NotifySCM
Secure Content Management

Advanced Integration
High availability

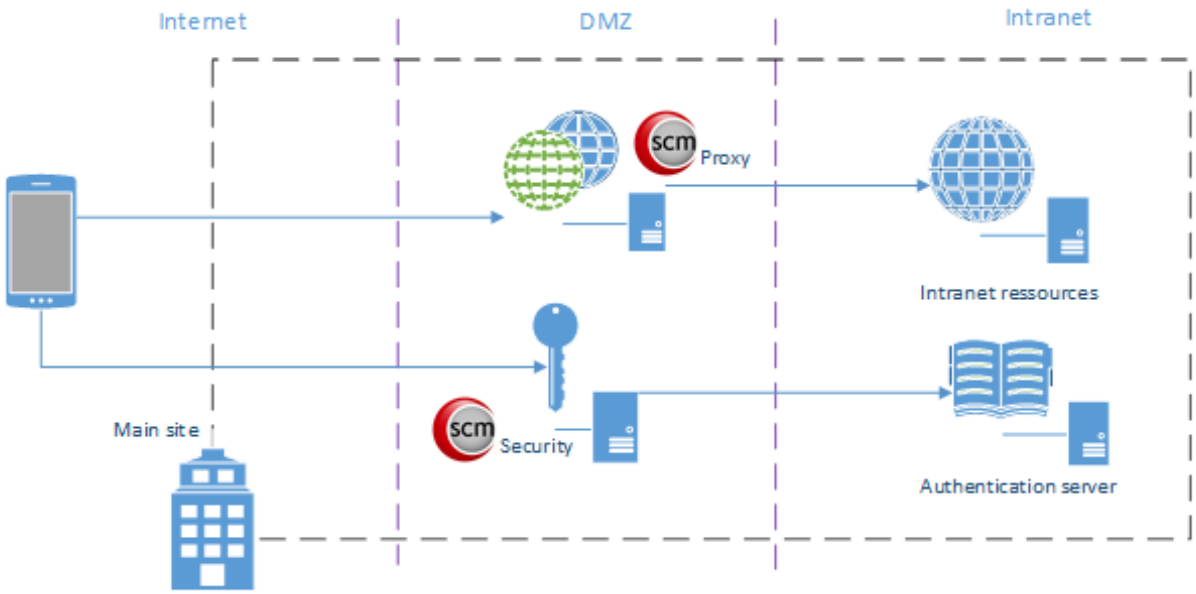
TABLE OF CONTENTS

1 OVERVIEW

This document describes the implementation of high availability setups to prevent or limit disruption of service in the NotifySCM environment.

2 PRINCIPLE

The server is composed of the following components:



- Sense Security - This component is stateful since the existing sessions are in-memory. A session contains all the information about the user, his/her credentials, and the session key that is being generated to each new session. A database is used to persist the shared keys between the apps and the server. The Security server is responsible for:
    - Synchronization and authentication of users;
    - Handling the shared keys between the clients and the server;
    - Creation and validation of sessions;
    - Handling the distribution and the update of the apps.

  To answer high availability constraints, this server, as well as its database, must be replicated.

