Technical Disclosure Commons

Defensive Publications Series

January 2020

SECURE AND EFFICIENT METHOD TO DISTRIBUTE CONFIGURATIONS IN WIRELESS CLUSTER DEPLOYMENTS USING HYPER LEDGER

Niranjan M. M

Nagaraj Kenchaiah

SECURE AND EFFICIENT METHOD TO DISTRIBUTE CONFIGURATIONS IN WIRELESS CLUSTER DEPLOYMENTS USING HYPER LEDGER

AUTHORS:

Niranjan M M
Nagaraj Kenchaiah

ABSTRACT

An enterprise wireless clustering deployment is comprises of cluster of Wireless Local Area Network (LAN) Controllers (WLCs), intended to provide collaborative services such as load balancing of Access Points (APs), distributed mDNS gateway, etc. Since these cluster deployments are typically very large, configuring individual WLCs is difficult. Presented herein are techniques to incorporate WLC cluster deployments with an authenticated distributed ledger to securely store the configuration and subsequent changes (e.g., only maintain changes from the previous one, using dictionary method: key-value pair to identify the difference). This avoids the use of control Datagram Transport Layer Security (DTLS) connection between AP and WLC for sharing the configuration, thereby giving access to the ledger based on the service registered by the worker WLCs or APs. For example, a worker WLC would register for services such as load balancing, mDNS gateway etc., to obtain the relevant configurations. Similarly, APs would register for wireless service to get configurations and policies based on the Site Tag (location).

DETAILED DESCRIPTION

Enterprise wireless clustering deployments are typically comprised of clusters/groups of Wireless Local Area Network (LAN) Controllers (WLCs), intended to provide collaborative services such as load Balancing of Access Points (APs), distributed mDNS gateway etc. A cluster is typically includes WLCs with designated roles as “Leader” or “Worker.” One WLC out of all WLCs in the cluster would be elected as the Leader using a consensus algorithm. The Leader WLC is the point of contact for all configurations, image management, load balancing (distribution) of APs among Worker WLCs, etc.

Since cluster deployments are typically very large (would include WLCs ranging from 3 to 10 or more), configuring individual WLCs is difficult. As such, in conventional arrangements, all configurations are handled only on the Leader WLC. Subsequently, whenever Worker WLCs connect to the Leader, the Leader WLC pushes the whole configuration to them. For sending the configurations, basic methods such as FTP/TFTP are used. For security reasons, these configurations would be sent using rsync or over IPSec, SSL/TLS tunnels. It is noted that the configurations are sent from Leader WLC to each Worker WLC independently. In addition, for any configuration changes on the Leader WLC, the Leader WLC needs to update all Worker WLCs with the same configurations independently.

There are also methods which fetch the configuration from the devices for any configuration change and then store the latest configuration. With these methods, the system loses the order of changes, the identity of who made the chance, and do not provide a distributed backup or distributed store.

In a cluster deployment, for a given Site/Location, policies are defined and mapped to the Site Tag on the Leader WLC. These Site Tags are mapped to the Worker WLCs and APs based on the Site location (either through static configuration or dynamically, i.e., AP Site Tag is deduced during discovery phase itself). Whenever a Worker WLC connects to the Leader WLC, the Leader WLC pushes all the configurations required based on the Site Tags to the Worker WLC independently. Similarly, whenever an AP sends a discovery request to the Cluster, the Leader WLC would do load balancing and elect the Worker WLC based on the Site Tag and sends back a discovery response along with this Site Tag.

Subsequently, when the AP sends a join request along with Site Tag (after CAPWAP Control DTLS tunnel establishing) to the Worker WLC, it would use this Site Tag to send the policy configurations for it. The AP uses the configuration to service the wireless network. Along with Control DTLS tunnel, the AP would establish CAPWAP Data DTLS tunnel with the Worker WLC for handling client specific data traffic.

From Cluster point of view, there are several drawbacks with these conventional methods. For example, the Leader WLC needs to push the configuration to every Worker WLC upon connectivity. In addition, for any change in the configuration, the Leader WLC need to push the same to every Worker WLC. If any changes are done (by mistake)

directly on the Worker WLC, it is difficult to trace. It is also difficult to track the changes that are made (without versioning system) and by whom (e.g., initially WLC1 could be Leader where configurations are done, but due to hardware failover, now WLC2 could be Leader, where administrator is performing changes).

There are also drawbacks with these conventional methods from the AP point of view. For example, each AP would establish a Control DTLS connection with the Worker WLC to share the configurations and image download. The whole Global and Site specific configuration has to be downloaded from the WLC for every AP independently. For any change in the configuration, the WLC needs to push the new changes to every AP in the deployment.

