PNNL-26560

Pacific Northwest
NATIONAL LABORATORY

Proudly Operated by **Battelle** *Since 1965*

Security Features of VOLTTRON™ Distributed Sensing and Control Platform (2017)

June 2017

BA Akyol
CH Allwardt
MC Roup
JN Haack
KE Monson

U.S. DEPARTMENT OF
ENERGY

Prepared for the U.S. Department of Energy
under Contract **DE-AC05-76RL01830**

PNNL-SA-26560

Security Features of VOLTTRON™ Distributed Sensing and Control Platform (2017)

BA Akyol JN Haack
CH Allwardt KE Monson
MC Roup

June 2017

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

This document updates PNNL-SA-106580.

Pacific Northwest National Laboratory
Richland, Washington 99352

Contents

1.0 Introduction

VOLTTRON™[1] enables rapid authoring and secure deployment of autonomous software agents for distributed sensing and controls. It is designed to be as secure as possible to meet desired security objectives; however, no software can be 100 percent secure and useful at the same time. Therefore, VOLTTRON uses a threat model approach for determining threats and vulnerabilities of the software and to reasonably reduce the attack surface and/or harm endured after a compromise.

VOLTTRON security development focuses on:

- Protecting the integrity of agent programming through cryptographic means [restricted[2]]
- Protecting agents from using excessive system resources to prevent platform instability [restricted]
- Protecting agent configuration (and work orders) from manipulation [restricted]
- Securing communications between VOLTTRON platforms and external data sources
- Securing communications between platform instances
- Securing communications between agents running on the same VOLTTRON platform
- Platform-hardening recommendations for all VOLTTRON users and use cases
- CurveMQ encryption and authentication enabled for TCP connections into the platform
- All connections are authenticated
- Authorization based on identity, domain, endpoint, and authentication credentials
- All messages include user id for authorization and attribution associated on the platform and cannot be spoofed by agents
- Platform refuses to run as root to prevent damage and escalation
- Automated secure deployment and key discovery
- Improved usability of security features

This document contains a short summary of the security features implemented in the VOLTTRON platform as of release 4.1. The next section provides a summary of the VOLTTRON security philosophy, followed by a discussion of the threat modeling associated with the platform and a list of security features (including features planned for future VOLTTRON releases). The VOLTTRON development team also recommends a thorough review of NIST SP800-82 Guide to Industrial Control Systems Security[3].

[1] https://volttron.org/sites/default/files/publications/PNNL-25499_VOLTTRON_2016.pdf

[2] In this context, "restricted" indicates that to be able to use this functionality, the user needs to execute a license agreement with PNNL. The license agreement is royalty free for use with buildings and grid.

[3] NIST SP800-82 Guide to Industrial Control Systems Security accessed at http://csrc.nist.gov/publications/nistpubs/800-82/SP800-82-final.pdf on November 17, 2014.

2.0 VOLTTRON Security Philosophy

VOLTTRON was designed with security in mind before a single line of code was authored. VOLTTRON team spent the better part of a year working on requirements for the platform, and security was a primary area of concern. The results of that work are captured in Akyol (2011)[1] and Akyol (2012)[2]. Initial goals of the VOLTTRON platform were to bring intelligence and distributed decision making to the edges of the power grid and to enable integration of smart loads and distributed generation with the bulk power system. To perform these tasks, VOLTTRON needed comprehensive security features. VOLTTRON security can be summarized in the following five categories.

Category 1: Securing the external interfaces of the platform. Availability, integrity, and confidentiality are key requirements for any control system (and in the order as written). VOLTTRON interfaces with many external entities as required to perform a function. These entities include (but are not limited to) Cloud services for storing data and analytics, external services for weather forecasting or climate data, power grid services for retrieving metering and rate information, and other VOLTTRON instances for collaborative control applications such as "intelligent load control[3]." VOLTTRON **only** uses communication methods that provide integrity and confidentiality when interfacing with external services. For interfacing with web services, agents running on VOLTTRON are recommended to use Internet Engineering Task Force's transport layer security (TLS) protocol with X.509 public key infrastructure certificates. By default, VOLTTRON does not have any trust roots (or trust anchors) configured (i.e., VOLTTRON trusts no one). The administrator must configure the trust roots as part of the installation. TLS supports many cryptographic ciphers. VOLTTRON uses ciphers that meet NIST recommendations for key length and cipher algorithm. All communications between VOLTTRON instances are also secured for integrity and confidentiality using elliptic-curve cryptography. Using cryptographically secure methods of communication allows for increased platform availability if the underlying hardware is robust enough to handle additional network traffic load. Another method to increase the availability of external interfaces is to use a service such as virtual private networking to further limit the attack surface. For web services, an additional piece of hardware (commonly referred to as a web application firewall, or WAF) can be used to offload processing, perform deep-packet inspection and filter out malicious traffic. VOLTTRON is fully compatible with WAF technology and the VOLTTRON team uses WAF in deployments. VOLTTRON also interfaces with devices and building automation systems using BACNet, MODBUS or other protocols. Since these protocols are commonly used without encryption or authentication, the VOLTTRON team recommends applying appropriate cyber security controls as discussed in Appendix A.

Category 2: Securing the inner workings of the platform. VOLTTRON platform features drivers that interface with external entities as discussed previously. The platform also includes the message bus, and agents that run on the platform to perform tasks such as analytics, fault detection, diagnostics, and implementation of control system algorithms. To secure the inner workings of the platform, VOLTTRON

[1] Akyol BA, JN Haack, CW Tews, BJ Carpenter, AV Kulkarni, and PA Craig, Jr. 2011. "An Intelligent Sensor Framework for the Power Grid." In Proceedings of the 5th International Conference on Energy Sustainability and 9th Fuel Cell Science, Engineering and Technology Conference, August 7-10, 2011, Washington DC, Paper No. ESFuelCell2011-54619. ASME, New York, NY.