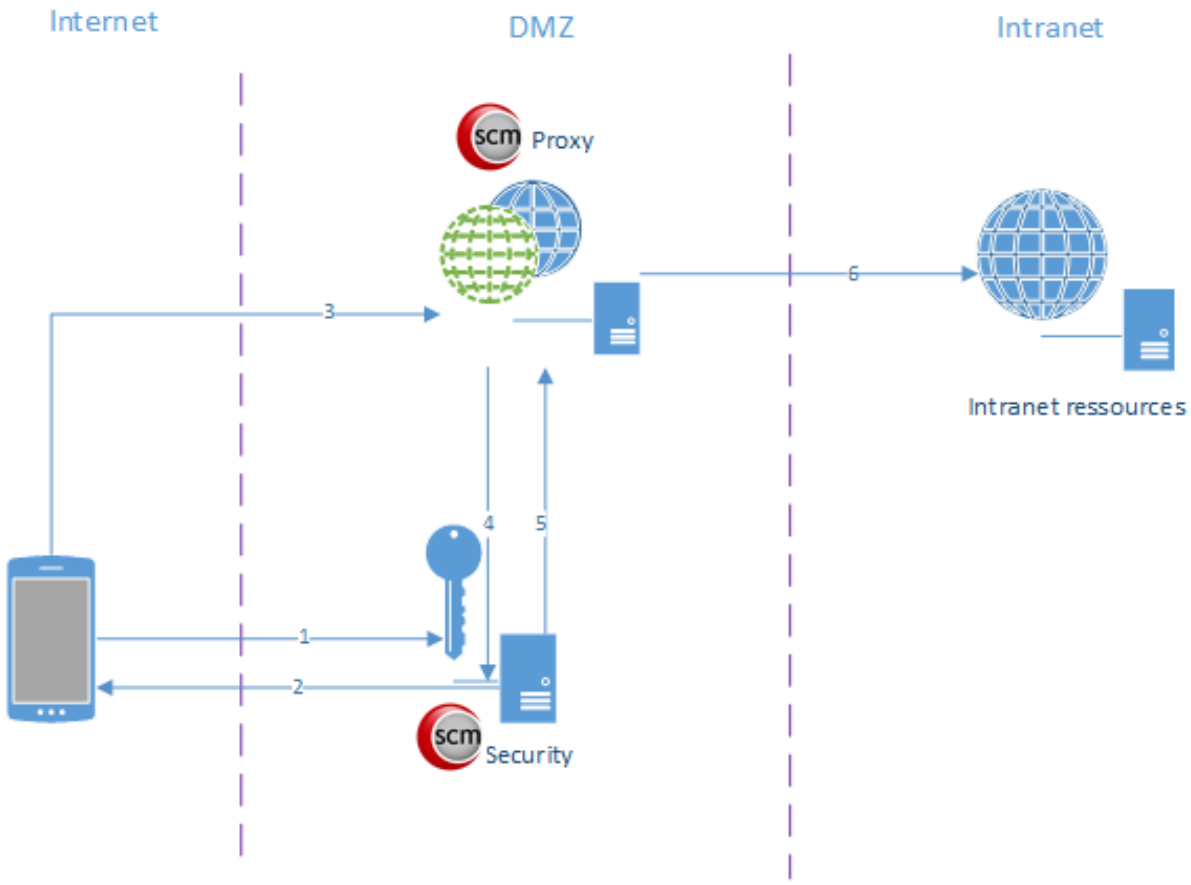
- Sense Proxy - This component is stateless. Multiple Proxy servers can be deployed per security server. A proxy server is responsible for:
    - Decrypting and encrypting inbound and outbound communications;
    - Querying the security server to validate a session;

- Validation of the app context;

To answer high availability constraints, multiple Proxy servers can be deployed per Security server in order to dispatch the processing charges.

3 WORKFLOW

Below is a schema that represents the interaction between the different components of the solution:



The process by which mobile devices gain access to backend resources is as follows:

1. The mobile device establishes a secure channel to the NotifySCM Security server.
2. The Security server creates a session and sends back the session ID to the device.
3. The device establishes a connection with the NotifySCM Proxy server and gives the previously retrieved session ID. Thereafter, all further requests go through the Proxy.
4. The Proxy server requests an authorization to the Security server according to the session ID.
5. The Security server grants access for the user according to the session ID.

