Daml and enforces the security and privacy of the transactions.
- The Canton blockchain protocol, a concrete implementation of a Daml ledger, that synchronizes and enforces the sequencing of executed transactions across the distributed participants. This is shipped as Daml Enterprise runtime.
- Daml Hub, Digital Asset's managed, hosted ledger service.
- The Canton Network, the privacy-enabled blockchain network designed for institutional assets. This includes a decentralized global synchronization service and network utilities to enable the composability of independent applications built with Daml.

The following provides further detail specific to the security features of Daml and distributed ledgers more broadly.

4.2 Daml

Daml is Digital Asset's open source, intuitive smart contract programming language used to digitize multi-party agreements and automate transactions in a precise and secure manner. Daml enables enterprises from startups to large, highly-regulated organizations to achieve more efficient business processes, reduce risk, and develop new products and services that can transform an industry.

The Daml smart contract language is designed for decentralized systems with very low trust assumptions. Trust is the confidence in participants and the network with respect to how well-behaved they are in terms of intention and reliability. The language, runtime, and ledger implementations have accordingly been developed with security as a high priority.

4.2.1 Daml Language and Packages

Daml is a strongly typed functional programming language that is compiled into an intermediate format (DAML-LF) that is interpreted by the Daml runtime during execution. Daml has built-in mechanisms to express authorization and privacy for multi-party workflows. Developers write the Daml-based workflows that define the data, transactions, and permissions, and when combined with other components for UI and automation, form a “Daml pplication.” For more details, see the section Daml Applications" below.

Because the Daml language is a strongly typed language, this restricts many of the traditional vectors that can introduce software vulnerabilities. Daml is more rigid than most smart contract languages and even most general-purpose languages. It is fully statically typed, all data is immutable, dependencies are always an acyclic graph, all Daml is compiled to DAML-LF, and Daml comes with built-in test tools in the form of Daml Scripts.

Many common errors, including whole Mitre Common Weakness Enumeration (CWE) categories (e.g., incorrect casts, buffer over-/underflows, wrap-around errors, indexing/memory allocation errors, errors with pointers, or re-entrancy bugs) are prevented by the Daml language and toolset. These include restrictions on memory, network and other I/O, file system access as well as removing the need to handle or manage cryptography or key material. As such, traditional SAST tools are not generally available or needed for scanning Daml-based workflows. These tools are still relevant for the non-Daml components (see the "Daml Applications" section below).

The Daml compiler is based on the well-supported open source GHC (https://www.haskell.org/ghc/) compiler[1] with many years of research and community effort behind it. Digital Asset has incorporated extensions to GHC’s Parser to add support for language features unique to Daml and components to convert the GHC-internal GHC-core representation to the DAML-LF[2][3] format.

.DALF Files: A Daml package is a binary-encoded protobuf message of type “Package” as defined by the Google protocol buffers. Daml packages are typically stored as *.dalf files. Each DALF file is linked to the hash of the binary contents, and this ensures that the package cannot be unknowingly modified. The version of a Daml package is tied to this hash, such that modified versions would be seen as completely independent libraries and ignored.

---

[1] **Glasgow Haskell Compiler** is a state-of-the-art, open source, compiler, and interactive environment for the functional language Haskell

[2] DAML-LF is an extension of System-F (the polymorphic Lambda Calculus) and is represented in the DAML-LF protobuf schema.

[3] Google Protocol Buffers (Protobuf) is a language-neutral method of serializing structured data i.e.: an extensible way of serializing (something which is being published or broadcast in a number of parts) structured data for use in communications protocols and efficient data storage. It is a popular, supported, widely-used framework.

.dar files: The output of the Daml compiler is a *.dar file, which is essentially a Zip archive containing metadata, the *.dalf Daml package of the compiled Daml file(s), and the *.dalf files of any dependencies. Since the dependent Daml packages are included in the DAR file, the runtime has everything it needs to run the application. The Daml packages are uploaded and distributed across ledger nodes at runtime, subject to Change Approval and verification by the node operator.

All references to templates are keyed off hash values of the template contents, so attempts to alter the code would be ignored as they would be seen to be another unrelated package.

4.2.2 Daml Studio and Linter

Daml Studio IDE is a user-friendly developer environment for Daml templates and packages. It is based on the open source Microsoft VS Code IDE and provides Daml-specific extensions for workflow development and testing.

As part of the Daml Studio, Digital Asset provides a consistently enhanced and configurable static analysis tool and linter - DLint - for the Daml language. The Daml DLint builds on the capabilities of the Haskell equivalent – HLint – a tool that has been in use and with ongoing upgrades for 15+ years.

DLint runs alongside developers as they code their business workflow logic and test scenarios, and it highlights a variety of best practices, including:

- DA-recommended Daml language best practices
- Lambda, Monad, and Recursion handling recommendations
- "Code smells" - *In computer programming, a code smell is any characteristic in the source code of a program that possibly indicates a deeper problem.*
- Reducing code duplication and better structure for templates
- Static detection of potential runtime errors

DLint can be run as a standalone tool as part of the security and quality checks of a Continuous Integration (CI) build pipeline.

Digital Asset continues to enhance this capability as best practices, potential language traps, and recommendations evolve. Digital Asset also works with our partners to provide reference application examples, highlighting the greatest efficiencies, best practices, and optimized transaction patterns.

4.2.3 Daml Script and Test Scenarios

Daml provides Daml Script as a way to perform initialization and testing of workflows during development, during a CI/CD build and against live ledgers. Developers can write scripts within Daml Studio IDE or via CLI and REPL command line utilities to execute a variety of unit, integration, and regression tests against their Daml models.

Daml Studio provides developers with mechanisms to define “test scenarios”, including the setup and use of parties and contracts, that allow live execution and analysis of Daml workflows as part of the development process. These execute against a local in-memory Daml runtime for validation of business workflow outcomes. As the developer is writing and testing code, the IDE provides feedback on the execution of the Daml models.

Command line versions via CLI and Daml REPL allow for more automated execution of initialization and testing during build and deployment against live persistent ledgers.

4.2.4 Daml Runtime

The Daml runtime runs Daml code (in the form of DAML-LF). It is used to interpret commands to turn them into transactions and to validate transactions by “reinterpreting” them. Commands are sent to the runtime by Daml application components via the Ledger API.

