Security Documentation

Date: 09.22.2023
Version: 4

Author: Rafal Winnik,
Przemysław Izdebski

The contents of this document are confidential and intended solely for the use of the recipients named.

Contents

Introduction

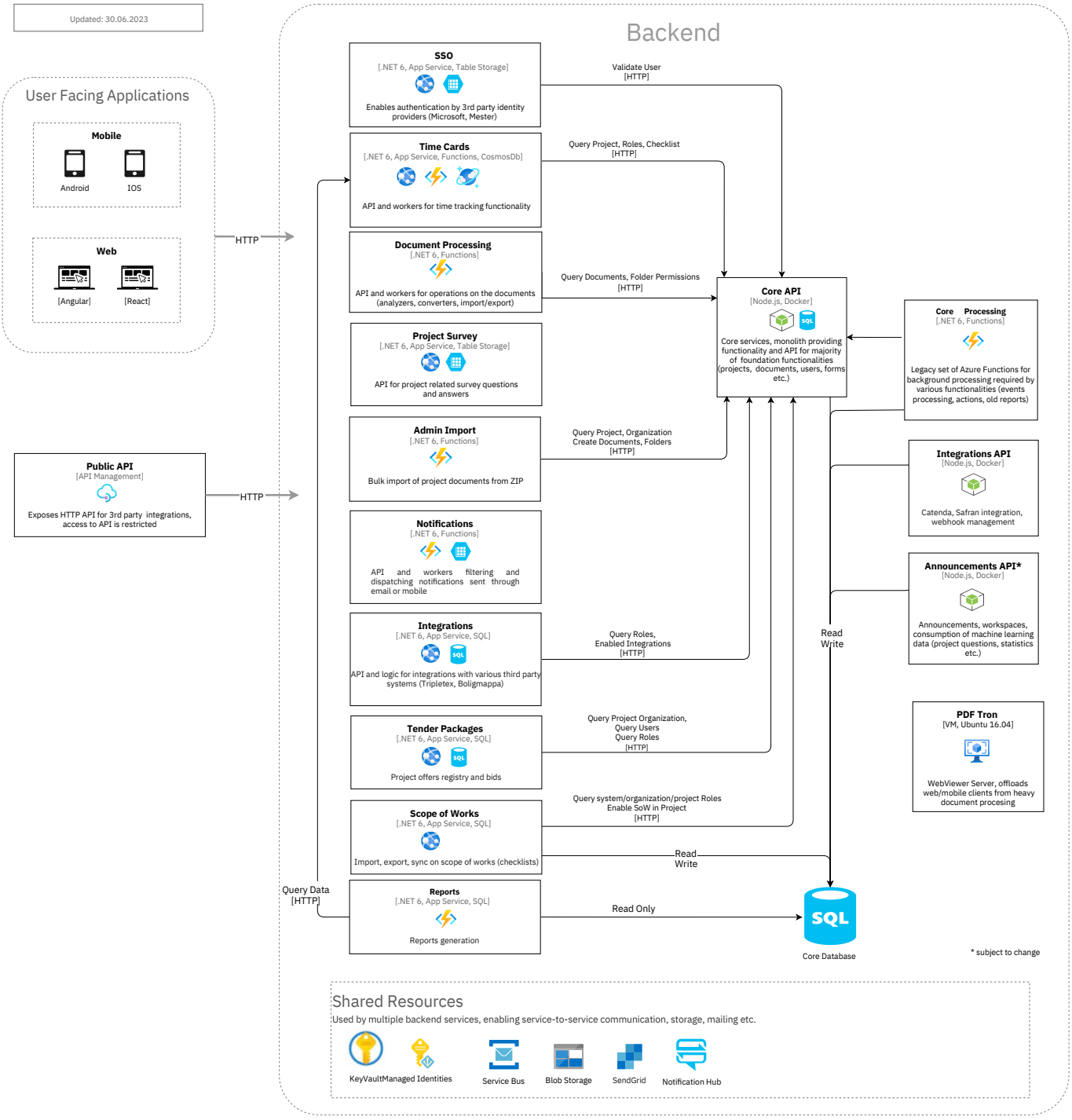
Disclaimer

The following document reflects the known state of infrastructure security as described by Fonn development team members at the moment of publication. The descriptions may become outdated because of intrinsic or extrinsic changes in functionality, requirements or 3rd party services (e.g. Azure platform).

The nature of this document is secondary, i.e. a description of an pre-existing setup. Thus, it shouldn't be regarded as a specification, guideline nor list of requirements.

Components

Infrastructure



Fonn Core Service (Azure App Service)

Main back-end service and primary entrypoint for client-facing apps (Web App, iOS App, Android App), written in NodeJS and deployed as a Docker image.