[2] Akyol BA, JN Haack, S Ciraci, BJ Carpenter, M Vlachopoulou, and CW Tews. 2012. "VOLTTRON: An Agent Execution Platform for the Electric Power System." In Third International Workshop on Agent Technologies for Energy Systems (ATES 2012). A workshop of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012), June 5, 2012, Valencia, Spain. International Foundation for Autonomous Agents and Multiagent Systems

[3] Kim W, Katipamula S. 2017. "Development and Validation of an Intelligent Load Control Algorithm," *Energy and Buildings* 135:62-73. 10.1016/j.enbuild.2016.11.040.

employs several security controls. The message bus validates the identity of every agent (authentication) and controls what agents get to subscribe to what topics (authorization). All message bus transactions may be logged using the Listener Agent. All logs are transferred off the platform to a separate log collector for further analysis. The controls implemented within the message bus allow the platform to secure intra-platform communications and validate the agents running on top of the platform. Agents run as separate processes on top of Linux to minimize interference with the platform. The platform itself refuses to run as a privileged user. Additional controls that use Linux cgroups are implemented to increase the availability of the platform by guaranteeing that the agents running on a platform do not exceed the resources available to the platform. Specifically, using Linux cgroups, we can limit the resources used by each agent including processor, memory, or storage. We can also limit agent access to external services, and limit agent communication. Another security control used by the platform to increase availability, integrity, and confidentiality is the use of cryptographic methods to protect the agent code and configuration. Agent configuration and code are only executed after the contents of the agent software package is validated via cryptographic means. Validating agent code and configuration before execution prevents unauthorized modification of agents and strengthens the security posture of the platform.

Category 3: Securing the interface between the platform and underlying operating system. Application developers trust the underlying operating system. VOLTTRON requires that the underlying operating system be secured according to platform-hardening recommendations.

Category 4: Securing the underlying operating system and reducing the attack surface. VOLTTRON is built on top of Linux. Linux is a modern open-source operating system used in many mission-critical applications. Linux is also used as a desktop operating system with features comparable to Windows or MacOS X. To secure a VOLTTRON deployment, the underlying operating system must be secured as well. Securing the underlying Linux installation is discussed in detail in Appendix B. Securing the underlying Linux installation reduces the total attack surface of the system. For example, if an organization uses a small number of systems as “jump boxes” to access VOLTTRONs deployed in the field, then putting in a firewall rule in those systems to limit all management access to only those jump boxes is the right thing to do. Instead of allowing all of Internet, this only allows a handful of systems access. This is especially important if the organization relies on passwords to control system access. Simple as it sounds, it is also recommended to not use common user names such as “root” or “admin.” It is also necessary to turn off services such as “sendmail” or “lpd” that are not needed in a control system environment. Appendix B discusses more techniques that could be used to secure VOLTTRON installations and reduce the attack surface.

Category 5: Security requirements for underlying hardware. The VOLTTRON platform is designed to be deployed in many control and sensing applications. Platform capabilities scale gracefully with the capabilities of the hardware that runs VOLTTRON. For any security conscious deployment of VOLTTRON, the following security features (not an exhaustive list) for the underlying hardware are recommended:

- Supply chain security to weed out counterfeit or malicious hardware. For VOLTTRON deployments in critical control systems and power grid, the hardware used should be manufactured following supply chain security recommendations made by NIST[1].
- Physical tampering detection. Uncontrolled physical access to any control system may result in cyber security compromise. The ability to detect physical tampering of hardware in addition to using physical security practices such as locked cabinets, fences etc. can indicate when physical security of the control system is compromised. With this knowledge, we can take appropriate action including

[1] NIST Supply Chain Risk Management recommendations can be found at http://csrc.nist.gov/scrm/ and http://dx.doi.org/10.6028/NIST.SP.800-161.

dispatching personnel to remove the suspect device from the control system or remotely take a device off the network.

- Hardware-based cryptographic module. Hardware-based cryptographic modules perform three important functions: protection of private keys and other sensitive information, a reliable random number generator, and acceleration of cryptographic operations. The first two items on this list are covered in detail by FIPS140-2[1] and several vendors offer processors and micro-controllers with FIPS140-2 certified cryptographic modules. The third item—accelerating cryptographic operations—is directly related to the availability goals of a control system. Offloading potentially expensive cryptographic operations to a co-processor allows the system to tolerate high amounts of protected traffic and remain available for control system operations. Modern central processing unit (CPU) families including x86, IA64, and ARM offer support for accelerated cryptographic operations for NIST validated algorithms[2].
- BIOS/Firmware/Bootloader security. Controlling low-level firmware of the system allows us to implement several important security features. For example, we can control from which image the system boots. The firmware can verify the image using cryptographic means before booting. The bootloader can lock the critical memory regions of the system as read only so that they cannot be modified. We can have the bootloader to verify cryptographic functions as part of the boot process to make sure they return known good results. As an availability requirement, the bootloader can be made redundant so that there is always a known good bootloader to initialize the system and boot a valid image. All of these techniques are well known in embedded system community and commonly used in building secure products.
- Hardware-based storage encryption. We can secure data during transport using encrypted and authenticated communications. Security of the data at rest depends on the physical security of the system as well as the security of the storage. For systems deployed in insecure locations where an adversary may gain access to the data at rest, storage encryption is a necessity. Similarly, systems handling sensitive information such as consumer transactions, occupant behavioral patterns, or market information, storage encryption is also required. The best way to guarantee system availability while employing storage encryption is to use hardware-based storage encryption. The hardware assist allows the system to perform encryption without impacting the main processor.
- Hardware-based network packet filtering. Any networked control system must be able to handle incoming traffic at the full line rate and still continue to function normally. This processing capability is a critical requirement for availability. Most control systems have main processors with limited power; therefore, a hardware-based network packet handling and filtering may be required to meet availability goals. A system that crashes or becomes permanently unavailable when scanned by an application (such as nmap[3]) is simply not acceptable.
- Memory address-specific read or write access protection. Limiting write access to certain locations prevents key application or operating system code from being overwritten either accidentally or maliciously. Being able to limit read access for a particular location allows us to protect sensitive information. Modern CPU families implement these capabilities as part of their security features.

This section presented a summary of VOLTTRON security philosophy by highlighting key security features. The next section presents a detailed threat analysis of VOLTTRON and the reasoning behind the security features described in this section.