The Daml runtime has read access to a Private Contract Store (PCS), which is a persistent key-value store storing currently active contracts. Keys are Contract IDs, which are restricted, typed strings, while values are binary encoded protobuf messages of type “ContractInstance” from the transaction schema.

The runtime is invoked either with an “Update” from the DAML-LF schema or with a transaction from the transaction schema. In either case, the engine will load the referenced Daml packages, try to fetch any referenced contract from the PCS, and then evaluate DAML-LF to generate or validate a transaction.

The Daml runtime is an abstract CEK machine written in Scala that runs inside a Java virtual machine (JVM). The CEK machine implements small-step semantics for the lambda calculus by starting with an initial state and taking incremental "steps" to evolve the computation. Like a board game, the CEK machine defines a set of configurations and defines a step function that transitions from one configuration to the next.

4.3 Daml Applications - Common Deployment Pattern

A typical deployment of a Daml-based system is depicted in Figure 4.

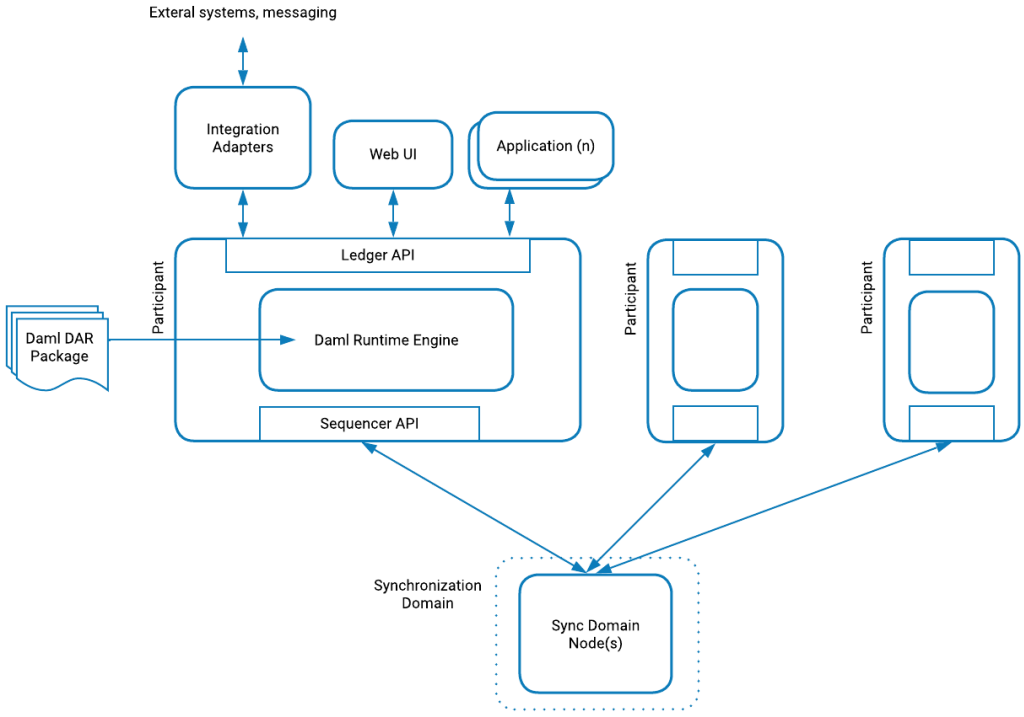

Figure 4: Daml Application

A Daml-based application is a combination of:

- Participant Node: executes and enforces state changes based on the rules and transitions defined in Daml workflows. Persists a local copy of the state for the hosted parties. Interfaces to one or more synchronization domains to execute and validate distributed transactions.
- (one or more) Synchronization (Sync) Domains: synchronizes the (immutable) state across participants and allows for verification and privacy of data.
- Web UI or other human interaction with the system: initiates changes to the state of the system
- Automation (aka Triggers): drives the state of the business workflow through automation and business rules
- Integration adapters: interfaces with external systems and users; this might include industry-standard messaging, such as ISO20022, CDM, Fix, etc.

The concepts of participants and synchronization domains and how these get connected in a Distributed Ledger are explained in the following sections.

Application components interact with the ledger via the Ledger API, served by the participant node. The Ledger API is a gRPC API with messages described by the Ledger API Schema. Applications and associated parties executing transactions are authenticated over secure TLS connections using JWT tokens.

An example of configuring (m)TLS connectivity and JWT tokens and use of the client-side tools and User Management services is available via the Secure Canton Infra reference application on GitHub.

4.4 Canton Distributed Ledger Technology

Distributed ledger technology provides data persistence with structure, integrity, and automation driven by smart contracts. A distributed ledger typically consists of multiple participant nodes, dispersed across organizations and locations, and distributed applications that define the workflows and agreements among mutually distrusting entities. Depending on the specific features of the technology, distributed ledgers provide varying levels of anonymity, privacy, performance, and trust assumptions.

The technology has a number of advantages over traditional databases, including:

- Security: Because the ledger is distributed, it is not vulnerable to single points of failure.
- Decentralized trust: The ability to provide trust in the absence of a central operator.
- Privacy: Specifically to Canton, the use of segmented data allows for full privacy of data and transactions.
- Reconciliation: Removal or reduction in reconciliation through a shared, trusted dataset amongst participants, reducing costs and risks.
- Immutability: Provides a tamper-proof record to enhance security and confidence in data accuracy.
- Atomic transactions: Ensures that transactions are executed atomically (either all of the transaction is completed or none of it is), and with guaranteed finality.

The specific details of the security capabilities of a ledger are dependent on the implementation chosen. The selection of a specific ledger implementation is driven by the trade-offs of various functional and nonfunctional capabilities, including performance, security, resilience, distribution, identity, and trust assumptions.