Fonn Reports Service (Azure App Service)

Secondary back-end service written in NodeJS for extracting heavyweight data for reports and deployed as a Docker image. To be retired in the future and replaced completely by Fonn .NET Reporting Service

Fonn Boligmappa Integration Service

Duplicated Fonn Core Service dedicated for Boligmappa integration solely. To be retired in late 2022 by a new .NET service for Boligmappa Integration.

Fonn Web App (Azure App Service)

Client-facing app (technically this is a hybrid application that consist of Angular and React standalone apps) which is accessed by browser. Connects directly to Fonn Core Service, dedicated microservices and Main Storage.

Fonn Android App

Mobile app which connects directly with Fonn Core Service and dedicated microservices. Receives PUSH notifications from Mobile App Notification Service.

Fonn iOS App

Mobile app which connects directly with Fonn Core Service and dedicated microservices. Receives PUSH notifications from Mobile App Notification Service.

Fonn External REST API Service V1 (Azure App Service)

Secondary back-end service which provides a public API for external services (e.g. customer services). To be retired in the future

Fonn External REST API Service V2 (Azure API Management Service)

Secondary back-end service which provides a public API for external services (e.g. customer services). Equipped with Developer Portal for browsing API documentation and testing tools.

Fonn .NET Reporting Service (.NET Azure Functions)

Dedicated self-reliant service responsible for reporting. Requesting report generation, scheduling, downloading, sending by email – all reports related logic contained in a single service. Direct data access to persistance layer with no proxies allows to skip dependencies on other services.

Fonn Admin Import Service (.NET Azure Functions)

Service providing bulk import of documents capability from zip and uploading binary of documents to Fonn from External API (Azure API Management)

Fonn .NET Timecards Service (ASP.NET Web API)

Microservice backend for Timecards module. It consists of ASP.NET Web API, Azure Function host for time triggered operations. REST API calls are made to Core API for core data like projects and users. Has its own storage (CosmosDB NoSQL database) does not connect to Core SQL database directly.

Fonn .NET Surveys Service (ASP.NET Web API)

Microservice backend for in-app surveys powered up by ASP.NET Web API. Has it's own storage (Azure Storage Tables). Makes REST API calls to Core API and puts event messages into shared Service Bus Queue

Fonn DocumentProcessing Service (.NET Azure Functions)

Dedicated Azure Function application for background processing of documents intended to supersede generic Fonn External Service V2 that used to be focused both on background processing and reporting (reporting being replaced with Fonn .NET Reporting Service)

Fonn SSO (ASP.NET Web API)

Microservice responsible for Single Sign On functionality via OAuth 2.0 (among the others Azure AD). Makes REST API calls to Core API.

Fonn Functions V2 (Azure Functions)

Serverless services written in dotNET Core for async operations, e.g. image transformations, notifications, emails.

Fonn Functions Premium (Azure Functions)

Serverless services written in dotNET Core for heavyweight async operations, e.g. generating complex Excel reports.Main Database (Azure SQL) – to be replaced by Fonn .NET Reporting Service

Main relational database. Accessed by Fonn Core Service and Fonn Functions V2.

Message Broker (Azure Service Bus)

Messaging queue for asynchronous operations (e.g. sending emails).

Event Database (Azure CosmosDB)

Document database (non-relational, NoSQL) for storing system events, containing detailed logs about business operations (e.g. created Issues, changes in Projects).

Main Storage (Azure Storage)

Cloud storage for files uploaded by end-users and auto-generated derivatives (e.g. thumbnails, BIM-based IFC files). Accessed by Fonn client apps (Web, Android, iOS) and Fonn Functions (V1, V2) using SAS URLs.

Search Index (Azure Search)

Cloud search service which ingests data from Main Database to provide high-performance search features.

Web App Notification Service (Azure SignalR)

Managed service for socket-based communication with Fonn Web App.

Mobile App Notification Service (Azure Notification Hub)

Managed PUSH notification service for Fonn mobile apps (iOS, Android).

Email Notification Service (SendGrid)

3rd party service for sending emails to end-users.

PDF Generation Service (Browserless.io)

3rd party service for generating PDF files.

Fonn PDF Service (PDF Tron Webserver)

3rd party service for browsing and marking up PDFs, MS Office documents and CAD formats.

Monitoring Services (Azure Application Insights and Firebase)

Multiple services (one for each Azure App Service, Azure Function and mobile app) for collecting/managing logs and metrics.

