Solution Architecture

Information Guide 1.2

Disclaimer

PACOM Systems makes no warranty of any kind with regard to this product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. PACOM Systems shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this product. This document contains proprietary information and is protected by copyright. The information contained within this document is subject to change without notice.

The PACOM website (www.pacom.com) contains the latest documentation updates. Some options, compliance claims or procedures described herein may not be supported if old versions of device firmware and/or software are used.

Copyright notices

Software license notice

Your license agreement with PACOM Systems, which is included with this product, specifies the permitted and prohibited uses of the product. It is protected by Australian and international copyright laws and international treaty obligations. Your rights to use the Software are limited by the terms stated below, and your use of the Software indicates your acceptance of these terms. If you do not agree with them, you must return, delete or destroy all copies of the Software. Your rights to use the Software terminate immediately if you violate any of the following terms:

- Any unauthorized duplication or use in whole or in part, in print, or in any other storage and retrieval system is forbidden.
- You may not reverse-engineer, disassemble, decompile, or make any attempt to discover the source code of the Software.
- You may not modify the Software in any way whatsoever.

Trademarks

Support

For product support, go to PACOM Support Portal.

Contact details

Level 6, Suite 6.01, 3 Thomas Holt Drive, Macquarie Park • NSW 2113 • Australia www.pacom.com

Contents

Overview

PACOM VIGIL CORE is an intelligent, centrally controlled network designed primarily for managing thousands of remote sites that make up a significant part of the asset base of any organization.

Core components

- A compact, low power, multi-function security control panel located at each site - the S1000 Controller. It supports alarm, access control and building management systems to enable complete site management in one panel.
- User interface devices such as keypads and readers to allow for local control of the areas within a site. These devices typically allow arming and disarming, door access, local commands and basic system configuration.
- An application platform for the management of alarms, access control permissions to areas and doors (including access audit trails) and data collection from sensors and other devices which can be connected to the control panel.

VIGIL CORE features

PACOM VIGIL CORE monitors security, access and telemetry activity, gathering data from sensors connected to the S1000 Controller located at each remote site and prioritizes events for multiple operator workstations.

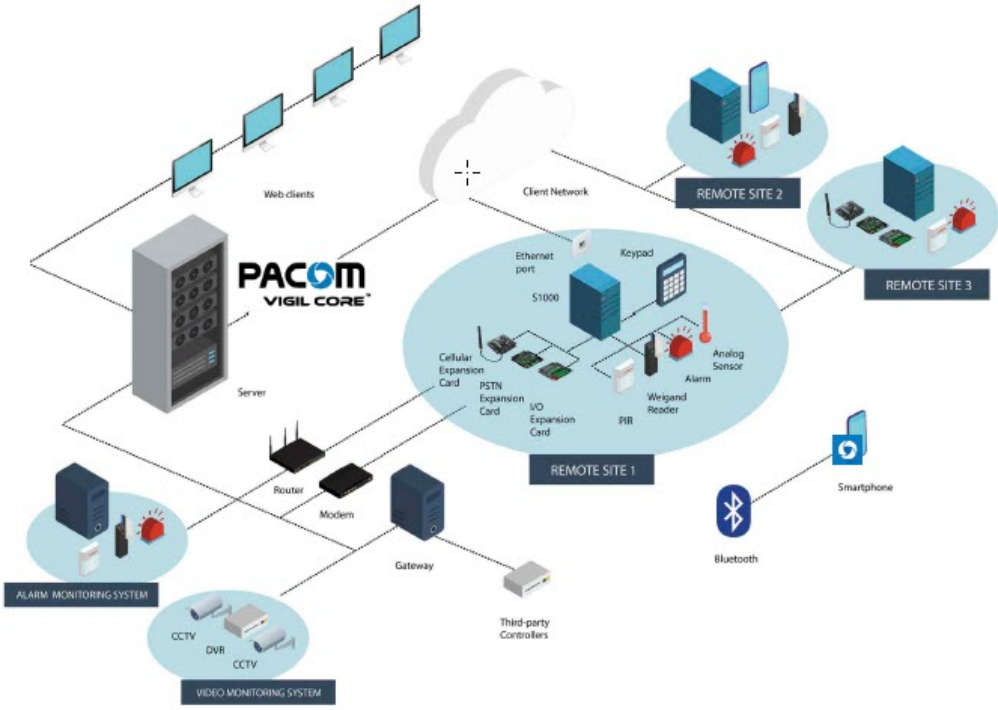
Interactive dashboards, graphical and geographical screens provide easy management and data consolidation displaying the current and historical status of a site. There is the ability to drill down to view the current status of the system configuration at any site and the details of the recorded history of activity at, and performance of, that site.

PACOM VIGIL CORE provides centralized management of site access rights. This includes control of door locks and the alarm panel using smart credentials or PIN codes assigned to users. The Platform communicates with the VIGIL CORE Manager App, enabling mobile users to efficiently manage and control the site. This includes opening doors, checking alarm statuses, and other remote management features. PACOM VIGIL CORE contains the full history of access events across sites.

- Interactive dashboard with graphical displays of overall system status
- Graphical site location and status
- Quick status views displaying open, unlocked doors
- Alarm management
- Site configuration
- Full site access audit trail history
- Built-in reporting engine
- Intuitive, multi-faceted interface
- Flexible and highly customizable
- SQL database (Microsoft SQL, PostgreSQL)
- Distributed workstations
- IoT compatible - device analytics
- AI capable - programmable logic rules
- Supports dynamic server environment - unlimited redundancy
- Horizontal scalability, unlimited connections

Solution Overview

The diagram below illustrates how PACOM VIGIL CORE, the S1000 Controller and peripheral devices can be connected.