4.4.1 “Virtual Shared Ledger”

Daml enforces a model of a virtual shared ledger. Individual parties can participate in a shared ledger where access to data is enforced by the Daml models. See the Daml documentation (Ledger Model) for the detailed specification, with particular focus on:

- Integrity
- Privacy
- Causality

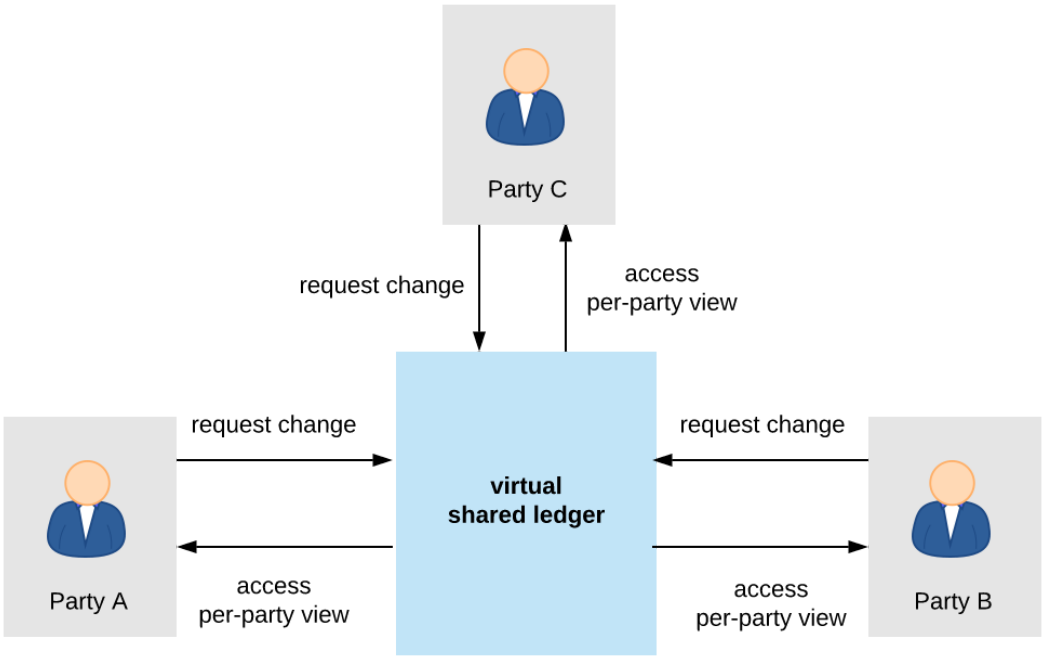

Figure 5: Daml Virtual Shared Ledger

A Daml party sees a single shared ledger where they only have visibility to contracts they are stakeholders in or observers of. A party can request changes to a common shared state and see the consequences of requests from other users.

4.4.2 Canton

Digital Asset's implementation of a distributed ledger is Canton and the Canton protocol. Full background details are available in the original whitepaper[4] and the Canton High Level Requirements[5] on the Daml documentation site.

Key Canton concepts:

- Participant node: a high availability (HA)-capable distributed ledger node that contains the Daml runtime, exposes the Ledger API to applications, maintains a local copy of contracts for parties hosted on the node, and uses the Canton protocol to transact with other participants.
- Canton protocol: a distributed ledger protocol that orders transactions flowing through the system and coordinates the global ordering, validation, and atomic commit of transactions across involved participant nodes, while ensuring privacy. A synchronization domain coordinates the validation and atomic commit across validating participants to a transaction.
- Synchronization domain: implements the Canton protocol and is offered in a variety of persistence and distributed consensus options.

[4] Canton:A Daml based ledger interoperability protocol: https://www.canton.io/publications/canton-whitepaper.pdf
[5] https://docs.daml.com/canton/architecture/requirements/requirements.html

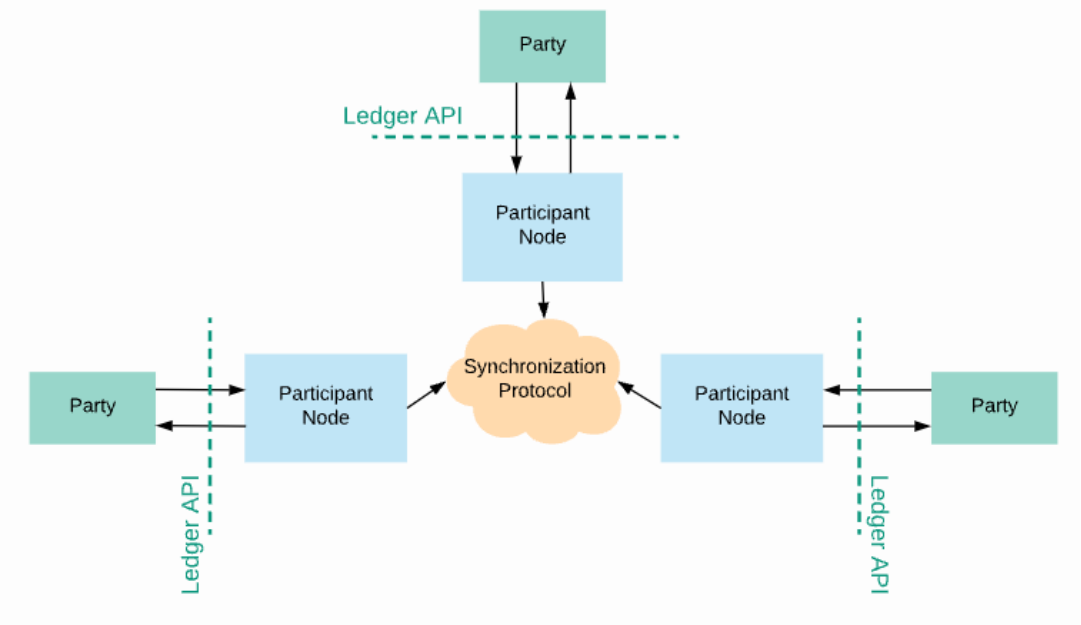

Figure 6: Canton-based Ledger

Canton is a next-generation Daml-based distributed ledger that implements the "virtual shared ledger" faithfully. By partitioning the global state, it solves the privacy problems and scalability bottlenecks of other DLT solutions.

When using a Canton-based ledger, each entity deploys a local participant node that hosts the Daml runtime engine, which executes Daml workflows and persists the set of contracts that the parties on that participant node have visibility to. The global state of the overall ledger is partitioned across the connected participants and no single entity has access to all of the global state (unless explicitly defined by the Daml model).

Each participant node connects to one or more synchronization domains that order requests, provide a multicast service, and aggregate verdicts/signatures of their virtual ledger. Each participant node validates the transaction by re-executing the Daml commands and uses the Canton protocol to order and atomically commit transactions to the shared ledger.

Daml packages that need to execute or verify the specific model's transactions are distributed across those nodes. Daml packages are validated by the Daml runtime prior to use. The participant node operator verifies the provenance of the Daml package prior to upload into their node. This is required before transactions involving the package are evaluated by the participant node.