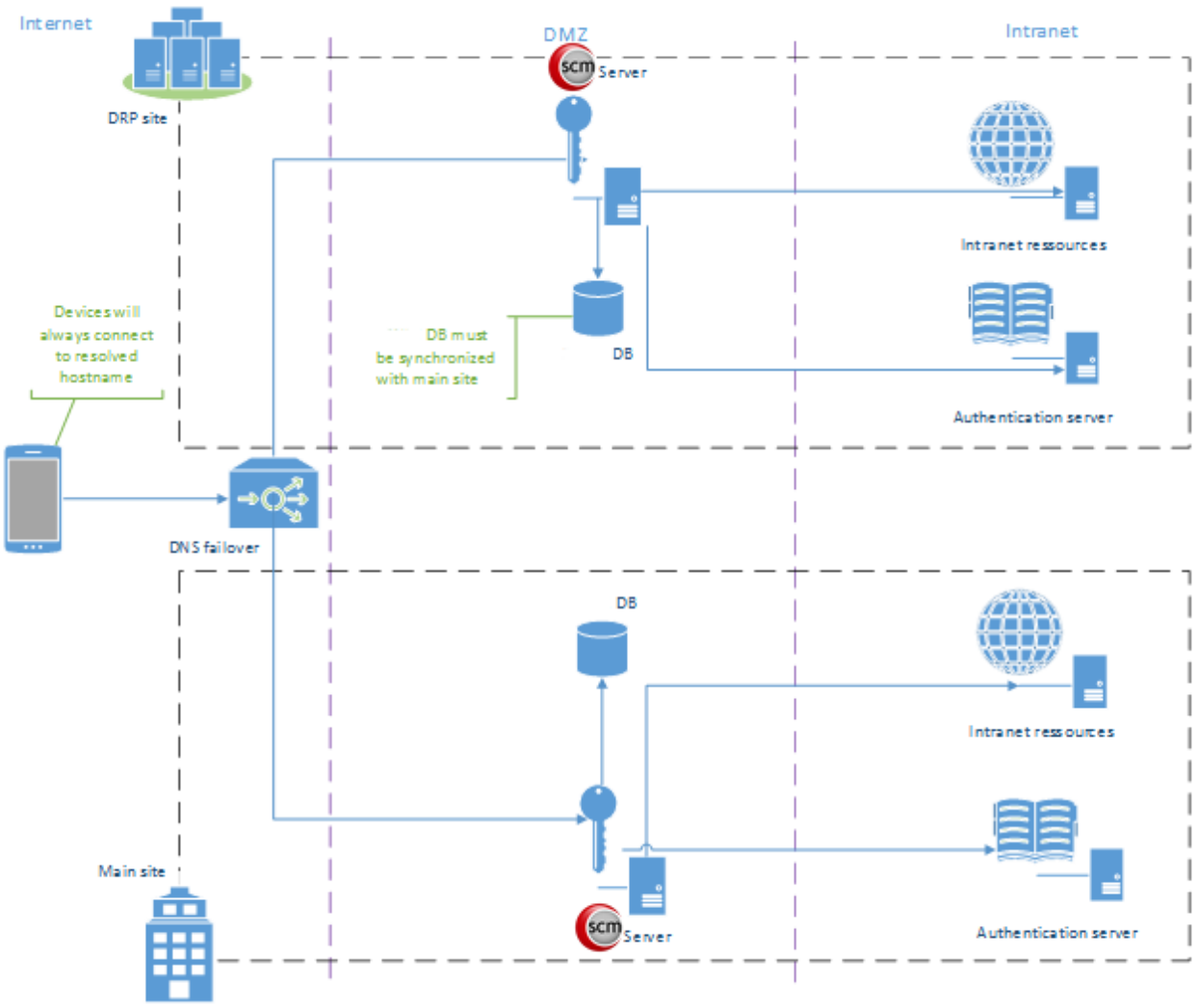
6. The Proxy decrypts and opens the access to the backend resources.

4 FAIL OVER

The security server is a critical component since its failure would temporarily prevent any Proxy from granting access to backend resources. However, a fail over mechanism can be easily implemented by replicating only the database. In the event of a service interruption on the Security server, the end user would only need to re-open a session by entering his/her credentials with the failover instance.

The replication of the database that contains the shared keys can be done on a regular basis with SQL script or by using the database in master/slave mode.

The following schema explains how the failover instance can be set up:



4.1 Data Recovery

To be able to recover quickly after a crash and to avoid users having to re-enroll, backup of the security server database is critical. A full installation of NotifySCM with database import, can take less than 15 minutes.

To backup the database, 3 methods are available.

4.1.1 VM Snapshot

It is the easiest way to backup the whole NotifySCM server, but the service may be disrupted during the Snapshot (in cases of heavy load). Also, copying a VM Snapshot to a recovery site may take time.

4.1.2 Databases Export/Import

The most efficient way is to backup the database every 5 minutes on a remote folder. In case of crash, you will be able install NotifySCM again on the recovery site and import the DB. You can follow the instructions from the **Back Up and Restore NotifySCM Databases** guide.