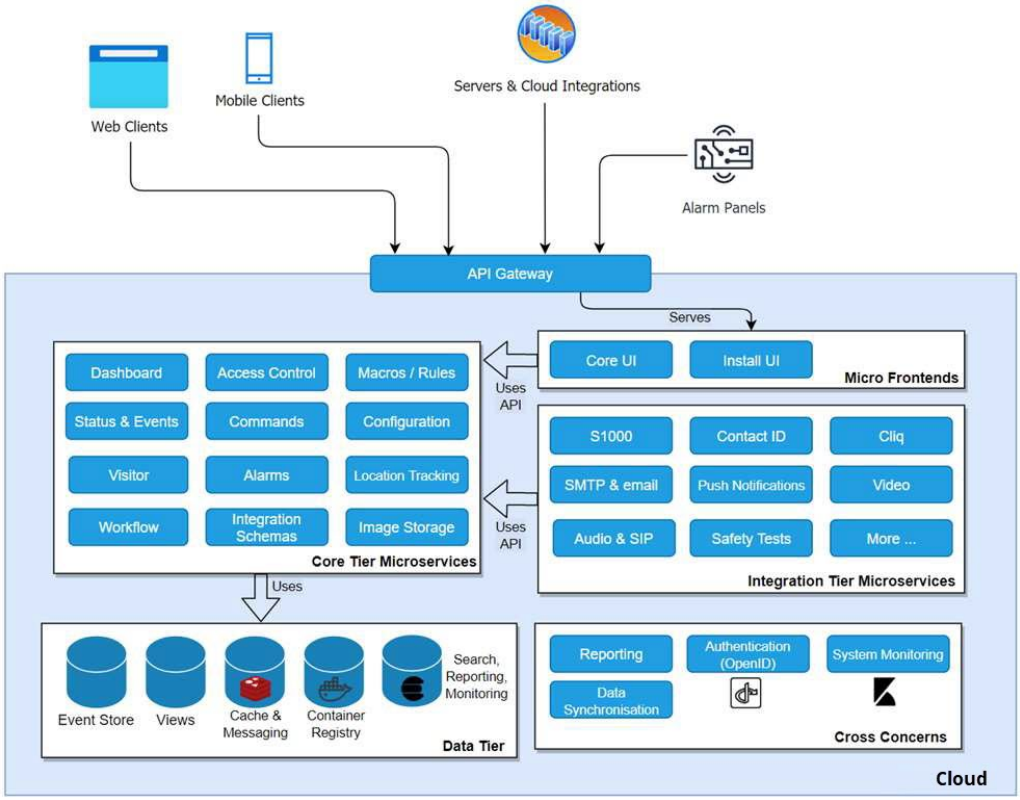
An S1000 Controller can be installed in a single site (for example, a cell tower, bank branch, ATM or a remote site). There is no limit to the number of S1000s that can be connected to VIGIL CORE. Expansion modules, keypads or other third party detectors and sensors can be connected to the S1000. The S1000 communicates with VIGIL CORE using TCP/IP over Ethernet as the default, however, 4G cellular data can be used as an alternative or as backup. There are multiple addresses programmed into the S1000 to allow it to communicate using these different paths. In addition, the S1000 can communicate to multiple VIGIL CORE servers using different IP addresses. There can be multiple instances of VIGIL CORE to provide redundancy.

PACOM VIGIL CORE is a web-based application that resides in a central location or cloud and collects all events from the S1000 Controller. In addition, it monitors the S1000s for connectivity using a heartbeat technique. If an S1000 fails to respond within a certain time, then it is marked as offline, triggering an event for operators to take the appropriate action. All events sent from the S1000 are collected by VIGIL CORE, prioritized and delivered to operators for action.

The S1000 also supports third party reporting protocols that can be used as a disaster backup, whereby they report events to the disaster system, while still maintaining the events in local memory for when the S1000 reconnects to VIGIL CORE.

VIGIL CORE connects to third party monitoring systems using Contact ID or SIA protocols.

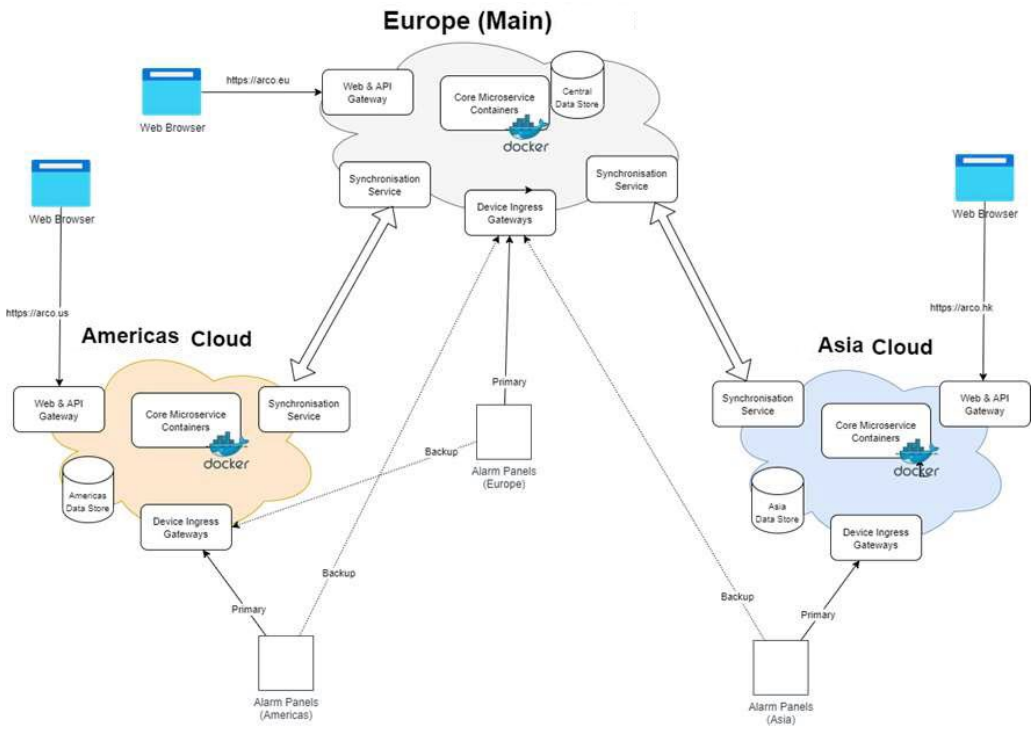
Network View



The above diagram shows the components of PACOM VIGIL CORE and broadly how they relate to each other. Each component is independently deployable, and as a rule, all components can be scaled and/or be redundant. All core micro-services are capable of replicating or synchronize their data between instances.

Core services are kept separate from integration or gateway services and are located on a separate virtual encrypted network. All access to the core services is done using the API gateway - API service. This increases security, by only exposing a single hardened endpoint, NGINX, which is an industry standard micro-service gateway.

Databases are kept as core components, and as such, by default, are only accessible from core services using the virtual encrypted network. If an organization's IT infrastructure requirements dictates that database access is required from other computers then access can be granted.



Hub topology

PACOM VIGIL CORE can be deployed as a single, large scalable system on a cluster of servers or cloud resources. Optionally, it can be partitioned amongst several servers that could be physically located in different regions with each accessing only a portion of the system data.

This hub topology has a core cloud or global accessible service connected to a number of smaller regional or site services. In turn, each regional service can manage a collection of devices or child services.

Regional services and devices at the bottom of the above diagram are downstream services. Services located higher on the diagram are upstream services.

An operator working from a regional service can only work on devices and services within that region (or downstream), irrespective of their permissions. An operator working on the global system can change configuration data anywhere in the system.

Each component within the system is responsible for synchronizing data to the upstream systems.

Downstream elements register for notifications for their portion of the system.

System failover

PACOM VIGIL CORE uses standard web patterns to handle load sharing and failover.

Within a cluster, there are always several redundant micro-services of each type running (X-axis or horizontal scaling), with each able to handle any request. Since each request can be handled independently, any service can process each incoming request.

Each service runs in an independent container. The container can be deployed anywhere within a Docker swarm. For each deployment, a system designer decides how many servers are deployed in the container swarm and how many copies of a container to run. The swarm management system distributes the servers / containers amongst the swarm servers and ensures that at least one copy of each container runs on each server.