[1] FIPS140-2 related documents can be retrieved from http://csrc.nist.gov/groups/STM/cmvp/standards.html.
[2] NIST published cryptographic algorithms that they validate at http://csrc.nist.gov/groups/STM/cavp/.
[3] Nmap may be obtained from https://nmap.org.

3.0 Threat Model

Figure 1 below shows two VOLTTRON systems communicating with each other and with external sources. The system on the left side is expanded to show internal components of VOLTTRON and communication with an external historian and a Cloud-based service. Each interface in the figure is numbered to be referenced in the text to discuss security threats associated with each interface. As noted previously, VOLTTRON is built on top of Linux. While VOLTTRON addresses threats associated with the platform and its interfaces, equal consideration needs to be given to the security of the underlying operating system. The VOLTTRON team relies on comprehensive security hardening of the operating system and keeping up-to-date with periodically applied patches to further enhance the security of the instances deployed in the field.

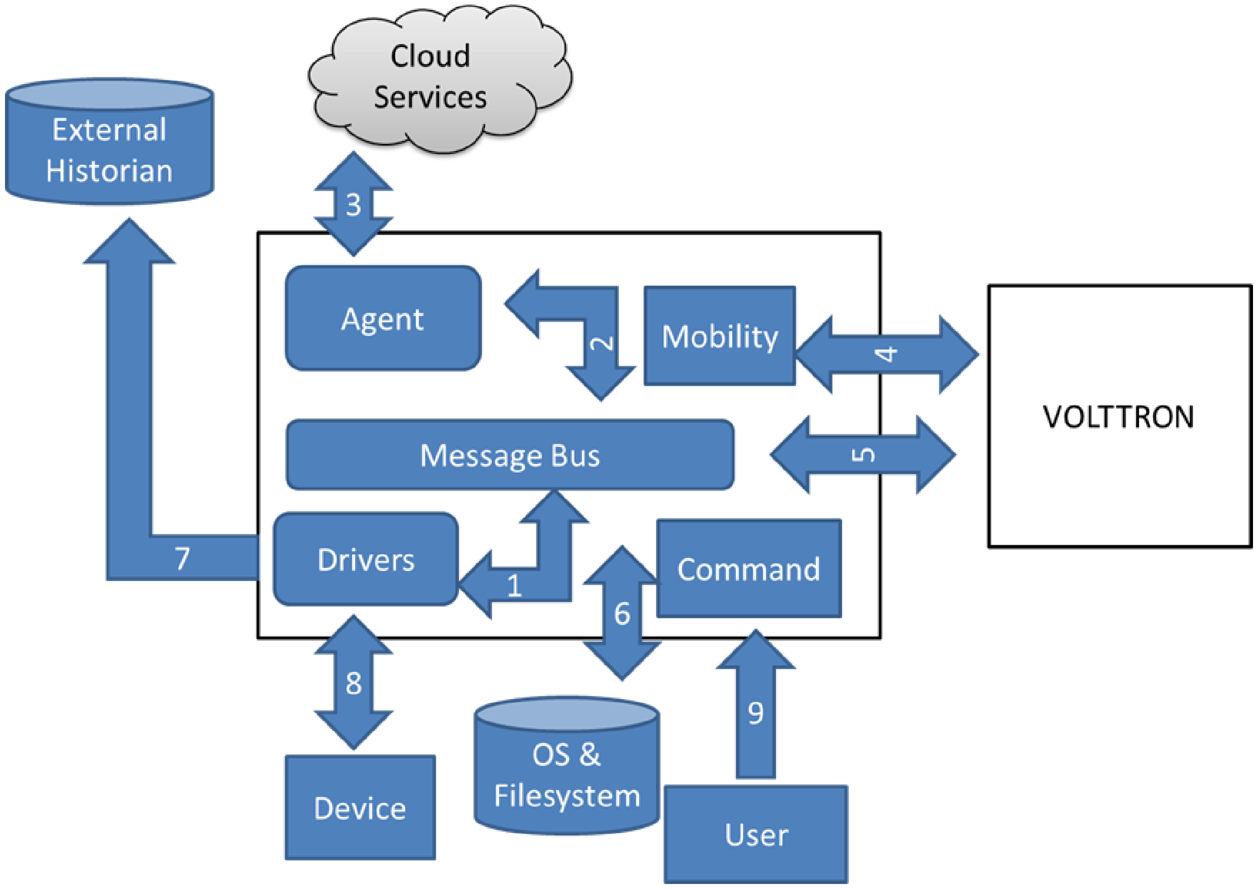

Figure 1. Schematic of VOLTTRON Internal and External Components

This section follows the terminology and structure shown below. Possible attack vectors against the VOLTTRON software are discussed followed by the associated risks and mitigations for reducing, alleviating, or even eliminating the risks. Some threats also include future strategies for improving the mitigations and even further reducing the risk. Each vulnerability is described in the following template:

1. Description of vulnerability/threat.

   R: Associated risk indicating what might occur if the vulnerability is exercised.

M: Mitigation that should/will be implemented to reduce or eliminate the associated risk.

F: Future strategies for mitigation (optional).

3.1 Communications between VOLTTRON and other services (e.g., Cloud, etc.) (Interfaces 3, 7)

The VOLTTRON platform allows agents to act as proxies to external resources to move information between a service and the platform (3). In addition, the platform can utilize an external service for storing data collected from devices managed by the platform and data logged by applications (7). Vulnerabilities associated with these interfaces are listed below:

1. Communication between agents and the Cloud services/external historian can be intercepted, tampered with, or snooped upon by a third party.

   R: A remote entity can insert itself between VOLTTRON and external entities. Once in the path, the remote entity can tamper with communications, affecting integrity, or snoop the communications, affecting confidentiality.

   M: VOLTTRON uses standardized communication protocols (TLS) when communicating with external entities. These protocols provide both message integrity and confidentiality services. Identities are authenticated by means of X.509 certificates.

2. Communication between agents and the Cloud services/external historian can be stopped by either communications network availability or third-party denial of service (DoS).

   R: A third party can, by means of DoS attacks or by other means, temporarily interrupt communication between VOLTTRON and external entities. This could result in loss of data or sub-optimal behavior of the control system.