In addition, clustering deployments also poses multiple security vulnerabilities while communicating among WLCs and APs, including:

- **Control plane vulnerabilities (Controller compromising):** In a clustered environment, single compromised WLC can use to manipulate configurations of all the other WLCs in the network.
- **Link Level vulnerabilities:** Communication between WLCs and APs is susceptible to vulnerabilities like man-in-the-middle attack, eavesdrop the configurations and policy information, fabricate falsify policy information and that can circulate legitimately in distributed Wireless Cluster deployments.

As such, there is a need for a mechanism for devices to securely store configuration changes in a distributed way to share with other devices. In addition, the mechanism should be optimal to fetch configurations later (i.e., store the configuration required against the service offered or based on site/location etc.,) and also consider efficiency (in-terms of memory requirement (i.e., store only the different configurations compared with the previous versions). The techniques presented herein are configured to track the “who, what, when” associated with a configuration change in the private Blockchain. The techniques presented herein also addresses issues with the addition of a new worker WLC ("device"), replacement due to hardware failure, and allow the cluster to self-configure the new worker WLC without the need for external 3rd party systems. The techniques presented herein also

consider the optimization and efficiency to handle huge configuration of Wireless LAN Controller.

The techniques presented herein propose to incorporate WLC cluster deployments with an authenticated distributed ledger to securely store the configuration and subsequent changes (i.e., only maintain previous changes, using dictionary method: key-value pair to identify the difference). As such, the techniques presented herein avoid the use of a Control DTLS connection between APs and WLCs for sharing the configuration, thereby giving access to the ledger based on the service registered by the Worker WLCs or APs. A Worker WLC would register for services such as load balancing, mDNS gateway etc., and APs would register for wireless service to get policies based on the Site Tag.

A data DTLS connection could continue to be provided and used for handling client specific data traffic. Currently, the Control DTLS connection is used for client association, PMK sync etc., also, we need to move these to use Data DTLS connection. The WLCs and APs in the cluster deployment form a group to construct a private Blockchain and all configuration actions are stored in the created private Blockchain as Hyper Ledger. This would allow an accurate tracking of the "who, what and when" associated with configuration changes. This would also remove the need to store the whole configuration on the Leader WLC (centralized way), which in-turn needs to have high availability to handle failure cases (Apart from storing on the Leader WLC, need to take backup outside the deployment as well).

Many devices have a Plug and Play (PnP) agent integrated into it and is applicable even to WLCs and APs in the cluster deployment, which play a role in the initial configuration of the device. PnP agent can interact with the device configuration management system running on the Leader WLC and run a lightweight Blockchain client to update device configuration changes and record the information about the changes in the Authenticated Hyper Ledger, which is distributed across the cluster deployment of the WLCs/APs for that enterprise/site.

The advantage of using a PnP agent is that it is already there is most of the devices, thus less adoption issues would be incurred, and the PnP agent works with the configuration management system to extract or push any configuration changes to the device. PnP agents are well integrated with device security frameworks for initial on-boarding, which can be

leverage for securing the Blockchain client certificate information in the device. Certain devices use the ACT2 chip, which is a tamper-proof hardware chip used for storing the certificates. A Blockchain client running on the WLCs and APs authenticates itself using SUDI certificate stored in the ACT2 chip to prevent malicious devices from writing to the Blockchain or even reading from the Blockchain, which is a requirement to use Authenticated Hyper Ledger.

With the thin cloud being developed, configurations are directly pushed to individual APs using GRPC from the Cloud software. The proposed mechanism can be used even for thin cloud deployments by creating private Blockchain among cloud software and APs in the enterprise (on-premises) deployments.

From the Cluster point of view:

Both Leader and Worker WLCs register with the private Blockchain with specific services (such as Load Balancing, mDNS gateway etc.,) which they are interested, so that they can access the authenticated Hyper Ledger provided for that service. All configurations are done only on the Leader WLC. To simplify the configuration, config namespace is introduced to configure Site Tags and their policies, WLANs, VLANs etc., for a particular cluster. These config namespaces are mapped to the Worker WLCs on the Leader WLC, for example:

- <WLC>config namespace BGL18
- <WLC-BGL18> config t
- <WLC-BGL18-Config>ap sitetag Site1

Worker WLCs connect to the Leader WLC and learn about mapped config namespace containing information about one or more Site Tags. The Worker WLC would fetch global and Site Tag configurations/policies of particular config namespace from the Authenticated Hyper Ledger.

From the AP point of view:

Whenever AP is booted up, it also registers with the private Blockchain as a Wireless service, which in-turn has policies/configurations related to the Site Tag. The AP would fetch the Site Specific configurations/policies from the Authenticated Ledger. It is

noted that, since Blockchain has implicit secure connectivity, Control DTLS tunnel between each AP and WLC is not required for config transfer. Currently, image upgrade/download on the AP is also over Control DTLS tunnel. For this, Blockchain can be used to provide the image version(s) available on the WLCs and the AP can download directly from the WLC using secure transfer mechanisms. Optionally, SFTP/HTTPS secure transfer methods could be used.