4.4.3 Canton Protocol

At a high level, the primary goal of the Canton protocol is to ensure the consensus of the shared state across its participant nodes, where the shared state is the active contract set for which

they are joint stakeholders. Each participant node has multiple subsets of active contracts for the transaction in which it was a stakeholder. Participant nodes never see, store, or process contracts for which they are not involved.

To achieve this, Canton has implemented a state replication service that ensures consensus by requiring deterministic transaction processing (deterministic state machine) and globally ordered transactions (achieved through a consensus mechanism). Each participant applies the transactions in the given order and achieves consistency due to determinism. To ensure privacy, the entire global state of the "global virtual ledger" is not replicated to all participants.

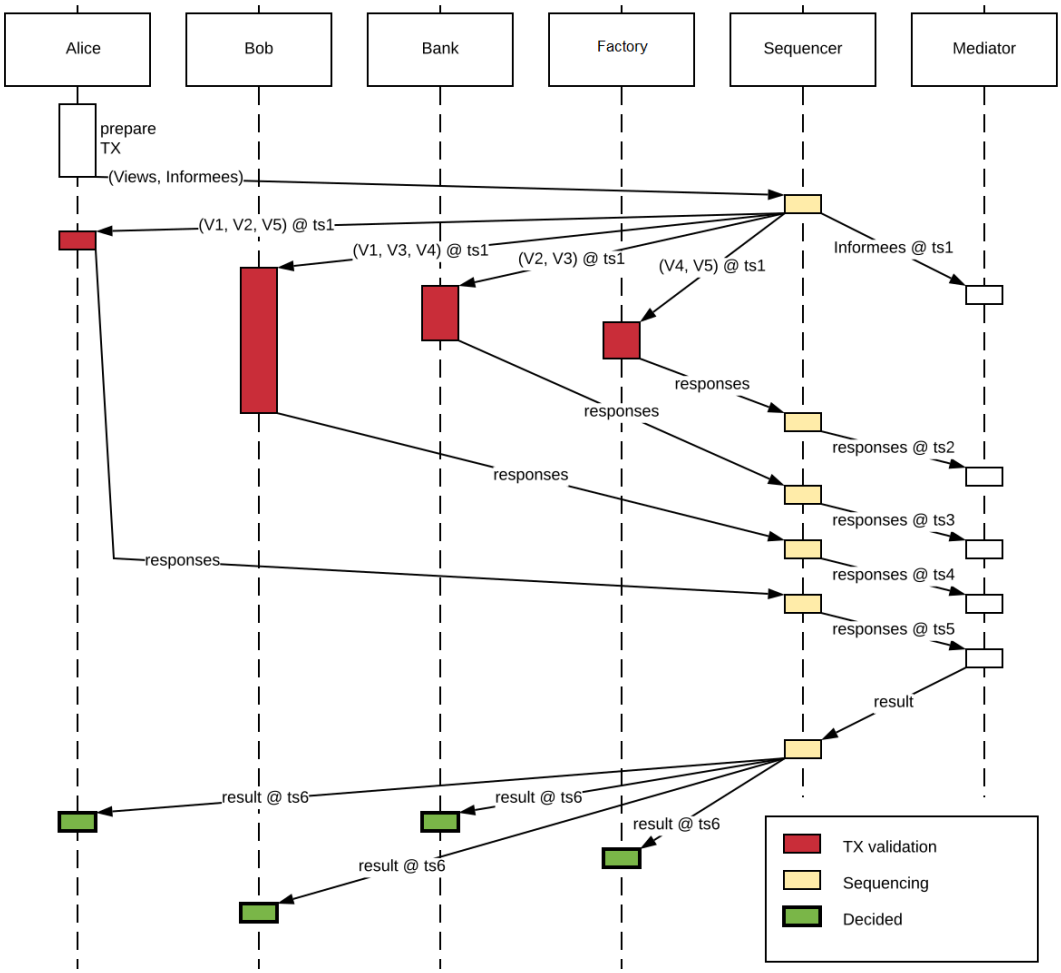

Figure 7: Canton Message Flow

In the example in the preceding figure, Alice initiates the process by submitting a command via the Ledger API to her local participant node. The node executes the command against the Daml model, which produces a hierarchical tree of (sub-)transactions. A view of the transaction tree is generated for only those sub-transactions involving each peer participant and a *confirmation request* is sent through the synchronization domain to each recipient. The request is encrypted pair-wise with the recipient's keys so that the domain has no visibility into the transaction

contents; nor do recipients see each other's transactions. Each receiving node validates the set of subtransactions by re-running the Daml model and attempting to lock the involved contracts. If the validation fails because the submitter is malicious, or because a lock is already taken, the recipient sends a negative response. Otherwise it sends a positive one. All responses are signed and sent to the mediator, a component run by the synchronization domain. Once all responses have been received and the complete transaction tree is validated, the mediator sends a final commit or abort request to the set of involved participant nodes. After receiving the response, the receiving nodes release the locks and, if there is a positive result, apply the transactions to their local contract store. The mediator's role is to ensure privacy: each participant node never sees the sub-transactions for other participant nodes.

Total global order is achieved via the sequencer, which provides an incrementing time counter for transactions. The sequencer can be deployed in a variety of configurations, from database-backed for more centrally trusted deployments to BFT/DLT-backed where distributed trust is required.

4.4.4 Key Management and Connection Security

To achieve the above global synchronization, Canton utilizes industry standard cryptographic primitives and algorithms. Canton utilizes asymmetric and symmetric encryption, hash primitives, and HMAC as part of authentication and authorization, encryption, and signing of all activities, as well as secure connections (TLS) and Ledger API access (JWT tokens).

For more details about cryptographic primitives, see the Daml documentation: https://docs.daml.com/canton/usermanual/security.html

On initialization, each node (participant or synchronization domain component) creates a unique namespace (the fingerprint of the initial public key of the node). From this namespace, it then generates separate encryption and signing keys, which are used in the protocol interactions with other nodes. On joining a synchronization domain, the public keys are shared with other participants. This allows the verification of the identity of other interacting nodes, including the synchronization domain, and enforces security of all Canton protocol communications.

The Canton keys are managed internally within each node and are functionally similar to self-managed PKI certificate authorities and wallets. However, it is not possible to use PKI infrastructure for the creation or management of these keys. Note that each Canton node signs transactions on behalf of the parties hosted on the node. Each party does not maintain a separate "wallet." Depending on your needs, deploying an individual participant node for a single party would be functionally equivalent.