   M: VOLTTRON buffers all data that cannot be transmitted to external services. The buffer amount is only limited by the storage available to the platform.

   M: VOLTTRON can enter safe control mode if it cannot communicate with an external controlling entity (for example, an entity that sets control policies). The handling of loss-of-control policy communication between agents and devices should be handled by the agents. The platform has the ability to initiate a control override to lock out control of devices in cases where policy is unclear due to loss of communication with the central instance.

3. An external entity communicating with VOLTTRON can be compromised and data coming from or going to VOLTTRON can be intercepted at the non-VOLTTRON end of the communication channel.

   R: If an entity communicating with VOLTTRON is compromised, data coming from or going to VOLTTRON can be intercepted or modified.

   M: All entities that communicate with any control system should be protected as well as the control system itself. If the security of a third-party entity is suspect, VOLTTRON developers recommend not trusting the entity at all.

4. Communication between agents and the Cloud services (or other outside, network-based services) could be used as an attack vector to compromise the system by intercepting unencrypted or

improperly encrypted communications, compromising the remote system, or by performing other attacks.

R: A remote entity could send carefully crafted messages to agents, causing the execution of unauthorized code.

M: Agent code will be reviewed for security issues and malicious intent. VOLTTRON foundation will support review functions. All agent developers should use encrypted and authenticated services and should validate all inputs from other sources before use. VOLTTRON uses elliptic curve cryptography when communicating with other platforms including VOLTTRON Central. All communications with external services use TLS.

F: Full agent containerization limits the effects and reach of a compromised agent.

5. Agents may communicate with any system in the Cloud with which any other agent is also able to communicate (assuming the firewall allows such communication).

R: If an agent is inadequately authenticated or knows the credentials of the remote account, it could send malicious communications.

M: Agent code will be reviewed for security issues and malicious intent.

F: Agents will be sandboxed to disallow unauthorized communications.

R: An agent may exfiltrate data to systems in the Cloud.

M: Agent code will be reviewed for security issues and malicious intent.

M: Firewall rules may be used to limit communications with remote systems.

F: Agents will be sandboxed to disallow unauthorized communications.

3.2 Communications between multiple VOLTTRON instances (Interfaces 4, 5)

VOLTTRON platforms can communicate with each other using the VOLTTRON Interconnect Protocol (VIP) (5). VIP is a layer on top of ZeroMQ that allows secure remote procedure calls to be made to the message bus and allows agents to interact with remote message buses. VOLTTRON also provides a Mobility Service (4), which enables agents to move between platforms and enables administrators to provision VOLTTRON devices in the field. Vulnerabilities associated with these interfaces are listed below:

1. Agents may communicate between the messaging buses of platforms located on different systems utilizing VIP.

R: Agents may subscribe to any topic, without limit on the remote platform.

M: Agent code will be reviewed for the proper use of remote subscriptions.

M: Security features built into VIP provide authentication and authorization for access control to the messaging bus.

R: Agents may publish to any topic, without limit, when remote publishing is enabled on the remote platform.

M: Agent code will be reviewed for the proper use of remote publishing.

M: Security features built into VIP provide authentication and authorization for access control to the messaging bus.

R: Multi-node messages may cross uncontrolled networks providing an opportunity for interception or modification.

M: Multi-node messaging uses the elliptic-curve encryption technology of ØMQ's CurveZMQ protocol to authenticate and encrypt traffic between nodes. Communicating parties are mutually authenticated to prevent unauthorized communications.

R: A port must be opened when the Mobility Service is enabled.

M: Using VIP-based communication and agent transfer service removes the need for these additional ports.

M: Firewall rules may be applied to help limit the effectiveness of such attacks. VOLTTRON provides no protection itself against DoS attacks.

F: Opening a single port and multiplexing all traffic will limit the number of network ports requiring exposure to the Internet.

2. Agents may move between platforms over the network introducing them to possible man-in-the-middle attacks, spoofing, or other network-directed attacks.

    R: An unauthorized user might intercept an agent and modify it for malicious use or create their own agent and send it on behalf of a platform they are not authorized to use.

    M: SSH tunnels are used to authenticate and encrypt communications between platforms when moving agents. Agent code is cryptographically signed using X.509 certificates. SSH keys are managed using standard SSH configuration files and signing keys are managed using X.509 certificates [restricted].

    M: Communication between platforms is encrypted and access to the message bus can be restricted by IP address and credentials.

    F: SSH key usage will be converted to use the X.509 certificate infrastructure.

3. JSON-RPC for the Mobility Service (JavaScript object notation-remote procedure call) function calls are issued over an SSH tunnel to coordinate agents moving between platforms.

    R: JSON messages must be completely read into memory. Receiving many extremely large messages could result in a low memory condition. JSON was chosen for serialization because of its simplicity and its unlikeliness to be the source of inadvertent vulnerabilities.

F: A maximum message size could limit memory used. Limiting the number of concurrently parsed messages could help reduce memory usage, but at the cost of slowing communications.

3.3 Agent multi-tenancy inside a VOLTTRON platform (Interfaces 1, 2, and others)

VOLTTRON supports multiple agents and services to run simultaneously. This ability is referred to as multi-tenancy. The following are potential vulnerabilities related to agent and service multi-tenancy inside the platform:

1. All agents run under the same user account, with the same privilege level.

   R: A malicious agent can interfere with other agents, possibly sending them signals, killing them, or overwriting data.

   M: Agent code will be reviewed for security issues and malicious intent.

   M: Agent code and configuration is signed (multiple times) and verified before each execution. This prevents an unauthorized third party from tampering with agent code. [restricted]

   F: Full containerization of agent code will effectively isolate agents.

2. Agent code runs under the same user account and at the same privilege level as the platform supervisory daemon.

   R: A malicious agent can interfere with the platform supervisor, killing it and/or overwriting data. It could also assume the supervisor's role as manager of the communications bus, allowing it to intercept, modify, and/or drop agent communications. This also includes the ability to remove CPU and memory limits [restricted].

   M: Agent code will be reviewed for security issues and malicious intent.