For deployments in the Azure of AWS clouds, the containers run and are distributed within the cloud. Again, the system designer decides how many containers to deploy, leaving the distribution to swarm management. The software forces at least 2 instances of each service to be present.

For a cluster to cluster synchronization, the same mechanism works if a single server or server instance fails. If the network fails completely between a cluster and its parent cluster, both clusters remain offline to each other, and they must synchronize their data when the network is restored.

Device failover

The following rules apply to the S1000 Controller and associated devices:

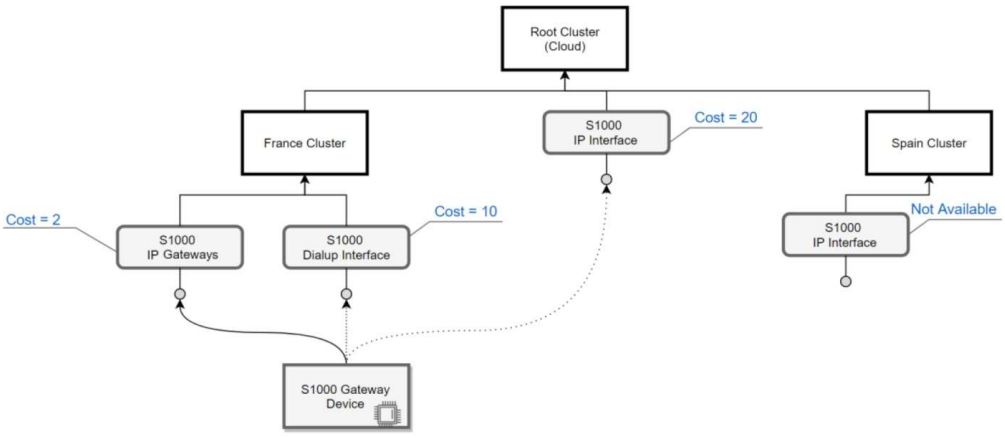
- Each device gateway configured on a cluster has a communication cost.

  Each device uses the list of available interfaces for it to work out how to communicate with the system at the minimum cost. If one device gateway fails or becomes unavailable due to network conditions, the device tries the next one on the list.

- When a device first connects to a device gateway, it registers for notifications, so that this device gateway point will be responsible for notifying this device. On fallback to another lower priority server, the device must unsubscribe from these notifications and subscribe at the new device gateway.

Many third party devices only support basic types of failover, and for devices that communicate using web standards, there should be some level of failover specific to that device's capabilities.

Example



In the above failover example, the device is registered to the France node in the organization hierarchy. It has 3 S1000 gateways available to it.

The Spain cluster interface cannot be used as it does not belong to the organizational branch.

Interfaces will be used in the following order based on cost:

1. IP gateway on the France cluster.
2. Dial-up gateway on the France cluster.
3. IP gateway on the Root cluster.

Deployment

PACOM VIGIL CORE is deployed in a flexible virtualized container environment to allow the system to rapidly respond to service failure and periods of high system load. Monitoring system performance is key to being able to detect when a containerized service has failed, is under high load or is suffering from poor performance.

Containerization of VIGIL CORE allows the system to be upgraded or downgraded easily incurring minimum system downtime.

A central container image service is deployed which retains previous versions of VIGIL CORE for an easy downgrade if there are unforseen issues during the upgrade process. This service also enables the system to be partially upgraded so that half the services or containers are running on the newer software and half are running on the old software. This way, VIGIL CORE is run in a segregated way for a few days to ensure that everything is operating smoothly before upgrading the entire system.

Network and Infrastructure Requirements

Irrespective of whether clients are using a cloud instance or an on-premise instance of VIGIL CORE, in order for access control and intrusion detection to work, S1000 Controller needs to be installed on-premise and configured to connect to the VIGIL CORE instance.

The S1000 controllers have been designed with a default security profile that generally does not need changing.

However, the following recommended guidelines should be taken into account when installing and configuring S1000 Controllers:

- The client should provide the connectivity between the S1000 controllers and VIGIL CORE.
- To secure this connection, it is recommended that firewall is configured such that only port 7016 for inbound traffic to VIGIL CORE and port 7016 for outbound traffic from S1000 is open.
- VIGIL CORE gateway address should be checked to ensure it is configured correctly within the S1000.
- S1000 controllers should be installed in enclosures that can be locked.
- Cabling from the S1000 to the general network infrastructure are securely installed as per general company policies.
- Considerations should be made for redundant connections from the S1000 Controller to VIGIL CORE using 4G connections or WiFi or otherwise.
- Bluetooth connectivity must be disabled after the initial configuration is complete.

- On-board DIP switch 3 is left at the default OFF position so that the S1000 continues to communicate using encryption. Non encrypted communication should be used only for development purposes.
- Any engineers who engage in installation and configuration have participated in PACOM provided training.

Data View

Data in PACOM VIGIL CORE is stored in an event store database which provides an audit trail indicating which user or device create the data, the time of the change and it also allows changes to be reverted. Data can originate from sensors on a device (temperature, humidity, when a door was opened), changes from a user (a device was changed) or changes from a third party system (an alarm was actioned or a command was sent).

All data within the system is partitioned as a tree-based hierarchy. This is a way to:

- manage permissions within the system.

  When permissions are granted at the parent node, child nodes inherit the permissions.
- partition the system, so that a company or agent of that company can only view devices within their scope.

In this way, system permissions and views are organized in a flexible manner. Depending on requirements, the system can be organized based on:

- Organization structures - the company that a user or device belongs to and the division within that company.
- Regional structures - the device or user may be within a specific region.

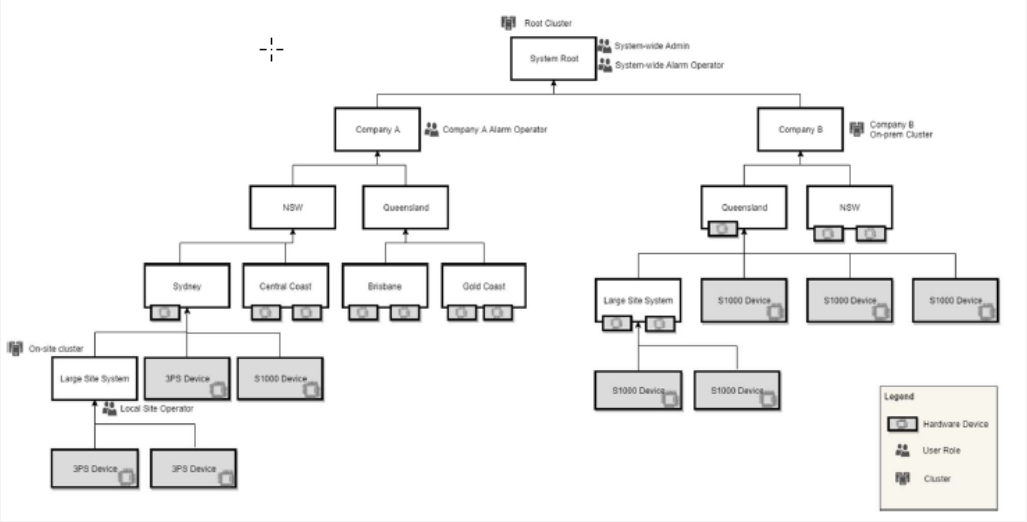
Organizational Hierarchy

The following can be located within the organizational hierarchy:

Users

A user has permissions to access groups which define the scope of the system that is accessible to them. When users are given permission to view an organization node, they automatically receive permission to view any child organization nodes.