For secure deployments, Canton supports the use of KMS / HSM solutions (currently AWS and Google Cloud KMS) to generate and store keys. We support two modes of operation:

- Envelope Encryption: All node keys (Data Encryption Keys) are generated in memory, and the storage and persistence of the keys uses an HSM-managed symmetric key (Key

Encryption Key). This offers the greatest performance while securing the keys for persistence.

- KMS-based Keys: All node keys reside natively within the KMS and are never exported outside of the device. This option offers the most secure keys (up to FIPS-140 hardware security modules (HSM)) but the tradeoff is performance: certain operations require network access, and consequently network delay, to process.

All API endpoints support minimally the configuration of TLS (using traditional PKI X509 certificates). The Ledger and Admin APIs also support mutual TLS client certificates and JWT tokens.

JWT Authorization is detailed in the Daml documentation[6] and the "Secure Canton Infrastructure" [7] reference application.

4.4.5 GDPR, Locality, and Right to Forget

Canton's ability to process data only on the node where parties are allocated enables deployments that support the data locality requirements of GDPR and similar data privacy legislation. Participant nodes can be deployed into specific geographic regions to host the entity data, if required.

The GDPR data subject Right to Forget requirement often causes issues for other distributed, fully replicated global state systems. Canton offers the ability to prune archived contract entries through an auditable process, thus enabling the removal of PII data from the node.

See Ledger and Participant Pruning under Advanced Operations in the Daml documentation.

[6] https://docs.daml.com/app-dev/authorization.html
[7] https://github.com/digital-asset/ex-secure-canton-infra

4.4.6 Multiple Synchronization Domains and Composability

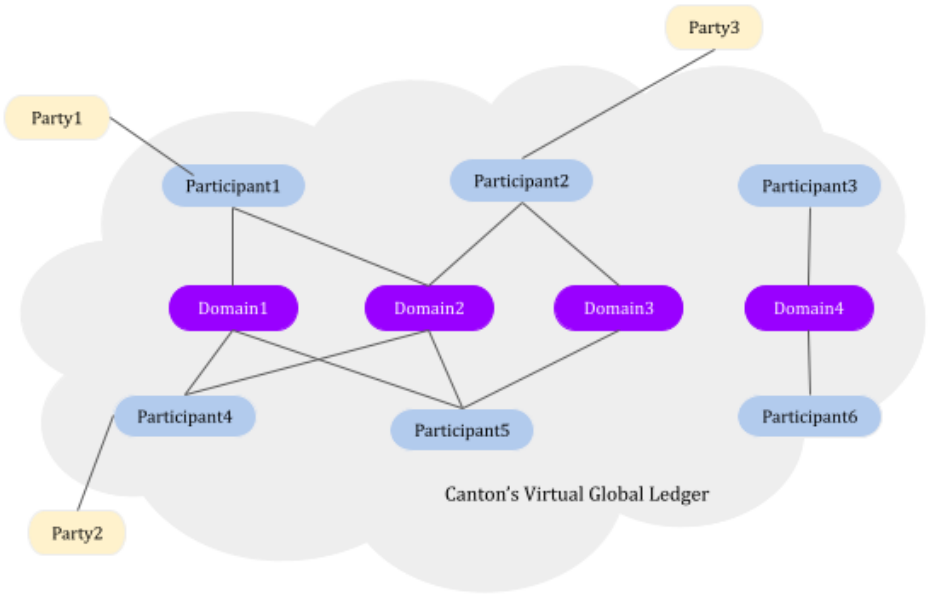

Figure 8: The virtual global ledger

A single Canton synchronization domain allows for the atomic composition of arbitrary workflows hosted on that domain. Participant nodes connecting to multiple synchronization domains also enables:

- greater composability of workflows and the generation of greater business value from network effect of the larger network.
- flexibility of deployment balancing scalability, throughput, trust assumptions, resilience, etc.

For example, having a single synchronization domain might result in high latency for participants connecting from other geographical locations. A single synchronization domain might become a bottleneck in the sequencing and processing of transactions; multiple synchronization domains allow for increased scalability through parallel processing of unconnected workflows, greatly increasing the throughput of the overall system. Regional synchronization domains can be deployed for localized performance, reverting to a global synchronization domain for any cross-regional workflows. Participants sharing a common connected synchronization domain can agree to sequence their transactions through that synchronization domain, should the original synchronization domain no longer be available.

Additionally, independent synchronization domains can be deployed for other operational reasons. A more restricted synchronization domain might be deployed for more sensitive

transactions behind firewalls to protect from denial of service or other malicious attacks, or where there are specific trust requirements of the participating entities.

4.4.7 Centralised vs. Distributed Trust for Canton Synchronization Domains

The Canton protocol supports a variety of models around trust and consensus of the synchronization domain sequencing. For situations where the entity operating the synchronization domain is trusted by all participants, a database-backed synchronization domain option is available. The domain operator runs the domain and the sequencing actions are persisted to a database (currently PostgreSQL or Oracle).

For deployments where there is less trust of the synchronization domain operator, or a consortium of entities collectively needs to ensure the synchronization domain is trusted against malicious actions, Canton offers options to use a DLT or BFT backend.

In a situation with multiple synchronization domains, it is possible to deploy combinations of these, each matched to the trust assumptions and requirements of the relevant applications.