- Fonn Core Service (Application Insights)
- Fonn Reports Service (Application Insights)
- Fonn Web App (Application Insights)
- Fonn Functions, Fonn Functions V2, Fonn Functions Premium (Application Insights)
- Fonn External Service V1 (Application Insights)
- Fonn External Service V2 (Application Insights)
- Fonn PDF Service (Application Insights)
- Fonn Android App (Firebase)
- Fonn iOS App (Firebase)

Secrets Store (Azure KeyVault)

Secure storage for secrets/credentials/keys to external integrations.

Environments

Development

Used for previewing newly integrated features. Unstable and accessed only by the development team.

Staging

Used for QA and review of beta releases. No new features are introduced on these environments.

Production

Stable releases accessible by end-uses. Contain business-critical customer data.

Azure access categories

Subscription admins

Full access (create/edit/remove) to all services created under an Azure Subscription.

Directory admins

Can manage users and groups (i.e. Security Principals) within an Azure Active Directory.

Service admins

Contributor (RBAC) access to a specific service or service group. Can modify service properties (e.g. scale up/down) and has access to connection strings.

Service developers

Reader (RBAC) access to a specific service or service group. Can view service properties but no modify them. Doesn't have access to connection strings.

Data admins

Admin access to all database and storage services.

Data Storage

User-generated data is kept in the following data stores:

- Main Database
- Event Database
- Main Storage
- Search Index
- Monitoring Services
- Application logs
- Message Broker
- Secrets Store

Main Database

The following data is stored:

- Billing (e.g. address, tax)
- Document meta-data
- Change Requests
- Budget packages
- Submittals
- RFIs
- Issues
- User accounts
- Organization accounts
- Checklists
- Project meta-data
- Milestones
- Check-ins
- Tasks
- Time Tracks
- User devices

Event Database (CosmosDB)

- System events (every action is logged with sensitive data included) -

Main Storage (Azure Storage)

- Document files (originals and processed, e.g. thumbnails, annotated)
- Azure Table storage for some of the .NET microservices

Search Index

The following data is indexed for search purposes:

- Document meta-data
- Change Requests
- Submittals
- RFIs
- Issues
- Checklists
- Project meta-data
- Tasks

Application logs

This includes the direct output of an application, e.g. stdout.

- Fonn Core Service: errors/exceptions, app status, request meta-data.
- Fonn Reports Service: errors/exceptions, app status, request meta-data.
- External API: : errors/exceptions, app status, request meta-data.
- Functions, Functions V2 and Functions Premium: errors/exceptions, app status, request meta-data.
- Fonn PDF Service: errors/exceptions, app status, request meta-data.

Message Broker

ServiceBus stores pending messages, which may contain sensitive user information.

Secrets Store

Azure service used for storing and wrapping secrets (passwords, keys, certificates), e.g. access keys for 3rd party services.

Access Management

Main Database

The main database stores most critical and sensitive information (i.e. user accounts and generated content). It's accessed by the following services:

- Fonn Core Service
- Fonn .NET Reporting Service
- Fonn Functions V2 (data related to processing documents and sending messages)
- Fonn Functions Premium
- Search Index

All data is encrypted at rest using Microsoft's Transparent Data Encryption feature: https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption?view=sql-server-2017

The database's security is monitored using Azure's Advanced Data Security feature: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-advanced-data-security

Apart from the services deployed to Azure, developers can use their individual SQL users (which, if possible, are connected to AAD (Azure Active Directory) users or Service Principals) when debugging a service against a live database.

To provide traceability of DB operations, all access has to be made using dedicated SQL identities. Shared credentials and main root accounts are forbidden. Because of tooling limitations on macOS (IntelliJ platform IDE), not all developers can use AAD for authentication (individual SQL users are used instead for direct DB inspections). If a non-admin requires to connect to the production DB directly, he requests temporary access to his individual SQL user, which may be granted by the DB Admin.

Access to the Azure SQL database is possible in the following ways:

- For web apps hosted on Azure: using Azure Active Directory managed identities (authentication is resolved automatically per-app by Azure and hidden from users) or dedicated Service Principal accounts (if performance is a big factor).
- For other Azure services (without support for managed identities): using dedicated Service Principal accounts with credentials stored as connection strings OR plain SQL users
- For developers debugging on local environments: using (in order of preference:
  - personal Azure Active Directory user accounts
  - individual Service Principal accounts
- For database server admins: using their personal Azure Active Directory user account

Access to the production database is limited to the following origins:

- Azure services
- office IP address1 (31.186.217.66, 83.144.110.138)
- office IP address2 (95.158.65.216)
- External VPN 1 (143.244.32.166)
- External VPN 2 (143.244.32.167)
- Remote connections must be routed through above addresses via VPN.
- Using the built-in query editor in Azure Portal

General SQL security rules

1. Do not use main SQL root accounts
2. Do not use shared credentials across users, apps and environments (one consumer, one account)
3. Always assign the lowest required permissions. No permissions is the default.
4. If possible, only use AAD (Azure Active Directory) identities for authentication. Otherwise revert to individual Service Principals for access. If AAD authentication is not supported, revert to dedicated SQL users.