Example:



If a user has permission for the Company A/NSW group as shown in the illustration, they are able to view devices and clusters that are registered (or scoped) to any child nodes.

This is the standard way of configuring operator role permissions and scoping an operator's access rights.

Clusters

Clusters are server resources that have a specified scope which defines the level of the organization scope structure that they run at and the data that is stored.

There must always be a cluster configured at the root node which contains the data for the whole system.

Clusters that are configured at lower branches of the organization scope structure are sub-servers that only see a limited set of data within the system.

Clusters at root scope and higher in the organization structure diagram are called upstream services, and clusters at lower scopes are downstream services.

Generally, data created at downstream services are synchronized upstream all the way to the root node, and any clusters that it passes through on the way to the root also contain this information.

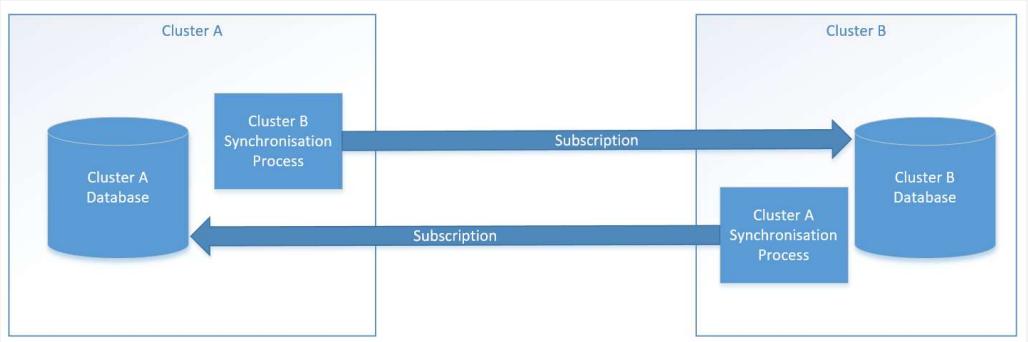
There does not need to be a cluster defined at each node in the organization structure, but each cluster will only synchronize data upstream to the next single cluster above it.

Data synchronization

Data synchronization is bi-directional. Downstream clusters receive changes from upstream services, but these changes will be scoped to only the data visible to this service (and objects defined in the scope). Upstream clusters always receive all changes from downstream clusters.

The synchronization occurs using the subscription mechanism, so each service tracks what changes it has received from the event stores of upstream and downstream clusters. This mechanism occurs automatically. Whenever an event is stored in a cluster's event store, it contains a version, so that the cluster tracks the last version that it has stored. If a cluster is restarted, then it requests changes from the upstream database from the last change that it stored.

Example:



In the illustration, when an operator makes a change on the Company B cluster, a synchronization process running within cluster A is notified of the change and writes the change to server A. As all communications within the system are encrypted, so are the subscription and synchronization communications.

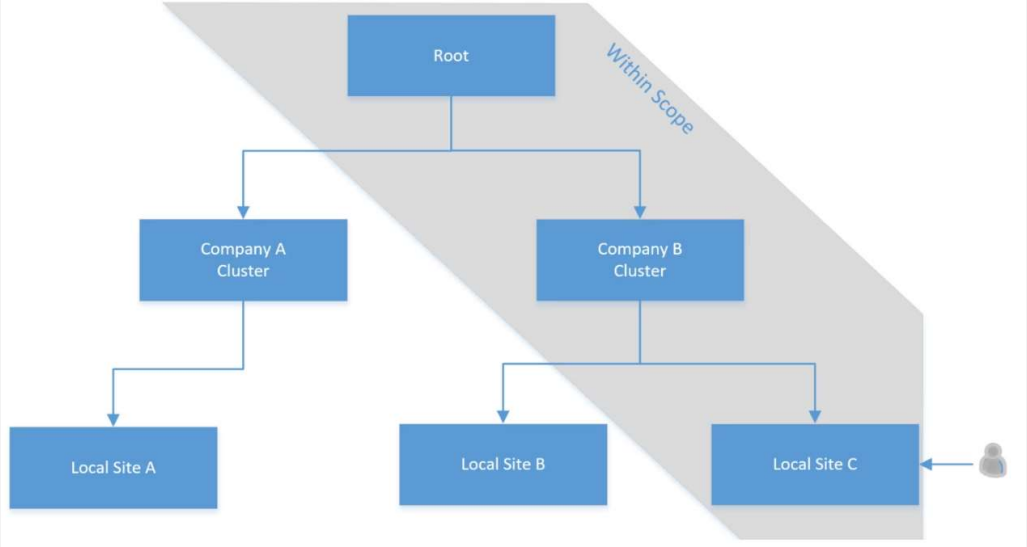
If a non-root cluster ever loses its entire database, it can recreate its data by requesting data from its upstream cluster.

Most systems only need to run a root cluster, and rely on database technologies.

Scope

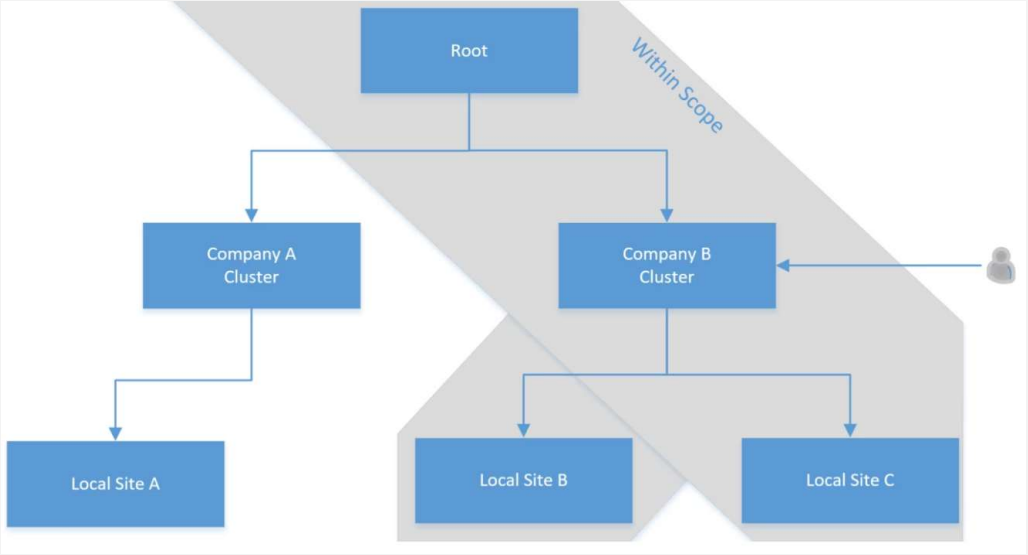
A user with permissions defined at the root node is synchronized to all clusters.