   M: Agent code and configuration is signed (multiple times) and verified before each execution. This prevents an unauthorized third party from tampering with agent code. [restricted]

   F: Full containerization of agent code will effectively isolate and hide agents from the supervisor.

3. Local communication between agents over the message bus is authenticated, preventing a malicious agent from mimicking other agents or the supervisor and sending messages on their behalf, potentially causing the creation of unauthentic data or the unauthorized actuation of a device.

   M: Agent code will be reviewed for security issues and malicious intent.

   M: Agent code and configuration is signed (multiple times) and verified before each execution. This prevents an unauthorized third party from tampering with agent code. Platform will not allow execution of unauthorized agents. [restricted]

   M: Security features built into VIP provide authentication and authorization for access control to the messaging bus.

M: Operating system and VOLTTRON-level features defend against compromise of the internal messaging bus.

4. Local communication between agents over the message bus is encrypted preventing a malicious agent from subscribing to every message sent on the message bus and retransmitting it or using it for other unintended purposes.

    M: Security features built into VIP provide encrypted inter-agent communications on the messaging bus.

    M: Topics can be restricted to an allowed subset of agents.

    M: Agent code will be reviewed for security issues and malicious intent.

    M: Agent code and configuration is signed (multiple times) and verified before each execution. This prevents an unauthorized third party from tampering with agent code. Platform will not allow execution of unauthorized agents. [restricted]

    M: Operating system and VOLTTRON-level features defend against compromise of the internal messaging bus.

3.4 Communicating with and controlling devices (Interface 8)

VOLTTRON drivers (8) allow the platform to both collect data from devices and send control commands. These drivers utilize protocols such as MODBUS, BACnet, or custom-built software to communicate with the device and use the platform's message bus to communicate with agents on the platform. Overall security of the underlying communications protocol such as BACnet is outside the scope of this document. VOLTTRON developers recommend using appropriate cyber security measures to protect the underlying protocol. See the Appendices and the article by Neilson (2013)[1] for a good example on how to secure a control systems network.

1. Unsecured communications between VOLTTRON and legacy control devices (e.g., Modbus, BACnet) may be intercepted and modified by a third party.

    R: A third party can modify communications between VOLTTRON and controlled devices using legacy protocols. It is even possible to impersonate a controlled device. An agent can then react incorrectly to information coming from such a device.

    M: Security measures as described in NIST SP800-82 and Neilson (2013) are to be used to protect legacy control system devices that do not have sufficient security protections.

    M: VOLTTRON agents can be written to validate information being received from legacy devices to check for range, historically known trends, etc.

    M: VOLTTRON device drivers are written to prevent potential exploits from "overflow"-type attacks to gain access to the platform by means of data being passed by a legacy device. The VOLTTRON platform cannot be compromised by means of incorrect data streams.

2. Agents may communicate directly with local devices bypassing the platform drivers.

[1] Neilson, C. 2013. *Securing a Control Systems Network*. ASHRAE Journal. November 2013.

R: An agent could send commands outside the supervision of the scheduler/actuator causing equipment to operate in an unsafe way.

M: Agent code will be reviewed for security issues and malicious intent before deployment.

M: Agent code and configuration is signed (multiple times) and verified before each execution. This prevents an unauthorized third party from tampering with agent code. Platform will not allow execution of unauthorized agents. [restricted]

M: If underlying protocol supports it, VOLTTRON can implement secure and authenticated communications. For example, for thermostats that support secure socket layer (SSL) communication, VOLTTRON can communicate with the device using SSL.

F: Agents can be sandboxed to disallow unauthorized communications.

3. A device could be physically tampered with by malicious actors.

    R: The device could be modified to send inaccurate readings in an attempt to make agents take incorrect and possibly damaging actions.

    M: Building and property owners must maintain awareness of their devices to ensure protection and follow best practice guidance described by Neilson (2013) and the guidance provided in the Appendices.

    M: Building owners must implement appropriate physical security measures.

    M: VOLTTRON includes agents that can be configured to send alerts if monitored points exceed certain allowed thresholds. This mechanism could be combined with machine learning for more sophisticated solutions.

    F: In an extension of extending on existing fault detection work, agents could be developed to monitor devices for improper and unexpected behavior. PNNL has been awarded a contract by DOE Office of Energy Cybersecurity for Energy Delivery Ssystems program to incorporate anomaly detection into VOLTTRON.

3.5 Underlying operating system and file system (Interface 6)

As VOLTTRON is built on top of a modern Linux operating system, he VOLTTRON team relies on comprehensive security hardening of the operating system as well as keeping up-to-date with periodically applied patches to further enhance the security of the instances deployed in the field. The following vulnerabilities (not an exhaustive list) are related to the interface between VOLTTRON and the underlying operating system:

1. Underlying operating system platform can be vulnerable to attacks due to incorrect or incomplete security update patching.

    R: If a platform is not maintained with respect to security patches, an attacker will be able to compromise the system and gain access.

M: All VOLTTRON systems deployed in the field are configured to automatically download and apply security patches. VOLTTRON developers recognize that in some environments, unattended upgrades are not practical or even can be dangerous. In this case, an engineer should apply security updates on a weekly or monthly basis.

M: VOLTTRON systems are protected by appropriate network security measures such as host-based firewalls, intrusion detection, and security-monitoring tools to prevent unauthorized access.

2. Underlying OS platform is insufficiently hardened.

   R: Security settings of the underlying operating system are not managed correctly and allow overly broad ("loose") access. An example could be the "guest" account or running unnecessary services.

   M: All VOLTTRON platforms deployed in the field by VOLTTRON developers endure a security hardening process that includes turning off unnecessary services, restricting access to necessary services, activating host-based firewall controls and allowing access to the system only by authorized users and hosts.

   M: The VOLTTRON team documents platform-hardening settings as part of the VOLTTRON user's guide and the recommendations are included in Appendix B.

3. Physical access to the VOLTTRON platform is not controlled.

   R: An attacker that has physical access can circumvent many security measures implemented in software.

   M: Physical access to any control system must be controlled. There are no exceptions.

   M: Linux supports hard disk or volume encryption but this is not sufficient to defend against an attacker that has physical access.