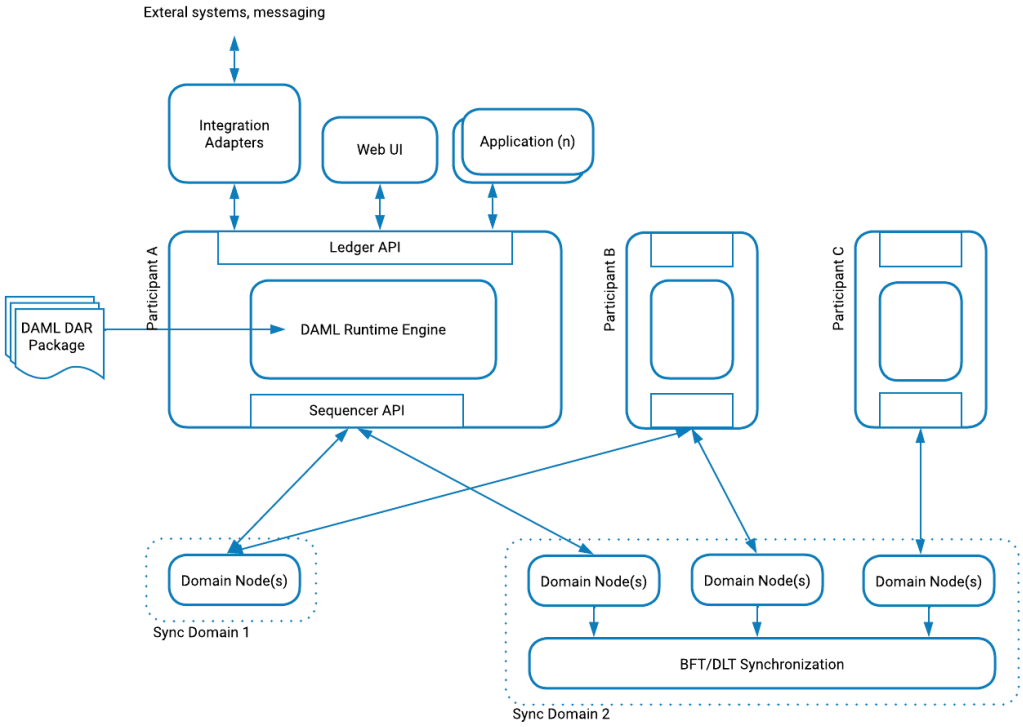

Figure 9: Multiple Domains

Figure 9 demonstrates the ability to connect to database-backed synchronization domains or DLT/BFT-backed synchronization domains in combination. Participants A & B can select which transactions pass through either Synchronization Domain 1 (database-backed) or Synchronization Domain 2 (DLT-backed). All participants, including C, can compose application

workflows through Synchronization Domain 2, which offers distributed trust if the synchronization domain nodes are deployed across different organizational entities.

For details of the DLT-backed option, see the documentation:

https://docs.daml.com/canton/usermanual/domains/domains.html

Additional DLT- and BFT-based consensus options are in development.

4.4.8 High Availability and Horizontal Scaling

Canton nodes, whether in the participant or synchronization domain configuration, can be deployed for high availability and scaling. These are detailed here:

https://docs.daml.com/deploy-daml/infrastructure-architecture/high-availability/main-menu.html

Depending on the node type, options for active-active or active-passive deployments are possible.

4.5 Daml / Canton Threat Model and Attack Vectors

The design of Daml and Canton provides protection against several broad classes of attacks. The following sections describe attack vectors, risk mitigation, and defense strategies.

4.5.1 Threat Model

The following diagram shows the primary trust boundaries within a Canton deployment:

- Ledger API applications are untrusted with respect to participants
- Participant nodes are untrusted with respect to their connected synchronization domain(s) and between each other
- Daml packages (DARs) must be approved before they are uploaded to a participant node

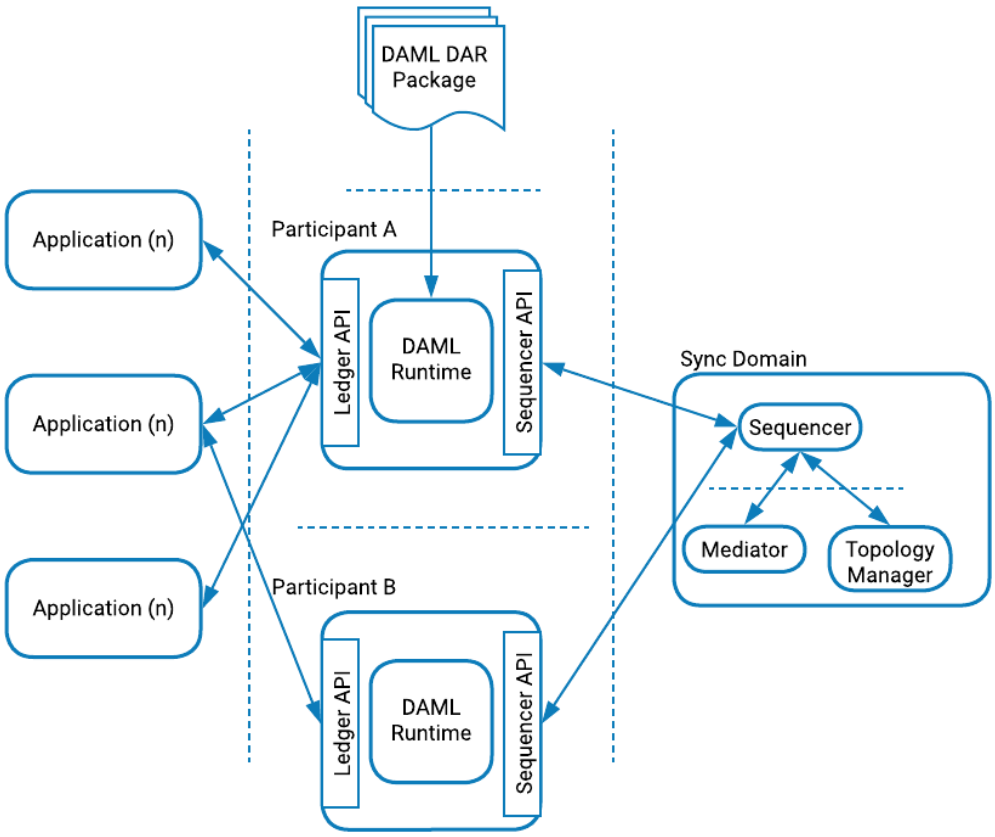

Figure 10: Trust boundaries in Canton

A Daml application includes all components that communicate through the Ledger API, including language bindings, ledger clients, Navigator, JSON-API, Auth Middleware, and Trigger Service.

A well-configured ledger exposes four areas against which a potential attack could be mounted. This section describes how using Daml mitigates risk in those areas:

- Attackers could send malicious commands to the Ledger API, attempt a denial of service (DoS) attack, or attempt to execute unauthorized commands.
- Attackers could upload malicious DAR files, either containing files that are not metadata or *.dalf, or containing Daml packages designed to do harm.
- Attackers could attempt to discover and exploit a weakness evident in a particular Daml model or the Daml application components.
- Attackers could send malicious messages via the Sequencer APIs, or attacks against the Canton protocols connecting the Canton nodes, attempting a DoS attack or damage to the ledger.