Example:



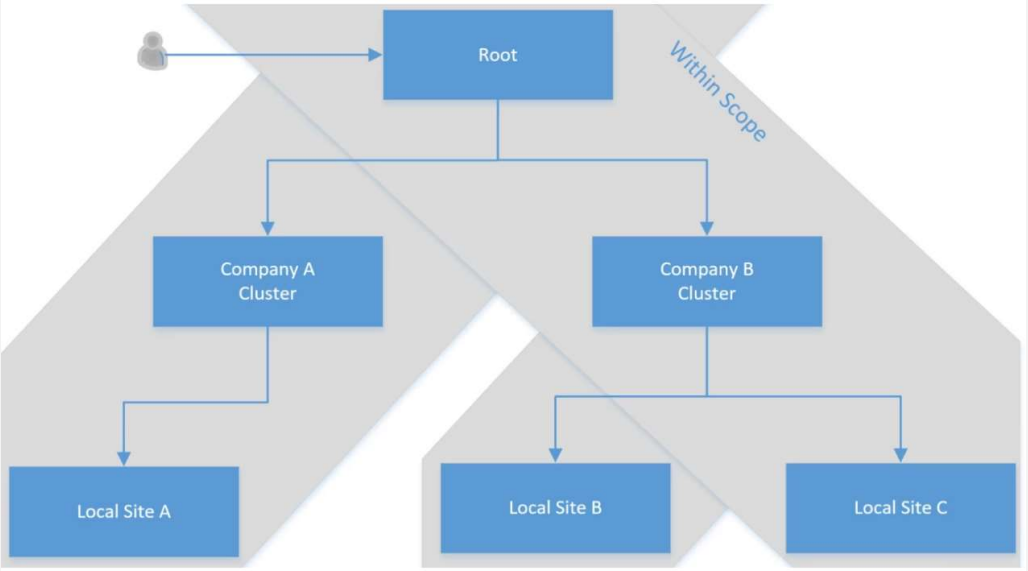
A user with rights to a single role that is scoped to Local Site C will also be visible to the Company B cluster and the Root cluster. All data is visible to the root cluster.
A user may be created from an operator connected to any of these 3 clusters, and working at the scope of any of these 3 clusters.

Example:



A user with rights to a role that is scoped to Company B cluster will be visible to Local Sites B and C, and the Root cluster. The user must be created at the Root cluster or the Company B cluster.
An operator working at Local Site C will not be able to edit the user, as they are downstream from the scope that the user was created at, but they can assign permissions to the user that relate to Local Site C.
If the user was created by an operator with a role (access permissions) defined at the Company B cluster scope (for example, a Company B admin role), then this user will be scoped to the Company B cluster and will not be visible at all on the Company A cluster or at Local Site A.

Example:



If the user was created by an operator with a role defined at the Root cluster level, then that user will be scoped to the root node and visible to the whole system.

Security View

Security is a primary concern and the aim is to have it hardened enough to be installed in highly secure environments. Connections from both users and devices use authentication and strong encryption to secure the connection and restrict resources that each are allowed to access.

All communication between components of the system is secure and encrypted using standards such as TLS and managed by certificates. Where possible, FIPS standard compliant implementations of security algorithms are used.

The core components of the system have only a single point of entry through the secure web API on port 443.

Docker container networking provides an extra layer of security so that communication between different components on the system occur on an isolated internal virtual network that is only accessible from within a Docker container, and only exposes an HTTPS endpoint using the API gateway service.

Authentication

PACOM VIGIL CORE authenticates all users, devices and integration or other services to allow for access.

Device gateway interfaces and other core system services are authenticated using shared certificates.

Clients are authenticated by a unique username and password. Clients are passed session tokens (Java web token) and every action that the user performs on a web session passes this token. Multi-factor authentication is applied when users log into a client computer.

Authentication from third party systems is allowed using the same mechanism as system services. Communication with services and clients is secured and encrypted using TLS v1.2 or v1.3.

Authorization

Authorization dictates what parts of the system are permitted to be used by specific users. Users are given a set of permissions from roles (access groups) that they are members of.

There is an optional set of access policies defined, which can be used as additional rules to access the resource, such as what times the operator is allowed to access a specific resource.

An operator who is logged in to the system that has multiple roles (admin, engineer, guard, etc) is able to select which role they are currently using, and this will define what they can view and what permissions they have.

There are 3 levels of permissions:

Permission	Description
Device	This is the set of device nodes (within the organization scope) that the user has permission to perform actions on. It is a set of more granular permissions which define which objects within the user's organization they have access to and what actions they can do on that object.

Permission	Description
Module	A set of modules that the user can access including user management, reporting, hardware administration, system administration. The user can broadly have view (read only), command (execute) and edit (write) access to each module. Some modules have extra permissions. For example, the user module may have the permission to block or suspend a user.
Organization scope	The scope defines what parts of the organization structure the role grants access to.

Design Patterns, Principles and Methodologies

The following designs, principles and methodologies are used within PACOM VIGIL CORE:

Design/Principle	Description
Anti-corruption layer	Device gateways are the external interfaces to other devices or third party systems, which contain conversion or translation code.
Bi-directional sync	This is the pattern used together with event sourcing to synchronize data with different parts of the system.
Command query response segregation (CQRS)	Command and query processing is segregated and stored in separate databases as required.
Domain driven design	The domain model is the core of each component, containing the business logic and rules. Business logic is not part of the views or data access layers.
Event sourcing	Changes to the system are stored as an event stream in most components/services. This provides an audit trail, rollback functionality and allows 2-way synchronization between components.
Micro-services	The functionality of VIGIL CORE is divided into components, each as independent as possible so that they can be treated as a separate entity.
Onion architecture	Infrastructure elements sit at the edges of the architecture, so that they can be replaced by similar products when and if needed.
Scaling (X-axis)	X-axis scaling allows several instances of a service to run concurrently, each able to handle one unit-of-work. So that changes do not arrive out of sequence, connections are sticky so that if a device or client application is communicating with a service, it remains on that service.