Access from Azure resources

If possible, instead of using connection strings with credentials, an app authenticates to Azure SQL using Azure Active Directory managed identities. This increases security by removing any credentials from app configurations.

If not possible, individual Service Principal accounts should be used instead. Credentials must be stored as Connection Strings in app settings.

One scenario for using plain SQL authentication instead of Azure AD authentication, is for the Fonn Core Service, which takes a tremendous performance hit when using Azure AD.

Access from local environment (e.g. developer's computer)

For local access, only use dedicated, personal SQL users. If possible, these users should be authenticated against AAD identities (user accounts or Service Principals). DO NOT USE SHARED CREDENTIALS.

Dynamic Data Masking

Access to sensitive data in SQL Dabatase is limited to application identities and main administrator. This help with data governance compliance in regards to privacy and industry regulations.

SQL administration as root

For SQL administration tasks requiring root permissions (e.g. creating SQL users and managing access) use the Active Directory admin role. DO NOT USE the root account directly.

1. A member of the Data Admins assigns an AAD user as Active Directory admin in the SQL server Azure resource.
2. The AAD user can now use his account for authentication.
3. Perform administration tasks (recommended: use the built-in Query Editor in Azure Portal).
4. After the task is performed, the Data Admins member removes this user from Active Directory admin.

Event Database

A shared master key is used by the following services:

- Fonn Functions V2
- Fonn Core Service

Data Admins can access the data using Azure Portal.

The database is encrypted at rest.

Main Storage

A shared key is used by the following services for read/write operations:

- Fonn Functions V2 (file operations)
- Fonn Core Service (requesting SAS, i.e. access tokens)

Limited access to files for clients is provided through SAS (Shared Access Signatures) tokens: https://docs.microsoft.com/en-us/azure/storage/common/storage-sas-overview

Storage blob data is encrypted at rest: https://docs.microsoft.com/en-us/azure/storage/common/storage-service-encryption

Search Index

Direct access to the index is only provided for the Fonn Core Service and Azure resource admins.

Azure Search data (incl. the index) is encrypted at rest and transmission: https://docs.microsoft.com/en-us/azure/search/search-security-overview

Access to the index is possible by Backend Admins through the Azure Portal.

Message Broker

Access to pending messages is possible by Backend Admins through the Azure Portal.

API Keys and Auth tokens

Client facing applications (mobile&web) JWT and API-KEY for server-to-server communication

Long living JWT authentication is implemented as a base strategy for authenticating SPA and mobile apps. Authentication token (passed via “Authorization” http header) consists of basic claims and can be easily revoked by the server since it's verified by authorization service (implemented as part of Core API) every request.

“API-Key" uniquely identifies the client application type (there are separate keys for client apps that cannot persist keys safely). Server to server communication between microservices public/external API proxy is handled securely thanks to dedicated API-Keys to be used between applications that can store keys securely. Only necessary endpoints for given server-to-server communication are available per token type.

Backup and recovery

Main Database

Automatic encrypted backups provided by MS Azure

All environments:

- 35 days PITR (point-in-time restore) backups

Production database only:

- 52 weeks of weekly LTR (long-term retention) backups
- 12 weeks of monthly LTR (long-term retention) backups

More details about Microsoft's automatic backup feature:
https://docs.microsoft.com/en-us/azure/sql-database/sql-database-automated-backups

Additional backups are also provided by geo-replication (1 instance in the West Europe region).

Recovery procedure

As the backups are provided by Azure, the recovery process is also handled using the Azure Portal.

Event Database

Backup functionality provided by Azure out-of-the-box (snapshot every 4 hours and at any point of time, only the latest 2 backups are stored). This database depends on the Storage container, but even after deletion snapshot are retained for 30 days.

More: https://docs.microsoft.com/en-us/azure/cosmos-db/online-backup-and-restore

Traceability

Traceable operations:

- Azure Subscription changes (configuration and access to Resources, Resource Groups, Subscriptions)
- Azure Active Directory changes (Users, Groups, Service Principals)
- Data operations on Main DB (no shared access keys)

Untraceable operations:

- File blob operations on Azure Storage (shared keys are used for access by Fonn Core Service and Functions/Functions V2/Functions Premiums)
- Data operations on CosmosDB (shared keys are used for access by Fonn Core Service and Functions/Functions V2/Functions Premiums)