4.5.2 Attacks Against the Ledger API

In production environments, the Ledger API is typically protected with SSL/TLS and uses JWT-based authorization tokens for party access and command submission. JWT tokens would be generated by a separate IDM and the ledger is configured to trust tokens issued by this service.

The Ledger API is gRPC-based, meaning the messages are decoded from a binary format on the wire to a typed schema by an efficient gRPC infrastructure. gRPC is open source, has wide adoption and community support, and is backed by Google. Message sizes are compact and restricted.

An attack that targets the runtime environment before a valid message from the API schema is handled by Digital Asset's components. These attacks would therefore need to rely on vulnerabilities in gRPC or the hosting JVM.

Special data values like ContractId and Party are checked for special characters, and all interactions with persistence are done through frameworks that additionally protect against injection attacks on the underlying storage (PCS, ACS, or transaction log).

Denial of Service by flooding the API server with requests is a potential vulnerability, but itcan easily be mitigated with common protection mechanisms in front of the API, such as rate limiting, throttling gateways, or traffic filters.

4.5.3 Attacks via Malicious Daml Package Files

When uploaded, DAR files are unzipped, all files other than *.dalf are discarded, and the *.dalf files are decoded using standard protobuf infrastructure. All contracts on the ledger reference their associated contract code immutably using a strong cryptographic hash of the defining *.dalf file. Replacing the code associated with a contract is therefore not possible.

Alternatively, an attacker could attempt to distribute a valid, but harmful, DAML-LF package that gets executed in the Daml engine (e.g., an attempt to cause a DoS via computation in unbounded time and memory).

Recall that the Daml engine fetches data from the PCS, which comes with the same protections against injection attacks as the APIs. It then performs a pure computation using an abstract CEK machine to turn the inputs (from the transaction schema) and Daml packages (from the DAML-LF schema) into a transaction (again from the transaction schema).

Because this pure computation runs in an abstract machine inside a JVM with no low-level data types like memory arrays, the attack surface is low. The main attack vector is DoS, by performing computations unbounded in memory or time. This is currently protected by the JVM's resource management. Further work on DoS protection is planned.

A further protection that individual networks or nodes can put in place is to tightly control which packages are accepted and distributed, and checking these for potentially unbounded computations. In most enterprise settings, packages are vetted before distribution and production usage.

4.5.4 Attacks Against Daml Business Logic and Daml Application Components

In common with normal application deployment best practices, Daml-based systems should be deployed to an appropriately secured environment. Environment security is often unique to each use case environment and is outside the scope of this document.

All components and services in a Daml system can be scanned with industry-standard tools for a variety of code quality, credentials-in-code, supply chain dependencies and associated licenses and vulnerabilities, and other best practices. These may include:

- Static Analysis (SAST) for non-Daml components
- Dynamic Analysis (DAST)
- Software Composition (SCA)
- IaC and other deployment scanning
- Runtime and other Indicator of Compromise (IoC) detection

The Daml language restricts the ability to introduce vulnerabilities through memory, file and network I/O, mishandled cryptography, etc. However, it is still possible that the business logic encoded into the smart contract is vulnerable, and appropriate test plans should be developed to test the model. This can be achieved through Daml language bindings, Daml script, and other testing methodologies.

In many projects, Digital Asset does not develop the components outside of the Daml runtime, so we recommend that our partners and clients leverage such security tools and practices themselves.

4.5.5 Attacks Against Canton

Canton has the following key requirements to ensure a valid ledger:

- Privacy
- Transparency
- Consensus
- Model Conformance
- Well-authorized transactions

Privacy, **Transparency,** and **Consensus** are demonstrated in the following diagram.

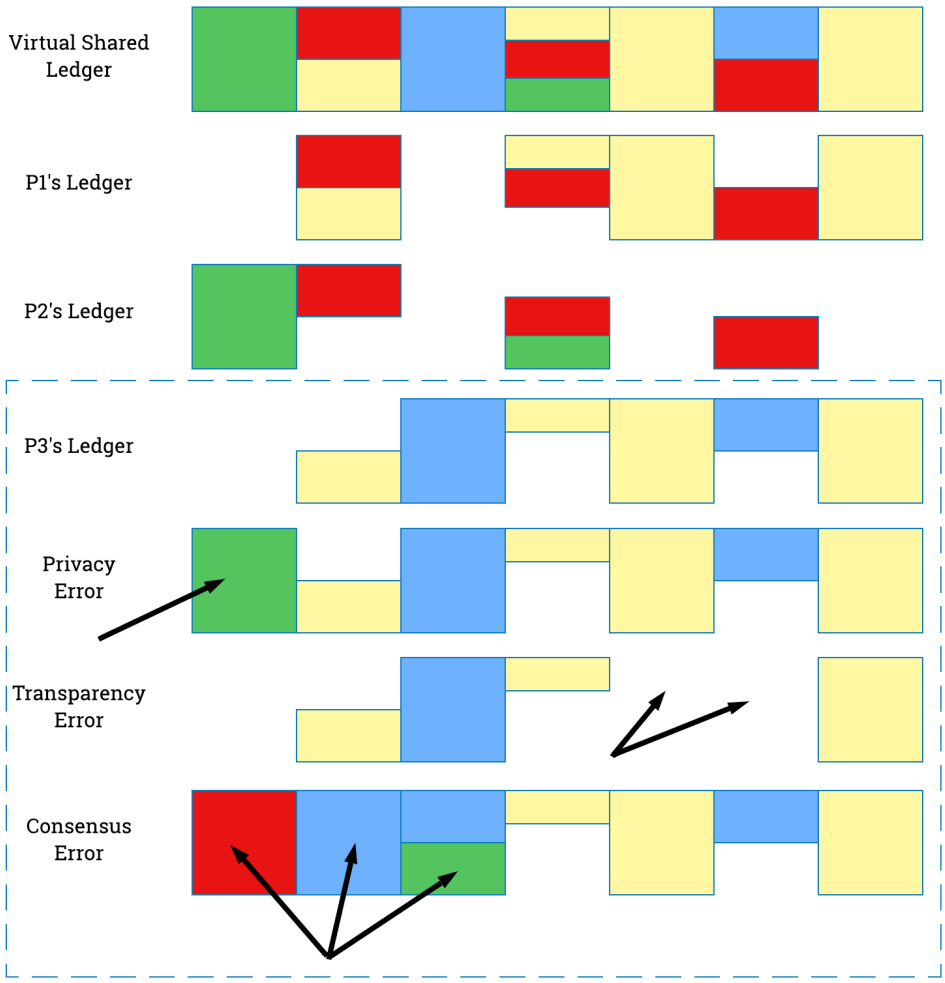

Figure 11: Privacy, Transparency and Consensus

The virtual shared ledger is depicted at the top, consisting of all contracts known to the system. Each block represents a subset of data held within the system. Each participant maintains their own private subset of the virtual ledger, depicted by P1, P2, and P3 slices. Note that each color represents a subset of shared data — e.g., green for P2 only, red for P1 & P2, blue for P3 only, yellow for P1 & P3.

