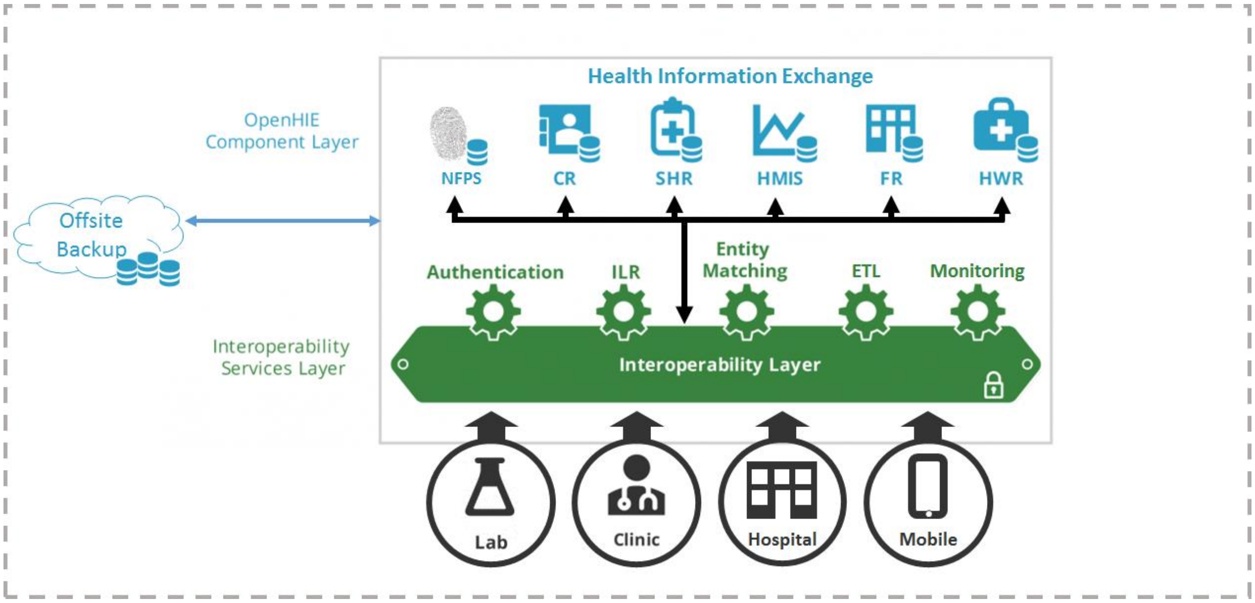
SEDISH Architecture

This document defines the hardware and software architecture of the SEDISH health information exchange.

Overview Diagram



Hardware Architecture

SEDISH requires a multi-server networked environment. This environment includes six total servers. One server is for the national fingerprint server, which requires Windows Server to operate. The remaining five servers will utilize Ubuntu 14.04 as the operating system and perform the other functions of the HIE. The hardware specification for each server can be found here. The services listed in each section includes the full technology stack running on that particular server (database, service layer, web layer)

Server #1
OS: Ubuntu 14.04
Services:
OpenHIM (IL)
OpenInfoMan (ILR)