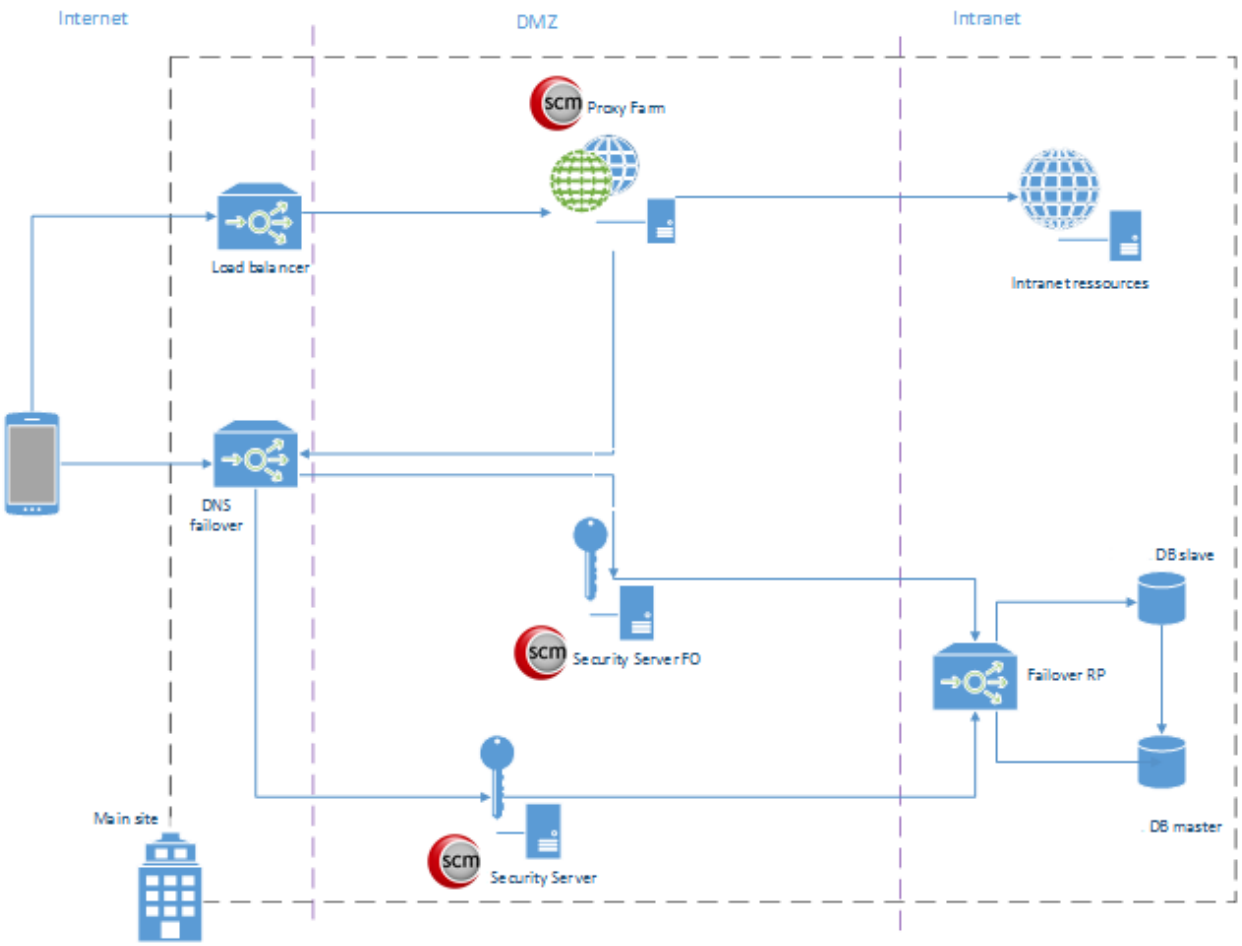
4.1.3 Live Synchronization

In a critical environment, failover must be done almost instantly. NotifySCM DB can work in master-slave mode so any change of the master will be replicated to the slave. Contact Notify Technology Corporation support if you have such a requirement.

5 LOAD BALANCING

The Proxy server relies on the Security server to grant access to backend resources. For this reason it is possible to deploy as many Proxys as needed. Most of the traffic will go through the proxy which can cause overload. A load balancer must be setup in front of a Proxy farm to guarantee load balancing and failover. If a Proxy node in the farm fails, the charge would then be dispatched among the others.