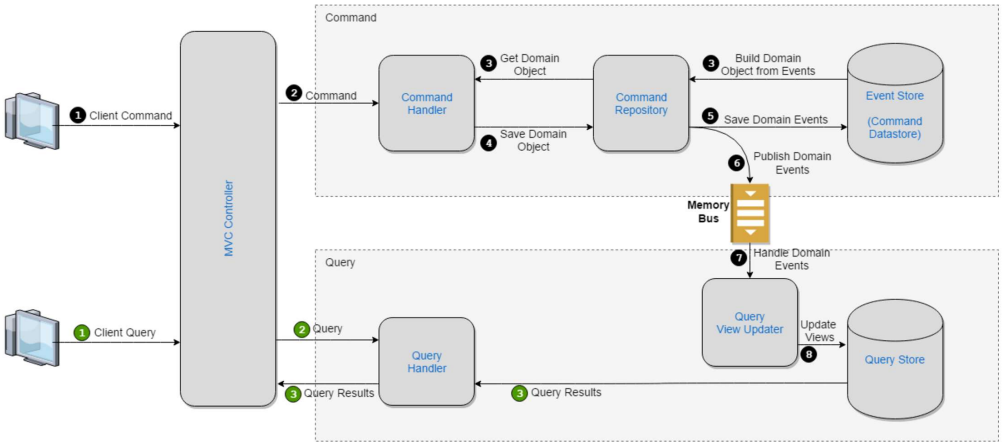
Micro-service internal structure

The CQRS pattern together with event sourcing is used for the majority of VIGIL CORE services. This adds some complexity but provides the following benefits:

- simpler scaling
- a developer can focus on query and command implementations separately with distinct performance issues and data storage decisions
- the event stream is a good synchronization pattern to follow when synchronizing data across clusters and down to devices
- very fast query response times.

The key feature of the CQRS pattern is the separate command and query datastores:

- Query store: stored as de-normalized views. It is very fast to access a separate table or view for each query.
- Command store: used in an event stream. It is separated by GUI identifiers.



Command workflow

This is the basic workflow when the WebAPI controller receives a command-side POST action.

A command is generally slower than a query operation and it may take some time to propagate throughout the system. The command datastore or event store is the source of truth for data.

1. A client sends a command, such as block credential to the ASP .NET WebAPI. Client level validation ensures that the client does not send an invalid request.
2. A command handler is created from the IoC container to handle the command
3. The command handler requests the most recent copy of the credential domain object from the event store. To build the domain object, the repository must process all the events in the event store's history for this object until it forms the most recent copy.

   For domain objects that see a lot of activity, the event list may be quite long. If this is the case, snapshotting is considered. This is where a snapshot of the domain event is taken at a certain point in time (for example, every 50 events) and stored in a table. When the domain object is rebuilt, a snapshot is taken and events are replayed only after the snapshot was taken.
4. The command handler then modifies the domain object with the appropriate command. A method is used on the domain object that corresponds to the command. The domain object itself undergoes state-based validation checks and throws exceptions if needed. The domain object is then saved in the repository.

For example, thee should be a Block() method available on the domain object to handle the business logic. This method can perform a state-based validation check, so if the credential is in the Deleted state, for instance, it would throw an InvalidOperationException("Credential cannot be blocked if it is already deleted.").

5. When the domain object is saved, it tracks any changes done to it, and it can transmit these changes as events. These events are first saved to the event store.
6. The domain events are then published in an in-memory bus so that the query side can retrieve the updated data. Once the events are published, the WebAPI controller can safely reply with a success message to the client.
7. Query side views of the data can register interest in various events so that they can update their views accordingly. They will receive a callback when an event of the specified type is received.
8. The query view updater then updates its datastore.

   There is also a start-up mechanism to recover if the service crashes before its views are updated.

Query workflow

The query workflow is a lot simpler than the command workflow, which has already done most of the work.

It is normally very fast, since its data is already de-normalized to a flat view and ideally it should not have to perform any joins with relational data.

Query data stores replicate the command data so that it is optimized for fast reads or queries.

1. A client sends a query.

   The query is validated.
2. The query handler is created by the IoC container.
3. The query handler does simple validation and requests the data from the query store.

   Ideally, it should only request data from a single table, but the results may need to be sorted or paginated.
4. The results are returned to the client.

   An exception is that a single join may be acceptable, if there is a need to provide output data to many similar streams.

Integrations View

The integrations layer of PACOM VIGIL CORE allows different devices from a variety of vendors to interact with the system. It does this in a well-defined way by providing access to the core services through a standard WebAPI with security checks.

Integration services manage the communication with third party devices and third party applications to synchronize and share data. They can also allow complicated interactions with the VIGIL CORE core services to provide additional functionality such as an analytics engine that can analyze data / events from the system in order to create and send notifications to the VIGIL CORE.

Integration layer services are kept separate from the core services from a security standpoint and access core services using credentials that restrict their access within the system to the scope of what they are intended to do. For example, a third party device may not be permitted to add a user to the system, but an integration designed to synchronize user data with a personnel management application may be allowed to.

Integration services are run in a demilitarized zone network and only have access to the core service through the standard API. There is no direct access to the core databases.

Integration services is also containerized and managed by Docker so that extra copies can be deployed if they are under high load and they can be restarted automatically on failure.

S1000 integration

The S1000 Controller is a key deliverable IoT device and has a tight, feature-rich interface into the core of VIGIL CORE. This integration is designed to be highly scalable, flexible and redundant.

Scaling is achieved by allowing a stateless connection to an S1000 integration gateway service, which exposes a standard web interface. As the request to the web interface is stateless, an S1000 can connect to any service behind a single gateway or reverse-proxy and have its request serviced by that endpoint. If there is an increased load on VIGIL CORE, then additional endpoints may be started in separate containers that the gateway services.

Up to 2 redundant cluster addresses can be assigned to the S1000, which can communicate using a variety of communication paths.

VIGIL CORE is also backwards compatible with older versions of the S1000, so that VIGIL CORE can be upgraded independently of the S1000 firmware.

All communication to the S1000 is using HTTP over TLS and authenticated in both directions using certificates. Certificates are validated and revoked using a shared certificate authority, which can be part of the existing IT infrastructure or it can be provided as part of the VIGIL CORE deployment. All communication occurs with the S1000 acting as the TCP/HTTP client on port 443 so that no site-side firewalls need to be configured.

Future deployments may also support the MQTT message queuing protocol as required for network environments where it is more appropriate.

Subscriptions, websockets and notifications

A subscription mechanism from the VIGIL CORE Gateway Service to the S1000 client can be established to deliver asynchronous command, configuration and access control change notifications.

A notification is a small message to indicate that something has changed. It may contain enough information for the S1000 to do a small command like door open or if the data cannot fit in a small message (a large configuration change or firmware update), then the S1000 retrieves the relevant data in a follow-up request. Asynchronous notifications have been implemented using the WebSocket protocol.

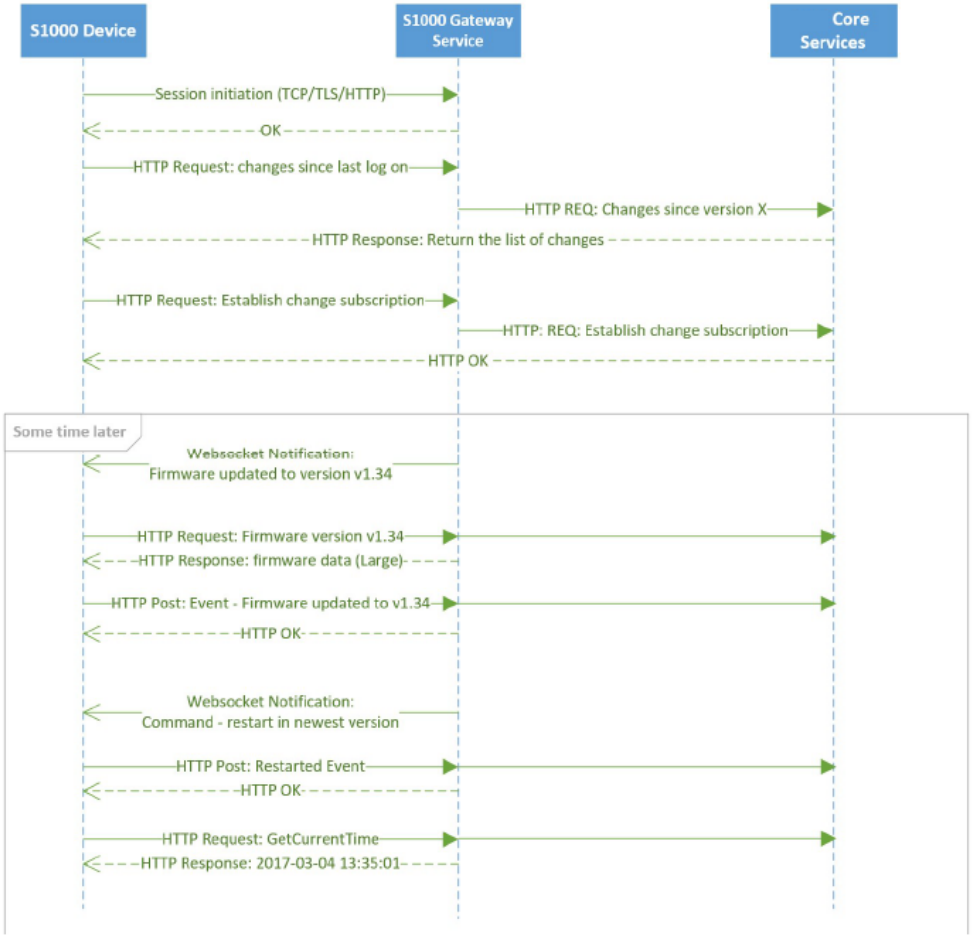
The S1000 can also choose not to act on the notification immediately and may queue it to see if there are further follow-up notifications.

> **Example:**
> For a configuration update, the S1000 may wait a few seconds to see if there are any more notifications as it is an expensive operation and may involve rebooting parts of the S1000. For a firmware update, the S1000 may want to finish processing a task before starting a large download.

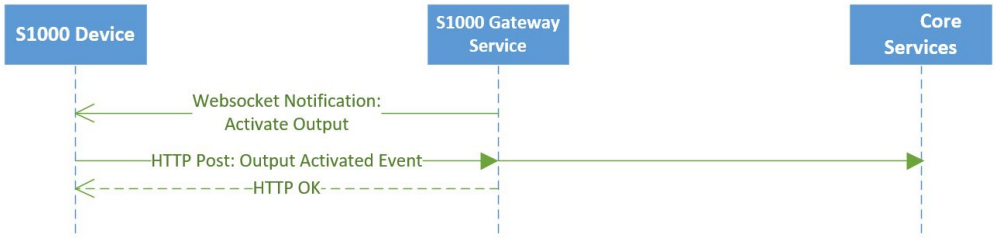
When the S1000 establishes a connection to a VIGIL CORE cluster, it registers a subscription to receive notifications from the last configuration and command notification messages that it received and processed. When the connection drops, the subscription also drops and must be re-established when the S1000 next connects.

It is the S1000's responsibility to track the last configuration, command and access control notification that it processed and when it resumes its subscription, it reports where it was last up to so that the VIGIL CORE cluster can resume sending notifications from that point onwards.

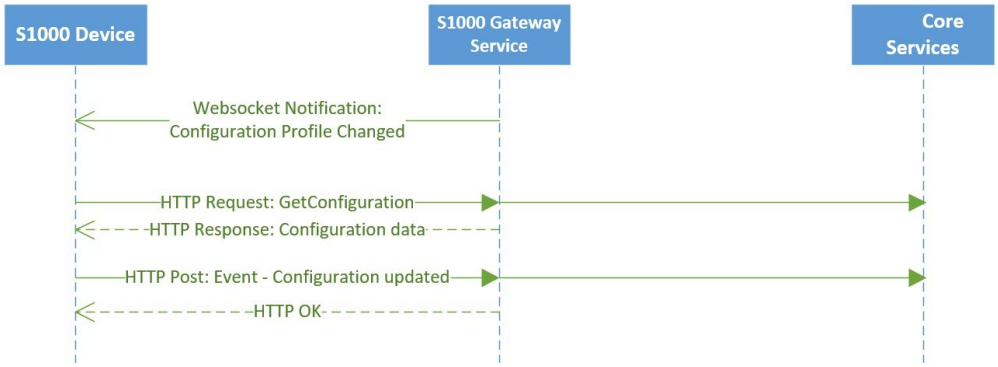
A sample interaction between the S1000 and VIGIL CORE is shown in the sequence diagram below. The communication between the gateway service and core services has been simplified.



The sequence diagram below shows VIGIL CORE issuing a command to activate an output (actuator).

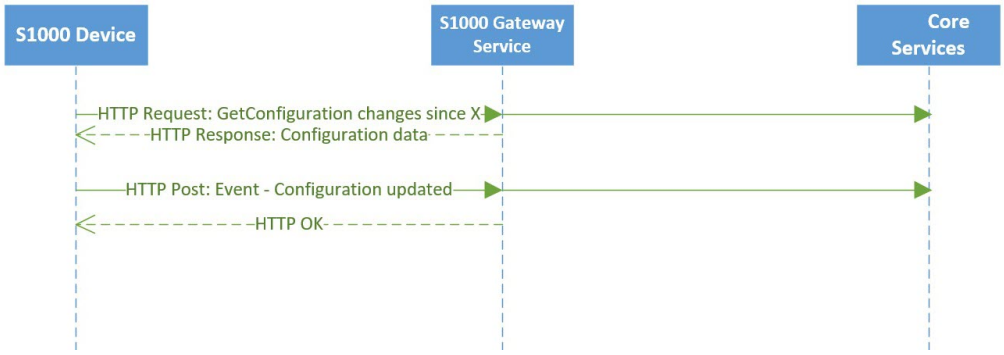


Once a session has been established, it is straightforward to issue a command and receive events. For example, the sequence diagram below shows VIGIL CORE updating the S1000 configuration.



An online S1000 receives a configuration notification immediately when a change has been applied to the controller by the operator.

An offline S1000, such as one on dial-up or low power mode may choose to check for configuration updates periodically. An operator can also force an S1000 on dial-up to connect to VIGIL CORE and immediately check for configuration updates. The sequence diagram below shows VIGIL CORE updating the S1000 configuration in offline mode.



Active Directory Integration

PACOM VIGIL CORE uses Active Directory for authentication and authorization, VIGIL CORE communicates through Active Directory Federation Services (AD FS) and OpenID Connect/OAuth.