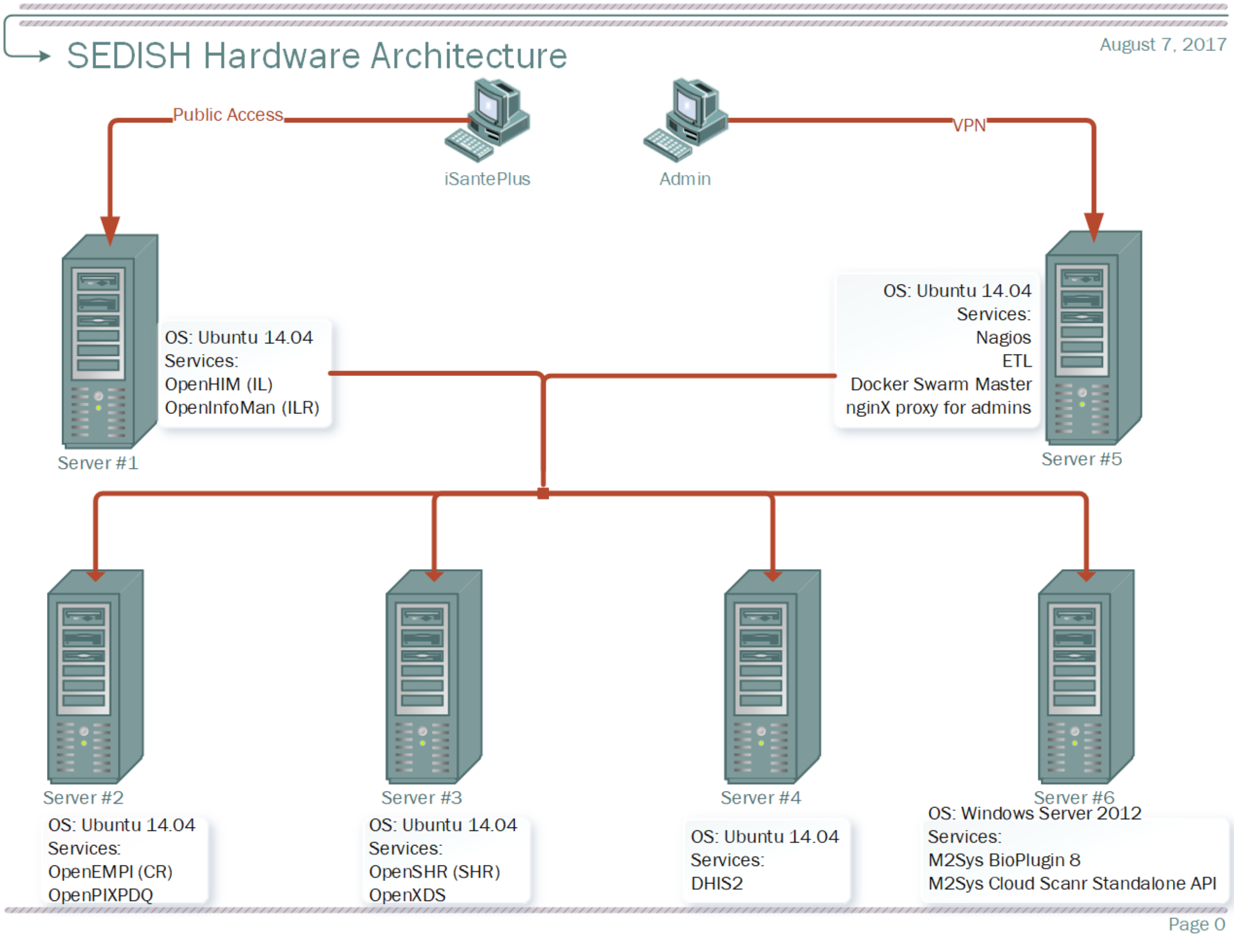
Server #2
OS: Ubuntu 14.04
Services:
OpenEMPI (CR)
OpenPIXPDQ

Server #3
OS: Ubuntu 14.04
Services:
OpenSHR (SHR)
OpenXDS

Server #4
OS: Ubuntu 14.04
Services:
DHIS2

Server #5
OS: Ubuntu 14.04
Services:
Nagios
ETL
Docker Swarm Master
nginX proxy for admins