4. Platform supervisor and agents run on a shared (multi-user) system (6).

   R: Other users on the system might be able interact with supervisor or agent processes, files, and/or sockets. A malicious agent might be able to access processes, files, and sockets belonging to other users of the system.

   M: In a production environment, the supervisor should run under its own unprivileged account. Files are writable only by the supervisor process's owner. Sensitive files are only readable by the supervisor process's owner. Appropriate file system permissions are set on Unix domain sockets to prevent their use by unauthorized users and/or peers are authenticated. TCP and user datagram protocol sockets require authentication.

   F: Full containerization of agent code will effectively isolate agents and hide much of the system from them.

R: An agent may consume too much memory, intentionally or through a programming error, causing the system to become unresponsive and forcing other applications and/or agents to terminate.

M: A resource monitor is used to place the agent in a memory Linux control group (cgroup) to limit memory usage to what was negotiated before execution. [restricted]

M: An agent's out-of-memory (OOM) killer priority can be set lower than critical applications and higher-priority agents so that an OOM condition will cause lower priority agents to be killed first. [restricted]

R: An agent may consume too many CPU cycles, intentionally or through a programming error, causing the system to become unresponsive and forcing other applications and/or agents to terminate.

M: A resource monitor places the agent in a CPU cgroup to limit its CPU utilization to what was negotiated before execution. [restricted]

Appendix B provides detailed recommendations for Linux Platform-Hardening for VOLTTRON users.

3.6 User administration of the platform, user interfaces, etc. (Interface 9)

VOLTTRON platforms support an easy-to-use, command-line user interface for platform administration. In addition the VOLTTRON team developed a VOLTTRON agent to act as a web-based management console. The management console allows a user to manage remote deployments without requiring a user to access individual boxes from a command line.

3.6.1 The vulnerability list discussed below is limited to discussion of the command-line interface only.

1. The platform is locally controlled via a Unix domain socket (9).

   R: Any user with local access to the system has the potential to send command and control messages to the platform.

   M: Access to the control socket is limited by file system permissions on the socket and the owner and group of processes attempting to connect to the socket are validated against an access control list in the platform configuration. The superuser may also be denied access.

3.6.2 The vulnerability list discussed below is limited to discussion of the web interface only.

1. Exposing VOLTTRON instance to the web.

   R: VOLTTRON does not require https communication.

M: The user should bind only to a locally available port and have a secondary web service (e.g., Apache) proxy the https connection.

F: Allow VOLTTRON to use https certificates directly.

R: Because the webserver is a platform service and agents installed on top of the platform can be aware of that web service, other agents can use the web service.

M: Agents by default are not able to talk with the web subsystem on the platform. Users should only install trusted web agents.

R: The web service allows an agent to serve files from the file system. This could potentially be an issue if the user running the VOLTTRON process has permissions with too wide of scope including sudo.

M: The user should know what the agent is able to serve from the file system.

M: VOLTTRON is not allowed to run as root.

2. VOLTTRON Central controls entire network of VOLTTRON instances.

   R: The user can access start/stop important agents on a remote instance.

   M: Access-level controls are provided (admin, reader) and implemented with specific agents not being allowed to be stopped even when admin (such as the platform connection embodied in the VOLTTRON Central Platform agent). Users belonging to the reader group cannot start, stop, deploy, or restart any VOLTTRON agents on any controlled VOLTTRON instances.

   F: Make it easy to modify user and agent access controls.

   R: VOLTTRON Central login becomes compromised.

   M: Change the password in the VOLTTRON Central configuration file and redeploy the agent.

   F: Enable two-factor authentication.

3. Potential DoS attack threat.

   R: A DoS attack from the web could potentially halt writing of data to the platform historian.

   M: Separate the movement of data into the historian into a secondary instances of VOLTTRON.

3.6.3 The vulnerability list below affects both the command-line and web interfaces.

1. VOLTTRON is written in Python and runs via the local Python installation.

R: Any vulnerability in Python could negatively affect the security of the platform and potentially the system.

M: The system administrator or owner must keep the system up-to-date with the latest security patches, especially with regard to the base Python installation.

2. VOLTTRON uses third-party Python libraries/packages.

   R: A vulnerability in VOLTTRON's third-party dependencies could allow VOLTTRON to be compromised.

   M: The VOLTTRON developers use mature and actively developed third-party packages with good reputations for stability and security; however, any complex code is likely to suffer bugs that may lead to compromise. While code written in Python is much less susceptible to certain attacks, third-party libraries should be regularly updated to the latest compatible version to take advantage of security patches.

4.0 Summary of VOLTTRON Security Features

VOLTTRON security features are based on the mitigations discussed in the previous section (denoted by M). These security features are summarized below:

- VOLTTRON is built on Linux to take advantage of its many built-in security features, such as powerful file system permissions, user management, Linux capabilities configuration, containers, control groups, and a first-class firewall.
- VOLTTRON accesses remote resources as securely as possible, utilizing the latest version of TLS protocols and with the largest key size available to both endpoints. Within VOLTTRON, OpenSSL[1] is used for TLS/SSL encrypted links. The system's OpenSSL libraries are kept as up-to-date as possible to prevent vulnerabilities such as HeartBleed.
- For multi-platform communication, VOLTTRON uses remote ØMQ[2] sockets using CurveZMQ[3] elliptical curve encryption. Keys must be configured for links to be encrypted.
- VIP implements encryption, authentication and authorization for both inter and intra-platform communications.
- Linux control groups (cgroups)[4] CPU and memory subsystems are used to limit excessive processor and memory usage.
- Platform control (Unix domain) socket utilizes a mixture of file permissions and access control lists to limit access to authorized users.
- Code is peer reviewed for correctness and security.
- Agent code and packages are signed and verified using RSA encryption with X.509 certificates. Unsigned code is not executed unless explicitly allowed by the administrator. [restricted]

[1] https://www.openssl.org/
[2] http://zeromq.org/
[3] http://curvezmq.org/
[4] http://en.wikipedia.org/wiki/Cgroups