Each user must exist in Active Directory and in VIGIL CORE.

> **Note:** Please contact your sales representative to access this feature.

Active Directory enrollment

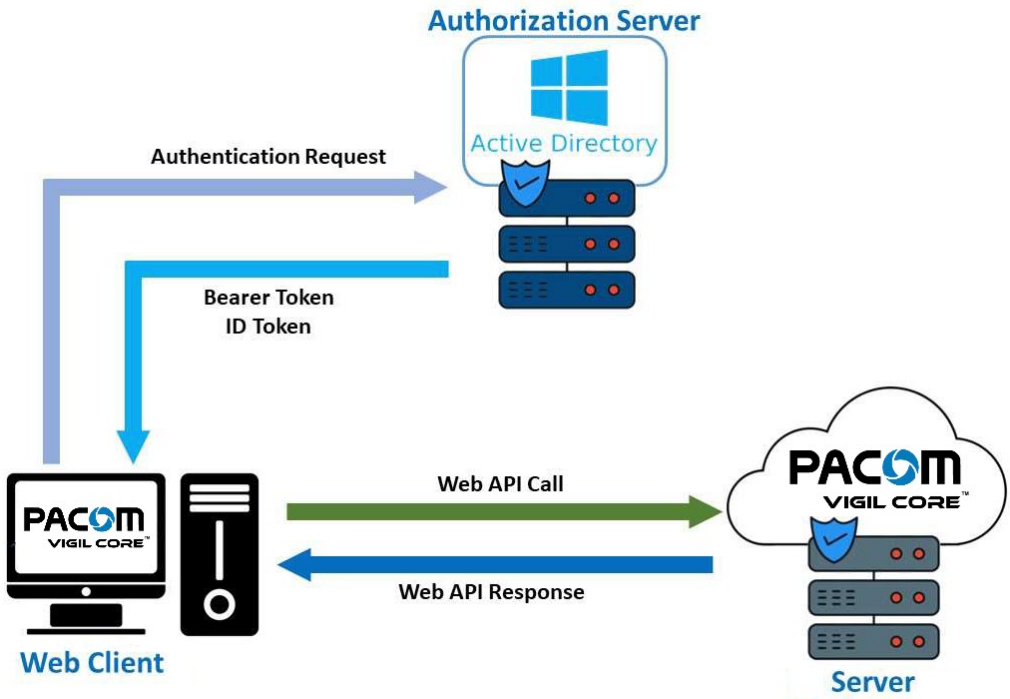
Once a user account is added to Active Directory, the user may be imported into VIGIL CORE from within the user screen. This maps the user's detail to VIGIL CORE and links the account to an Active Directory user.

The Active Directory groups are associated with VIGIL CORE roles using tags. As a user's groups are updated within Active Directory, their roles within VIGIL CORE are also updated.

The roles may be access or operator roles, controlling users' access credentials and their permissions within VIGIL CORE.

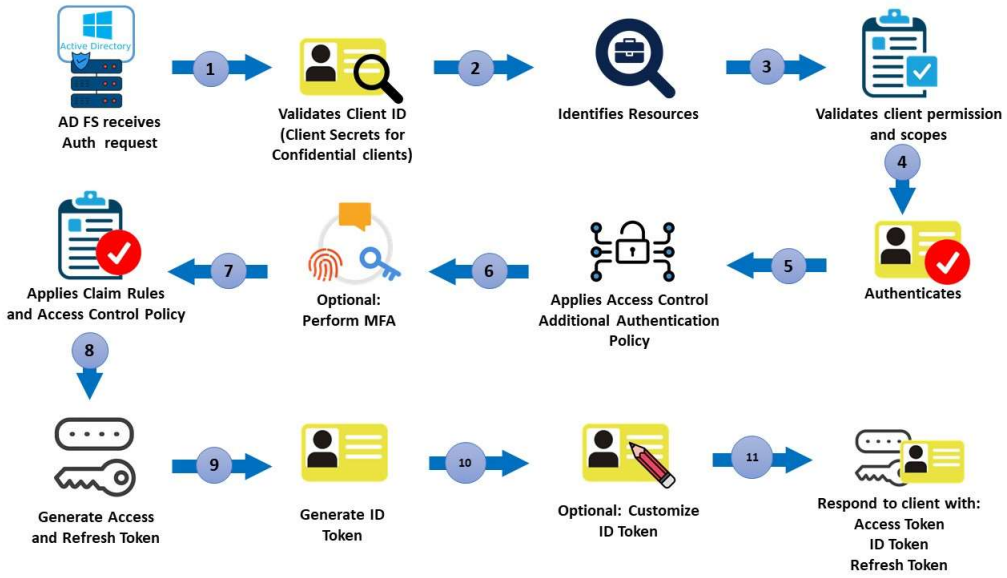
Active Directory authentication

VIGIL CORE authentication uses the standard AD FS authentication model for web applications using the OpenID Connect/OAuth concepts.



Active Directory authorization

Authorization is controlled through the allocation of tags to VIGIL CORE roles. These roles are then mapped after authorization. Changes in the Active Directory groups are monitored and the role allocated within VIGIL CORE is updated to reflect the allocations.



Active Directory configuration

Trust must be allocated between VIGIL CORE and Active Directory. This is achieved within VIGIL CORE through the set up of the Active Directory configuration of the VIGIL CORE Identity Server Service.

Technology Stack

PACOM VIGIL CORE's design is based on onion architecture principles so that it is easy to replace different parts of the infrastructure with alternatives.

> **Example:**
> If the database of choice is currently Microsoft SQL, with entity framework support it is relatively simple to support other databases including various cloud-based databases and various document databases (NoSQL, Azure, Redis, MonogDB).

Both the command side and query side databases are abstract enough so that they can also be replaced by othe caching technologies or databases.

Option	Description
API gateway	NGINX
ASP .Net core	Each service is a .Net core application, allowing deployment on Windows Servers, Linux Servers, in container and other virtualized environments or on a cloud-based platform.
Command datastore	This is the event sourcing storage mechanism that provides the audit log mechanism and is the source of truth for all operations that were successfully executed on the system. Basic version: Microsoft SQL or SQLite Advanced, large distributed system: Apache Cassandra or Apache Kafka
Environment	PACOM VIGIL CORE consists of a set of micro-services each hosted within a Docker container. The set of containers and the computer that they are managed on must all install Docker and be managed by a system using Docker swarm. This significantly reduces issues related to deploying in different IT environments.
Logging/Monitoring	Serilog with a centralized output to ElasticSearch or SQL.
Query datastore	These are the disposable query views that provide an optimized view of the data in the Command datastore. They can easily be rebuilt from the Command datastore as required by the system. Current: Microsoft SQL or SQLite Future: Redis
Web hosting	Each service runs in its own process running the light-weight and face Microsoft Kestrel web server. NGINX is used to host static web pages and act as a gateway for the Kestrel web services so that they are not exposed directly to the internet.