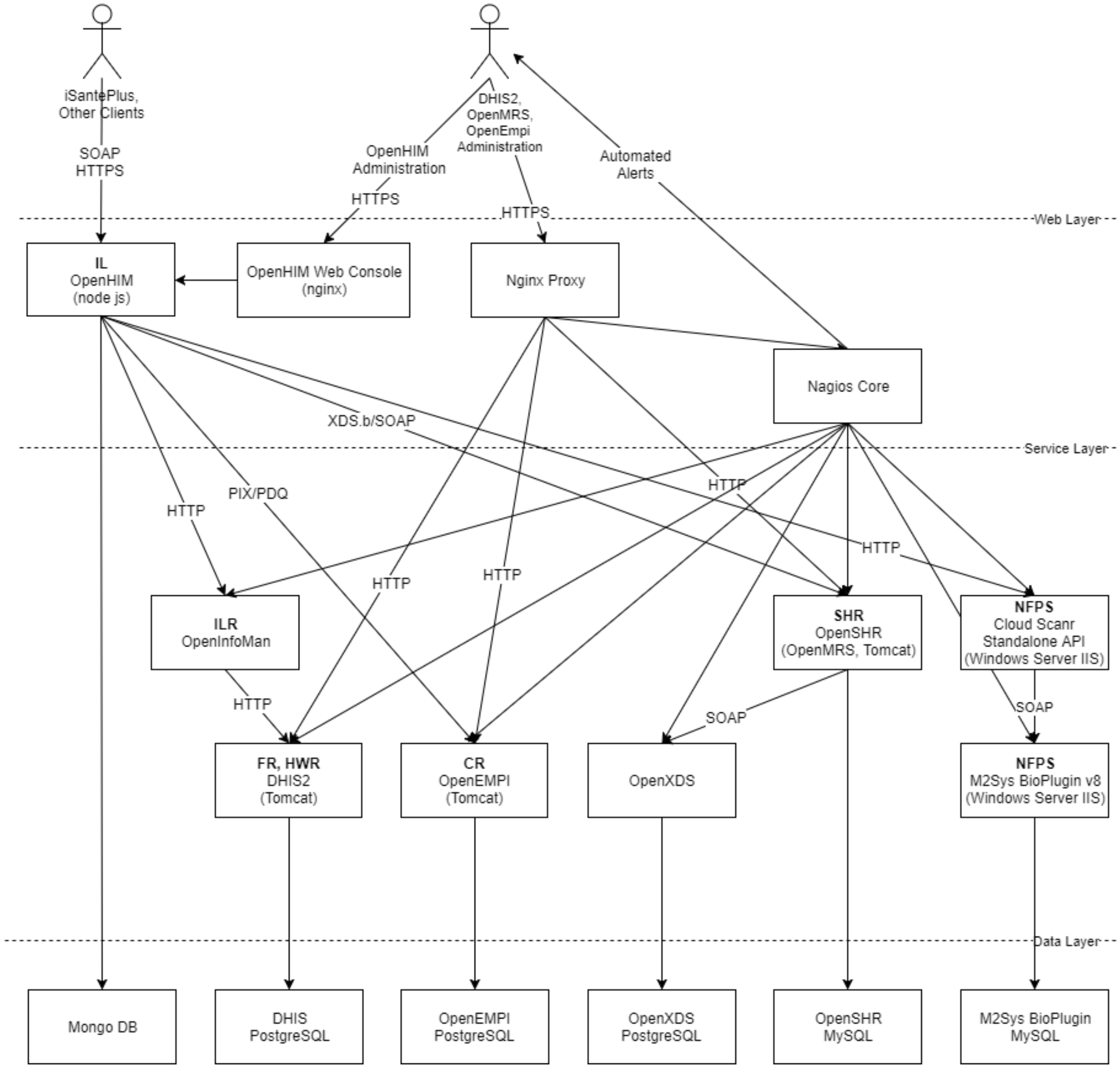
Server #6
OS: Windows Server 2012
Services:
M2Sys BioPlugin 8
M2Sys Cloud Scanr Standalone API



The software services will be managed by docker swarm for all five servers that are running Ubuntu. The Windows Server is specific for the National Fingerprint server and is not managed by Docker Swarm. Public access to OpenHIM will be through the specified ports and routed to the infrastructure systems. Administrators will be able to access the user interfaces of the infrastructure systems through a VPN and the nginX proxy that runs in a separate docker container.

Software Architecture

Note that the software services for each machine are listed in the hardware section. Docker Swarm will be used as a load balancer to pool and manage the resources across the hardware. The diagram below illustrates the architecture of the SEDISH HIE infrastructure:



It is divided into three layers - web, service and data.