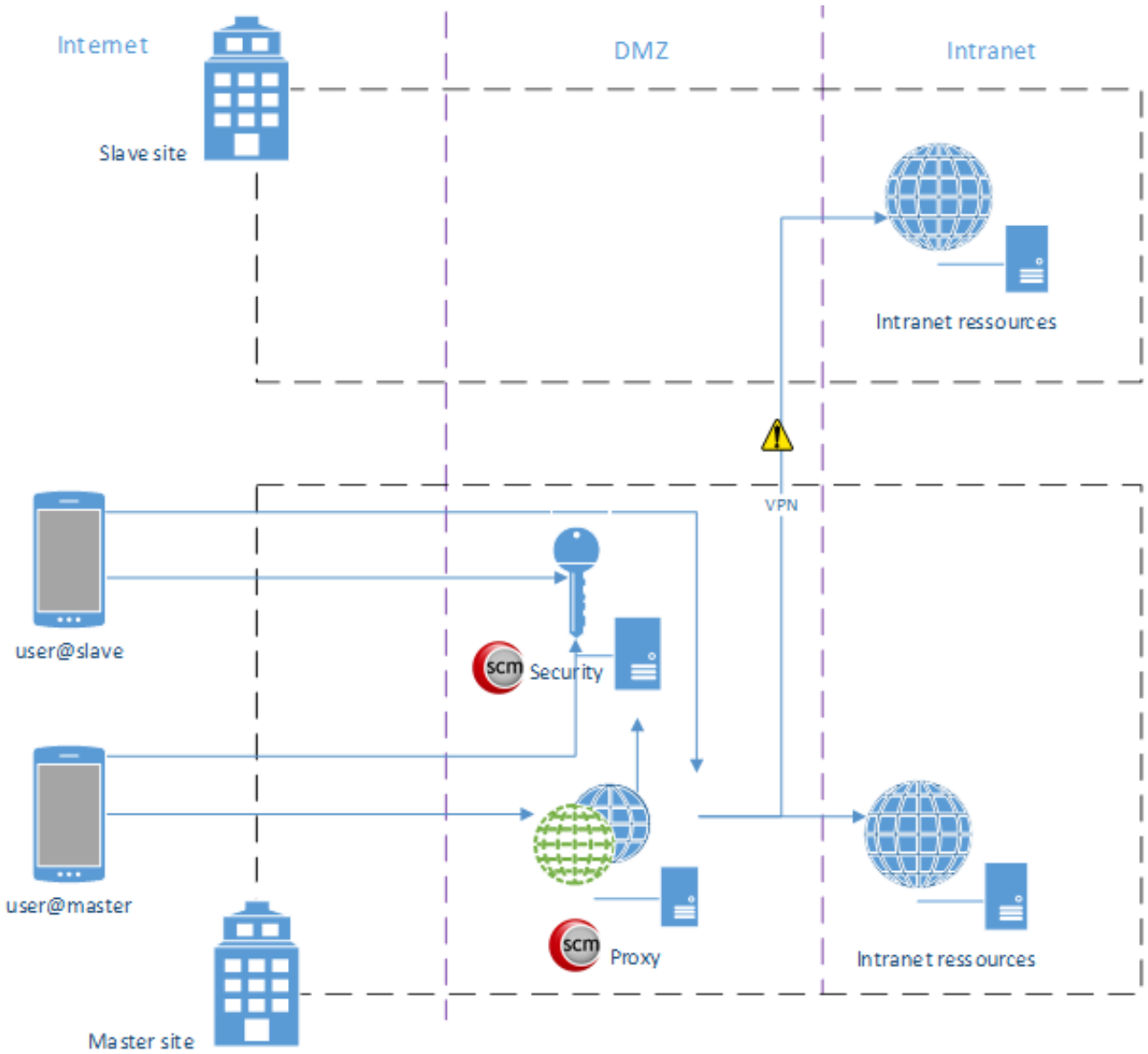
The following schema explains the architecture of both Proxy load-balancing and Security failover:



6 MULTISITE

Having a proxy farm can also prevent overloading the intranet bandwidth.

The following scheme shows a common way to give devices access to the intranet resources of a multisite company:



If you can house your failover resources in the same proximity as the production server you can avoid network equipment overload and gain access faster. Therefore, a different mobile client will be provided to allow the user to be authenticated against the main Security server, but get access to its Proxy through another URL: