Design Document

Vinz

Team May14-32
Michael Davis
Maxwell Peterson
Jacob Hummel
Zach Heilman
Eric Feldmann
Ario Xiao Qin

Advisor
Dr. Mitra

Client
Dave Tucker, WebFilings

Background

Summary

SSH (Secure Shell) is a network protocol for secure data transfer, remote server login and execution. They are widely used to manage shared resources in Linux Server when working on Cloud Computing environment. SSH uses a public/private key pair system in order to authenticate the user to the server.

The problem is that when companies or organizations have hundreds of Linux servers running, it becomes very hard to manage access to all of the servers. Current solutions consist of a number of options, including writing custom scripts to attempt to handle managing keys across an entire organization or attempting to use an LDAP system to handle authentication across an entire network. Custom scripts quickly fail when scaled to a large number of servers and LDAP and other authentication systems are very difficult to construct and maintain.

We aim to solve this problem by creating a system for managing SSH keys that will itself run in a cloud environment. The system will provide a secure way for administrators to manage SSH keys, apply access rules and group servers. Users can easily install and setup the system, and are able to manage access to logical group of servers. We believe that our system will solve this real issue for many companies that make use of cloud environments, and we will keep updating and enhancing the system beyond the duration of this course.

Target Operating Systems

Vinz itself is installed on a Linux operating system. This machine can either be a physical server within an organization or a cloud server that they operate.

Clients access Vinz with a web browser. Any operating system or web browser will work.

Functional Requirements

Users
Users are able to do the following operations after connecting to Vinz.

- Upload new SSH keys
- Delete SSH keys

Administrators
Administrators are able to do the following operations after connecting to Vinz.

- Create new users
- Add new servers
- Arrange users and servers in logical groups
- Manage the groups of users and servers, control what server/ group of servers can be accessed by which users
- View access reports and existing key usage.

Non-functional Requirements

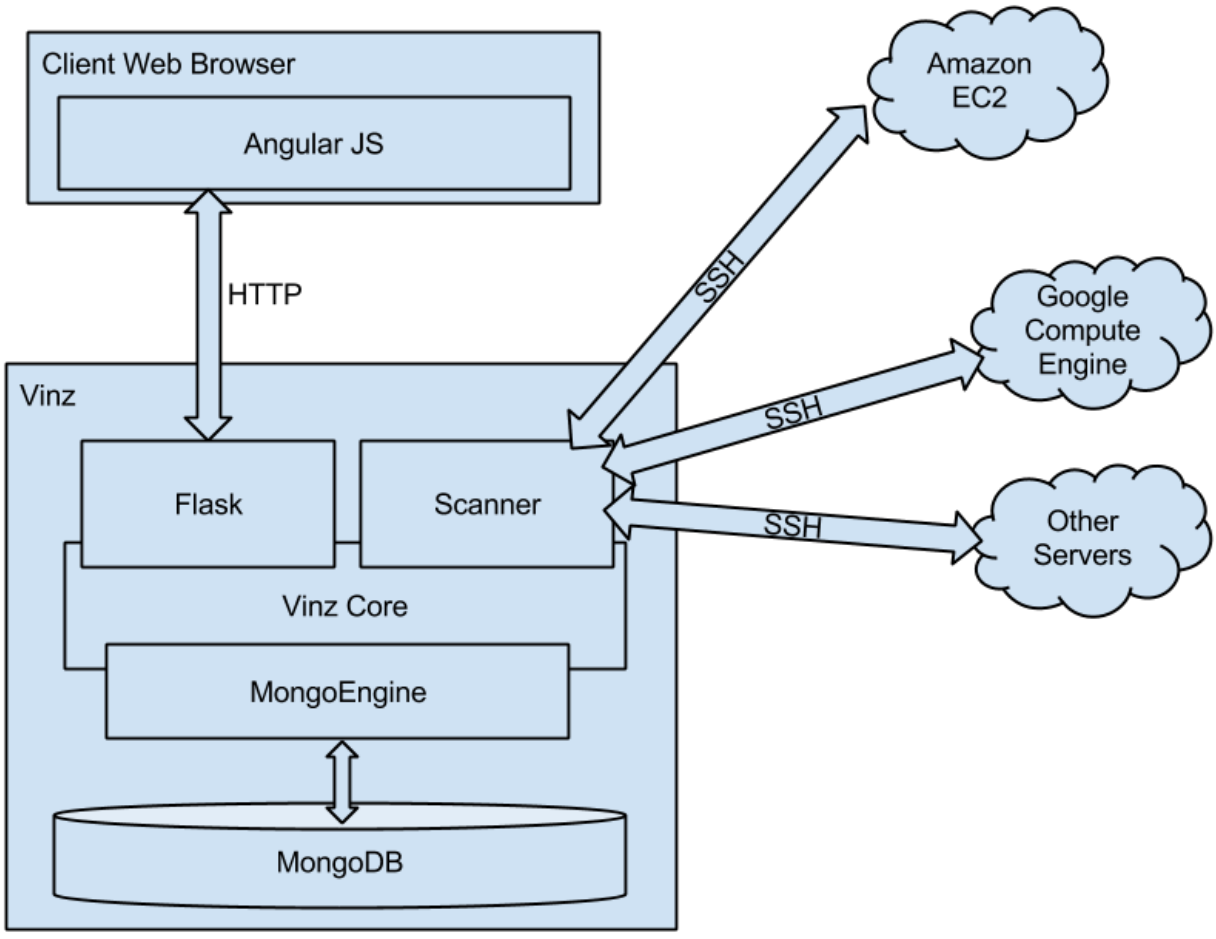
- Easy to install and use
- Account deletion and removal of access should be as quick as possible for the case of a user being fired
- Access restrictions must be enforced by scanning servers periodically
- Adding new APIs should be modular and easy to add

System Design

System Requirements

The system consists of one application running on a linux-based server. The application is responsible for serving up the frontend web application (described below) as well as responding to a number of different REST endpoints in its role as a REST API on the backend. It stores data in a MongoDB-driven database also hosted on the linux-based server. Additionally, the server must be connected to the same network as the servers it manages as a means of communicating with them. In most cases this network will be the Internet.