Web Layer:
This the layer clients communicate with directly in order to interact with the infrastructure. Three Docker applications run in this layer:

- OpenHIM - this the Interoperability Layer of the HIE. It orchestrates workflows and is the access point for clients interacting with the HIE, such as the iSantePlus software installed in clinics. It also initiates the OpenInfoman synchronization with DHIS2. Uses MongoDB as a database. OpenHIM can be considered as part of the service layer as well as the web layer.
- OpenHIM Web Console - a web console for administering OpenHIM. Accessed by administrators to configure the HIM, and view a log of the transactions. Note that most

HIM configuration is managed through the OpenHIM container, the web console should primarily be used for monitoring and troubleshooting.

- Nginx Proxy - this is a container running the Nginx server, which serves as a proxy for administration consoles for three services: DHIS2, OpenSHR and OpenEMPI. Forces HTTPS on clients attempting to connect.
- Nagios Core - this is used as a system monitoring service that can automatically notify team members when there are failures in the system.

These web containers can be scaled easily if need arises - it shouldn't except for OpenHIM. OpenHIM can be easily clustered using a load balancer (such as Docker Swarm) : http://openhim.readthedocs.io/en/latest/how-to/how-to-setup-a-basic-cluster.html

Service Layer:
The service layer provides the interfaces for the infrastructure components of the HIE.

- OpenInfoman (ILR) - the interlinked registry, synchronizes provider and facility data with DHIS2 called on regular intervals by OpenHIM. It is used by OpenHIM for provider/facility data during orchestration.
- DHIS2 (FR, HWR) - a DHIS2 server running on a Tomcat server. Serves as a registry for facilities and health providers. Uses PostgreSQL as a database.
- OpenEMPI (CR) - the client (patient) registry. Running on Apache Tomcat, uses PostgreSQL as the database. Accepts from PIX and PDQ requests on ports 3700 and 3600 (MLLP protocol). OpenHIM exposes a PIX/PDQ channel that converts between HTTP requests to MLLP - this channel is used for PDQ patient queries and PIX patient creation.
- OpenSHR (SHR) - built on top of OpenMRS 1 by addition of SHR modules. Connects to OpenXDS and allows preservation and retrieval of clinical documents. Uses MySQL as the regular OpenMRS database.
- OpenXDS - the OpenXDS system (Jembi's fork of it). Uses a PostgreSQL database to persist clinical documents.
- Cloud Scanr Standalone API - (NFPS) a REST API to interact with the M2Sys BioPlugin v8 server. Uses SOAP to interact with the BioPlugin server.
- M2Sys BioPlugin v8 server - The national level fingerprint matching server that stores fingerprints for all individuals in the country.

Services do not have state and as such can be replicated using Docker Swarm. (Note: orientDb data in OpenEMPI has to be moved).

OpenMRS - can be replicated, cannot be clustered currently.

Data Layer:

- MongoDB - non-relational database used by OpenHIM.
- DHIS PostgreSQL - PostgreSQL instance used by OpenDHIS.
- OpenXDS PostgreSQL - PostgreSQL instance used by OpenXDS.
- OpenEMPI PostgreSQL - PostgreSQL instance used by OpenEMPI.

- OpenSHR MySQL - MySQL instance used by OpenSHR.
- BioPlugin MySQL - MySQL instance used by M2Sys BioPlugin v8 server

The data layer uses named data volumes for persistence of data. The data layer is crucial for backup and persistence of data. A separate physical node can be used for placement of the data layer containers - Docker Swarm placement restriction can be used to restrict placement of data containers to the data oriented server.

The data layer is the only layer that is/should be backed up through the data backup scripts.

Mapping the Software Services to Hardware

Each component utilizes Docker containers. This allows us to separate the services and perform load balancing as needed within a particular server. Each layer for a particular service will run on the same machine as the service layer, as well as the web layer for OpenHIM.