If you take P3's slice of the virtual ledger, the remaining rows demonstrate failures of the ledger: data is shown to P3 that it should not see (Privacy error), data is missing from the P3's ledger that should be there (Transparency error), or data is different in P3 from the expected set for the virtual ledger (Consensus error).

- **Model Conformance** requires that all participants have validated that their transactions and contracts meet the requirements of the underlying Daml model. No transaction should be committed to the ledger if it breaks the model.

- **Well-authorized** means that all transactions have been appropriately authorized by their required authorizers (signatories, actors, etc.) as defined by the Daml model. Authorization cannot be bypassed by the replay of captured messages.
- **Consistency** is the property that exercises and fetches on inactive contracts are not allowed, i.e., contracts that have not yet been created or have already been consumed by an exercise.

These requirements are detailed in the documentation:

https://docs.daml.com/concepts/ledger-model/ledger-integrity.html

Attacks might include:

- Attacks against the Canton protocol, including transactions and topology actions, replay attacks, etc.
- Authentication and authorization attacks
- Malicious participants, including invalid transactions, withholding transactions, attempted impersonation
- For database-backed sequencers, an honest but curious synchronization domain operator
- For DLT/BFT-backed sequencers, malicious synchronization domain operators

The Canton protocol defends against these issues through:

- Deterministic language that allows revalidation of transactions by involved participants across the virtual ledger
- Total ordering of transactions, ensuring a deterministic application of these changes
- Requiring active contracts (UTXO style model) and state transitions that are defined in the Daml model
- Use of cryptography (point-to-point encrypted transactions and data structures (blinded Merkle trees)) to ensure privacy without a global state
- Authentication and authorization of access and execution of transactions through cryptography signing and encryption

4.6 Ongoing Security Testing of Daml and Canton

4.6.1 Threat Modeling and Code Security Reviews

Digital Asset has a dedicated Threat Modeling team whose focus is threat modeling our products and, using security code reviews, validating that implemented controls protect against the identified threats. As part of remediating any identified issues, security tests are added to the overall test suite and annotated as described below.

4.6.2 Ongoing Security Testing and Evidence as Part of the Build

As part of the regular development, build, and testing of Daml and Canton runtimes, Digital Asset runs an increasing set of security tests against the built products. For each release, the

tests tagged for security are extracted and provided as a “test-evidence-X.zip” file with each set of release artifacts. You can find this as part of the “Assets” section of each release.

Daml releases: https://github.com/digital-asset/daml/releases

4.6.3 Third-Party Security Audits and Penetration Testing

Digital Asset has utilized third-party security design and code auditors to review its products and associated security claims and is planning further engagements. This has covered many aspects of our products and services, including external penetration testing and infrastructure reviews of our services and code, API, and protocol audits of Daml and Canton.

Penetration tests are performed for our external services annually and security audits as deemed appropriate, for example, significant protocol changes.

4.7 Reporting Issues Discovered in the Daml or Canton

The Daml engine is thoroughly tested against a formal specification. Both code and specification are open source under the Apache 2 license and are available for independent review or formal verification.

Canton has undergone significant internal review; third-party security reviews are performed against many aspects of the runtime and protocol.

Digital Asset has a Responsible Disclosure program in place, under which security vulnerabilities can be disclosed in a secure manner to the Digital Asset security team.

5 Security of Digital Asset Services (Daml Hub, Canton Network)

Digital Asset provides a variety of SaaS services to our customers beyond our core technology products. These include:

- Daml Hub: our managed, hosted service for Daml ledgers
- Canton Network: along with Digital Asset partners, providing “Canton Transaction Synchronization” as a service via the provision of a core, open Canton synchronization domain.

5.1 Daml Hub and Security

Digital Asset provides a managed, hosted service for the creation and use of Daml ledgers. Customers are able, through the self-service administration portal, to provision ledger instances, upload Daml models and associated application code for UIs and automation, and operate their ledger, while Digital Asset manages and operates the underlying service infrastructure.

Digital Asset utilizes a public cloud environment for the core hosting services, including the use of Kubernetes, cloud load balancers, and other application and API defensive mechanisms. The mechanisms are configured to defend against security attacks, including minimally OWASP Top 10 and DDoS service floods.

Digital Asset has additionally deployed runtime defenses for the detection of potential compromise, enforcing segregating tenant access, and detection of unauthorized network, file, or process changes. Runtime and container security and hardening best practices are used to minimize potential vectors of attack on the service.

The service is tested annually via a third-party penetration test, with ongoing security testing performed more frequently by the in-house Security Team.

5.2 Canton Network

Digital Asset, along with 29 other market participants, announced the launch of the Canton Network on May 9, 2023. This network provides the decentralized infrastructure to connect independent Daml applications, with a focus on maintaining security and privacy. This is achieved through the provision of an open, globally available synchronization domain, supported by a growing group of independent organizations who have agreed to run the infrastructure needed to provide this service to the network. The global Canton Network sync domain will be provided with a BFT consensus mechanism, ensuring a secure distributed trust model. Governance of the domain and associated services will be managed via the governance tools and voting mechanisms of the Canton Network, with decisions evidenced on the ledger.

This service is undergoing rigorous testing with our partners. A schedule of activities is planned to perform security testing of many aspects of the service prior to full implementation, particularly:

- Further detailed security reviews of Daml language and Canton runtime and protocols
- Application reviews of Canton Network applications and services
- Business process governance reviews

Industry best practices for hardening and monitoring the infrastructure, testing software artifacts, and operating the production services are being applied and will be followed.

For further information, please see the dedicated Canton Network web site:

Canton Network

Or contact us for more detailed information.