Figure 1, below, is a diagram illustrating a method to distribute configurations on a cluster deployment, in accordance with embodiments presented herein.

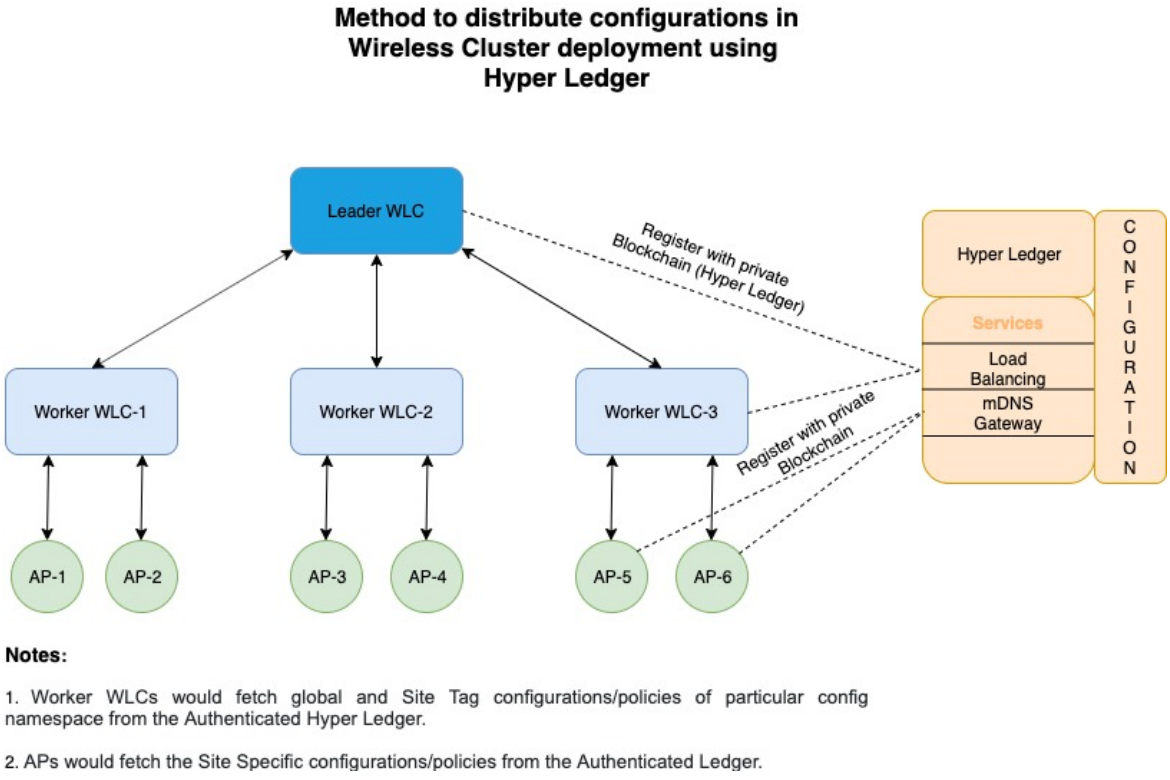

Figure 1

To add a new AP/WLC to the Cluster, after authentication with the Blockchain, access to the Hyper Ledger is granted to obtain the initial config as well sub-sequent "config diff," as per the registered service and apply all configuration in order. For RMA of AP/WLC, a new AP/WLC authenticates using a SUDI certificate and registers required services with the Blockchain to obtain all relevant configurations based on the registered.

The techniques presented herein have several advantages, including:

- Distributed secure storage of the configurations (De-centralized).
- Storing only the modified config changes in the Blockchain (storing only config diff).
- Avoids having Control DTLS connection between WLC and APs for sharing configuration (i.e., no need to have individual control DTLS tunnel between WLC and AP).
- Easy to track "who, what, and when" associated with individual changes.
- Authenticated way of accessing private Blockchain using device specific SUDI certificate stored in the ACT2 chip, thereby preventing malicious devices from writing to the Blockchain or even reading from the Blockchain (Security).
- Isolated networks from the Internet are unable to take advantage of certain vendor PnP and may require 3rd party software for configuration backup to be deployed, which means a customer that has multiple isolated networks would need multiple of those configuration backup systems. The customer has to be concerned with storage and managing of these 3rd party systems. If the Blockchain can be distributed among devices, then storage is built in and automatically supports isolated networks. Also having individual configuration changes in time will allow a customer or an engineer to better debug when an issue has occurred. (Isolated network).

In use, the techniques presented herein can be used to share configuration among WLCs in the Cluster. In addition, the techniques presented herein can be used to share Site tag specific configurations/policies with the APs. Moreover, the techniques presented herein can be used for thin cloud based deployments.