Appendix A: Example Best Practice for Securing Building Control Networks

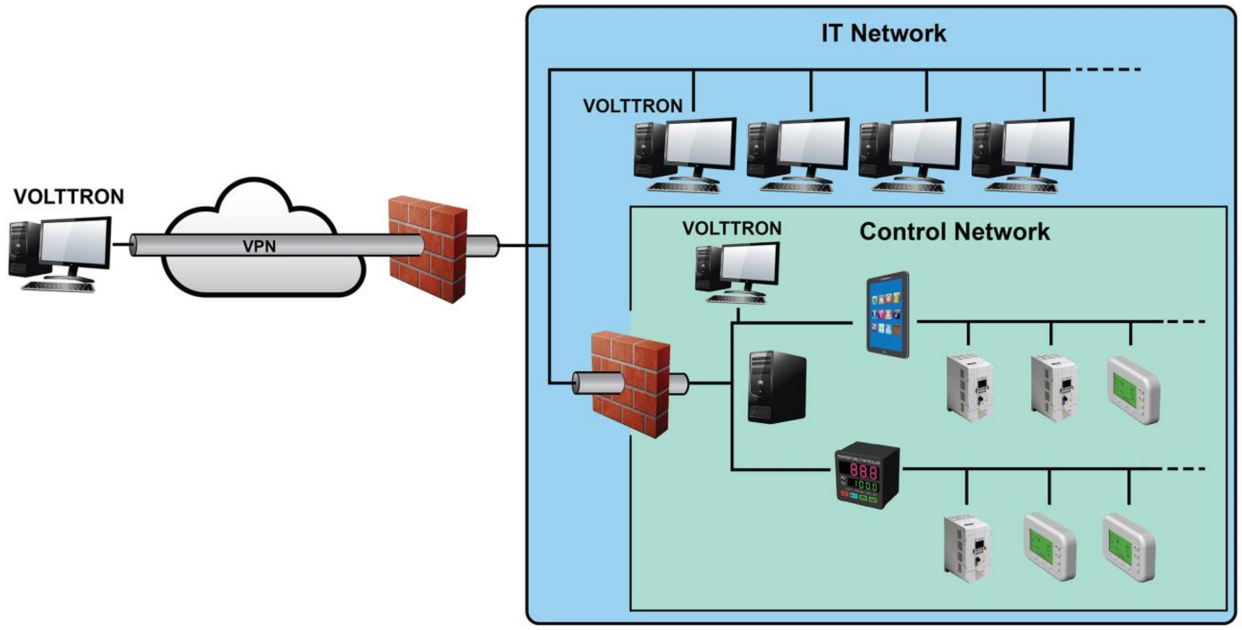

Figure 2. An example best practice of how to secure building control networks.

In Figure 2, the control network is completely segregated with its own network and a firewall. The control network is not connected to either the organizational IT (information technology) network or to the Internet directly. This type of segregation reduces the attack points for the building control network. The building control network is protected from the organizational IT network; the access to the control network from the IT network can be tailored by setting up the required firewall rules. Although Figure 2 shows two independent networks, this type of security can also be done using virtual local area networks. For details on this guidance, refer to Neilson (2013)[1]. Also, a VOLTTRON instance runs on the segregated building control network and an instance of VOLTTRON running on the IT network. These two instances of VOLTTRON can communicate securely using security features built into the VOLTTRON platform.

[1] Neilson, C. 2013. *Securing a Control Systems Network*. ASHRAE Journal. November 2013.

Appendix B: VOLTTRON Linux Platform-Hardening

The current version of this text is available at https://github.com/VOLTTRON/volttron/wiki/Linux-Platform-Hardening-Recommendations-for-VOLTTRON-users.

B.1 Introduction

VOLTTRON is an agent-based application development platform for distributed control systems. VOLTTRON itself is built with modern security principles in mind [security-wp] and implements many security features for hosted agents. However, VOLTTRON is built on top of Linux and the underlying Linux platform also needs to be secured in order to declare the resulting control system as "secure." Any system is only as secure as its weakest link. This section provides recommendations for hardening the underlying Linux platform that VOLTTRON uses. The cyber security strategy recommended in this document is based on risk management.

B.2 Linux System Hardening

The following are non-exhaustive recommendations for Linux hardening from the VOLTTRON team:

- Physical security: Keep the system in locked cabinets or a locked room. Limit physical access to systems and to the networks to which they are attached. The goal should be to avoid physical access by untrusted personnel. This could be extended to blocking or locking USB ports, removable media drives, etc. Drive encryption could be used to avoid access via alternate-media booting (off USB stick or DVD) if physical access cannot be guaranteed. However, drive encryption would require a passphrase to start system. Alternately, the Trusted Platform Module may be used, but the drive might still be accessible to those with physical access. Enable chassis intrusion detection and reporting if supported. If available, use a physical tamper seal along with or in place of an interior switch.
- Low-level device security: Keep firmware of all devices (including BIOS) up-to-date. Password-protect the BIOS. Disable unneeded/unnecessary devices including serial, parallel, USB, Firewire, etc. ports; optical drives; wireless devices, such as Wi-Fi and Bluetooth. Leaving a USB port enabled may be helpful if a breach occurs to allow saving forensic data to an external drive.
- Boot security: Disable automounting of external devices. Restrict the boot device. Disable PXE and other network boot options (unless that is the primary boot method). Disable booting from USB and other removable drives. Secure the boot loader. Require an administrator password to do anything but start the default kernel. Do not allow editing of kernel parameters. Disable, remove, or password-protect emergency/recovery boot entries.
- Security updates: Configure the system to automatically download security updates. Most security updates can be installed without rebooting the system, but some updates (e.g., shared libraries, kernel) require the system to be rebooted. If possible, configure the system to install the security updates automatically and reboot at a particular time. We also recommend reserving the reboot time (e.g., 1:30AM on a Saturday morning) using the Actuator Agent so that no control actions can happen during that time.
- System access only via secured protocols: Disallow all clear text access to VOLTTRON systems. No telnet, no rsh, no ftp, and no exceptions. Use ssh to gain console access, and scp/sftp to get files in and out of the system. Disconnect excessively idle SSH Sessions.