Architecture Diagram



Functional Decomposition

Frontend (AngularJS)

Our frontend exposes a graphical interface for users and administrators to be able to use Vinz. The interface allows users to upload their SSH public keys to the system so that Vinz can push them out to all of the servers that the user has access to.

Admins also use the frontend for a number of different tasks, including managing users on the system, managing servers that Vinz controls, add servers and users to different groups to allow for easier user and server management, and viewing audit reports.

REST API (Flask)

The Vinz server exposes a REST interface that allows access to the system and its features. This is the main interface for communicating with the Vinz Core and controlling all of the servers connected to Vinz.

Backend (Vinz Core)

The backend, which we have dubbed the “Vinz Core”, is responsible for managing the current state of the entire system. The Vinz Core utilizes MongoEngine to handle all database-related tasks and consumes the Scanner’s API in order to interact with servers that Vinz controls. Every interaction that the backend performs goes through one of it’s two APIs.

The REST API leverages this backend in order to handle all of the tasks needed. All of the frontend’s interactions occur over the REST API. Separate frontends, mobile applications for example, could easily be written using the existing API.

Scanner

The Scanner leverages Ansible in order to provide a easy, sustainable, and extendable method for Vinz to access servers that have been added to the system. The scanner provides an API to expose a number of different services to the backend to allow for managing SSH public keys on systems.

The scanner is responsible for accessing all of the systems that Vinz controls in order to verify that the SSH keys on each server represent what should actually be on the server. The scanner is automatically ran on a schedule by the Vinz Core.

Detailed Design

Component Interfaces

Frontend<-->REST API

The AngularJS-based frontend uses AJAX calls to query the Vinz server's REST API.

REST API<-->Backend (Vinz Core)

The Vinz server's REST API communicates with the Vinz Core to perform all actions on the Vinz server and on the systems Vinz manages.

Backend<-->Servers (Scanner API)

The backend Python application makes calls to the Scanner' API to add, remove, modify, and manage servers. It also uses this API when calling the Scanner to ensure that the correct SSH keys exists on each system.

Backend<-->Database (MongoEngine)

The backend uses MongoDB's ORM (MongoEngine) to perform queries against the database.

Database

Schema

The database is Non-Relational (NoSQL) and implemented with MongoDB. Each of the following underlinings represent one collection stored in the DB (logically similar to a SQL table).

Server

- id
- name
- hostname
- user_list
- group_list
- lowercase_name
- creator
- create_date
- modifying_user
- modified_date

ServerGroup

- id
- name
- server_list
- creator
- create_date
- modifying_user
- modified_date

PublicKey

- id
- owner → Reference to User
- value
- username
- key_name
- create_date
- expire_date

User

- id
- first_name
- last_name
- email
- key_list

UserGroup

- id
- name
- user_list
- lowercase_name
- creator
- create_date
- modifying_user
- modified_date

ActivityLog

- id
- actor → Reference to User
- timestamp
- object → Reference to Server, ServerGroup
- action

ScanLog

- id
- server
- timestamp
- status
- server_status
- users_expected
- actual_users
- unexpected_users

Target Audience

Vinz is targeted at any organization that uses SSH keys. The idea came to be because there are no easy to use solutions to the SSH key administration problem. Anyone who wants their SSH key distribution to be auditable, visual, and simple should be interested in this application. Vinz will be downloadable online, completely free of charge. The source code is available on GitHub for contributions and maintenance.

Implementation

The user interface is implemented using HTML, CSS, and JavaScript. To create code that is as clean and testable as possible, we made use of the Superheroic MVC JavaScript Framework by Google, AngularJS. To make things look nice (and more responsive) we used Twitter Bootstrap 3.0.

Page Descriptions

Login

When any user enters the url of the hosted instance of Vinz, they are directed to this login screen. The screen is standard, and asks that you enter a valid username and password. If valid credentials (username and password) are input, then the user will be directed to the *My Servers and Groups* homepage. If invalid credentials are given, then the user is shown the appropriate error message.

Dashboard

The dashboard functions as an overview of the entire system, containing information such as the number of users and servers managed by Vinz, and a history of the recent scans the system has conducted.

My Servers & Groups (Homepage)

The My Servers & Groups page is the landing page for the application. The center of this page will focus on showing, in a table view, the currently enrolled server and user groups. The table will show metadata about each group, and allow you to explore the group details further by selecting it.

Settings Page

Show and edit user settings for a user account:

- Profile
- Change password
- Upload public keys
- Delete public keys

User Management
The User Management page provides the admin with a way to manage users of their Vinz system. The page features a search bar and a browse view so that they can both search and browse through all the users of their system. It's important that there is a simple way to revoke access for users at any time. This page will enable them to perform a number of operations:

- Create new users
- Revoke user privileges
- Change user privileges
- Edit user settings

Server & Group Management
The Server & Group Management page provides the admin with a way to manage physical servers attached to the Vinz network, as well as groups. This screen shows a table which can toggle between physical machines and then from a larger view, their groups. A single machine can belong to multiple groups, and groups can have subgroups. The screen allows you to edit, view details of, create/delete these groups.

Account Creation
Account creation is a branch off of the User Management screen that is simply a form for creating a new user. When a new user is created, you will set up:

- Username
- Password
- Servers & Groups that the user has access to