- Disable remote login for "root" users. Do not allow a user to directly access the system as the "root" user from a remote network location. Root access to privileged operations can be accomplished using “sudo” This adds an extra level of security by restricting access to privileged operations and tracking those operations through the system log.
- Manage users and usernames. Limit the number of user accounts. Use complex usernames rather than first names.
- Authentication. If possible, use two-factor authentication to allow access to the system. Informally, two-factor authentication uses a combination of "something you know" and "something you have" to allow access to the system. RSA SecurID tokens are commonly used for two-factor authentication but other tools are available. When not using two-factor authentication, use strong passwords and do not share accounts.
- Scan for weak passwords. Use password cracking tools such as John the Ripper (http://www.openwall.com/john/) or nmap with password cracking modules (http://nmap.org) to look for weak passwords.
- Utilize Pluggable Authentication Modules (PAM) to strengthen passwords and the login process. We recommend:
  - pam_abl: Automated blacklisting on repeated failed authentication attempts
  - pam_captcha: A visual text-based CAPTCHA challenge module for PAM
  - pam_passwdqc: A password strength checking module for PAM-aware password changing programs
  - pam_cracklib: PAM module to check the password against dictionary words
  - pam_pwhistory: PAM module to remember last passwords

- Disable unwanted services. Most desktop and server Linux distributions come with many unnecessary services enabled. Disable all unnecessary services. Refer to your distribution's documentation to discover how to check and disable these services.

- Just as scanning for weak passwords is a step to more secure systems, regular network scans using nmap to find what network services are being offered is another step towards a more secure system. Use nmap or similar tools very carefully on BACnet and modbus environments. These scanning tools are known to crash/reset BACnet and modbus devices.

- Control incoming and outgoing network traffic. Use the built-in host-based firewall to control who/what can connect to this system. Many iptables frontends offer a set of predefined rules that provide a default deny policy for incoming connections and provide rules to prevent or limit other well-known attacks (i.e., rules that limit certain responses that might amplify a DoS attack). The uncomplicated firewall, or ufw, is a good example: If the system administrators for the VOLTTRON device are all located in 10.10.10.0/24 subnetwork, then allow SSH and SCP logins from only that IP address range. If the VOLTTRON system exports data to a historian at 10.20.20.1 using TCP port 443, allow outgoing traffic to that port on that server. The idea here is to limit the attack surface of the system. The smaller the surface, the easier it is to analyze the communication patterns of the system and detect anomalies. While some system administrators disable network-based diagnostic tools such as ICMP ECHO responses, the VOLTTRON team believes that this hampers usability. For example, monitoring which incoming and outgoing firewall rules are triggering can be accomplished with this command: `watch --interval=5 'iptables -nvL | grep -v "0     0"'` .

- Rate limit incoming connections to discourage brute force hacking attempts. Use a tool such as fail2ban (http://www.fail2ban.org/wiki/index.php/Main_Page) to dynamically manage firewall rules to rate limit incoming connections and discourage brute force hacking attempts. For example, sshguard (http://www.sshguard.net/) is similar to fail2ban but only used for ssh connections. Further rate limiting can be accomplished at the firewall level. Restrict the number of connections used using a single IP address to the server using iptables. Only allow four ssh connections per client system:

  ```
  iptables -A INPUT -p tcp --syn --dport 22 -m connlimit --
  connlimit-above 4 –j DROP
  ```

  Limit the number of connections per minute. The following example will drop incoming connections if an IP address makes more than 10 connection attempts to port 22 within 60 seconds:

  ```
  iptables -A INPUT -p tcp –dport 22 -i eth0 -m state --state NEW -
  m recent --set

  iptables -A INPUT -p tcp –dport 22 -i eth0 -m state --state NEW -
  m recent\ --update —seconds 60 --hitcount 10 –j DROP
  ```

- Use a file system integrity tool to monitor for unexpected file changes. For example, tripwire (http://sourceforge.net/projects/tripwire/) can monitor the filesystem for changed files. Another file integrity checking tool to consider is Advanced Intrusion Detect Environment (http://aide.sourceforge.net/).

- Use file system scanning tools periodically to check for exploits. Available tools such as checkrootkit (http://www.chkrootkit.org), rkhunter (http://rkhunter.sourceforge.net), and others can check for known exploits on a periodic basis and report their results.

- VOLTTRON does not use Apache or require it. If Apache is being used, use mod_security and mod_evasive modules.

B.3 System Monitoring

- Monitor system state and resources. Use a monitoring tool such as Xymon (http://xymon.sourceforge.net) or big brother (http://www.bb4.org/features.html) to remotely monitor the system resources and state. Set the monitoring tools to alert the system administrators if anomalous use of resources (e.g., connections, memory) are detected. An administrator can also use Unix commands such as netstat to look for open connections periodically.

- Watch system logs and get logs off the system. Use a utility such as logwatch (http://sourceforge.net/projects/logwatch/files/) or logcheck (http://logcheck.org) to get daily summary of system activity via email. For Linux distributions that use systemd, use journalwatch (http://git.the-compiler.org/journalwatch/) to accomplish the same task. Additionally, use a remote syslog server to collect logs from all VOLTTRON systems in the field at a centralized location for analysis. A tool such as splunk is ideal for this task and comes with many built-in analysis applications. Another benefit of sending logs remotely off the platform is the ability to inspect the logs even when the platform may be compromised.

- An active intrusion sensor such as PSAD (http://cipherdyne.org/psad/) can be used to look for intrusions as well.

B.4 Security Testing

Every security control discussed in the previous sections must be tested to determine correct operation and impact. For example, if we inserted a firewall rule to ban connections from an IP address such as 10.10.10.2, then we need to test that the connections actually fail. In addition to functional correctness testing, common security testing tools such as Nessus (http://www.tenable.com/products/nessus) and nmap should be used to perform cyber security testing.

B.5 Conclusion

No system is 100 percent secure unless it is disconnected from the network and is in a physically secure location. The VOLTTRON team recommends a risk-based cyber security approach that considers each risk and the impact of an exploit. Mitigating technologies can then be used to mitigate the most impactful risks first. VOLTTRON is built with security in mind from the ground up, but it is only as secure as the operating system on which it runs. This document is intended to help VOLTTRON users secure the underlying Linux operating system to further improve the robustness of the VOLTTRON platform.