

VOLTTRON™ 3.0 Technical Overview and Features

BRANDON CARPENTER

Pacific Northwest National Laboratory

Software Framework for Transactive Energy: VOLTTRON™, VTARI, Arlington, VA

VOLTRON Team

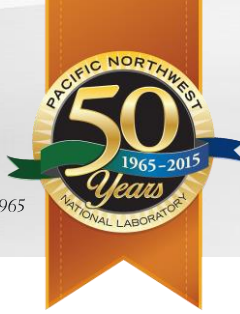
Software Development Team

- ▶ Bora Akyol
- ▶ Jereme Haack
- ▶ Brandon Carpenter
- ▶ Kyle Monson
- ▶ Craig Allwardt
- ▶ Poorva Sharma
- ▶ Tim Kang
- ▶ Robert Lutes
- ▶ Casey Neubauer
- ▶ Dan Johnson

Application Development Team

- ▶ Srinivas Katipamula
- ▶ Robert Lutes
- ▶ Wooyun Kim
- ▶ Rick Pratt
- ▶ Carl Miller
- ▶ Weimin Wang
- ▶ Siddartha Goyal
- ▶ Michael Brambley
- ▶ Lucy Huang
- ▶ Chad Corbin
- ▶ He Hao

VOLTTRON 3.0

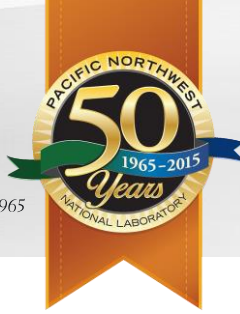


- ▶ Improve the modularity, flexibility and manageability of the VOLTTRON platform
 - Lets people use whatever technology they want
 - Makes it easier to contribute back new drivers, storage/historian strategies, other services
- ▶ Bring VOLTTRON closer to acceptance by vendor community and for commercial deployments
 - Need to gain visibility into system
 - Upgrade remotely
 - Easy way of seeing the status/resources of the platform especially when managing multiple systems
 - Address feedback from FY14 User and Vendor engagements

Core Libraries

- ▶ **gevent**
 - Fast, asynchronous networking
 - Built-in support for ZeroMQ
 - Easy API
 - Can be used with non-aware libraries using monkey-patching
- ▶ **pyzmq (ZeroMQ)**
 - High-level networking library with Berkeley sockets-like API
- ▶ **simplejson**
 - JSON implementation; faster than built-in *json* module
- ▶ **BACpypes**
 - Python package providing BACnet support
- ▶ **pymodbus**
 - Python package providing Modbus support

Improved Message Bus



- ▶ Based on community feedback and security review
- ▶ Limitations of previous message bus
 - Requires opening two sockets
 - Socket for publishing
 - Socket for subscribing
 - Possible to spoof messages
 - No support for private messaging
 - Not ideal for peer-to-peer communications
 - Not extensible

Leveraging ZeroMQ Super Powers

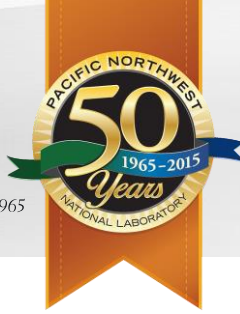
- ▶ Already using ZeroMQ (aka ØMQ)
- ▶ Encryption provided via CurveMQ
- ▶ Provides authentication via ZAP (ZeroMQ Authentication Protocol)
- ▶ Other features of interest:
 - Built-in message framing
 - Automatic reconnection
 - Zero-copy message passing (in process)
 - Single ØMQ socket for multiple endpoints (inproc, ipc, and tcp address types)
 - Abstractions for common communications patterns
- ▶ Support for many languages and operating systems
- ▶ Active and diverse developer and user community



<http://zeromq.org>



VOLTTRON™ Interconnect Protocol



- ▶ VIP defined in RFC-style specification:
 - <https://github.com/VOLTTRON/volttron/wiki/VIP>
- ▶ Inter-agent message routing
- ▶ Layers of security
 - Encryption
 - Authentication
 - Authorization
 - Attribution (spoof-proof)
- ▶ Private, peer-to-peer messaging
- ▶ Extensible and flexible
 - Easy to add new subsystems (protocols)
 - Provides new ways to arrange platforms
- ▶ Homogeneous communications for agents, control, multinode, etc.

VIP Frame Layout

send:

sender	recipient	"VIP1"		request ID	subsystem	...
--------	-----------	--------	--	------------	-----------	-----

recv:

recipient	sender	"VIP1"	user ID	request ID	subsystem	...
-----------	--------	--------	---------	------------	-----------	-----

- ▶ Sender/recipient frames contain peer identities
- ▶ Protocol signature indicates protocol and version: VIP1
- ▶ User ID provided by auth service
 - Can be used to authorize requests
- ▶ Request ID associates responses with request
- ▶ Subsystem indicates the protocol and handler
- ▶ Optional data frames are specific to subsystem

VIP Routing

peer	peer	"VIP1"	user ID	request ID	subsystem	...
------	------	--------	---------	------------	-----------	-----

- ▶ Alice sends message for Bob to *router*

"bob"	"VIP1"		"0001"	"ping"
-------	--------	--	--------	--------

- ▶ *Router* receives message (sender frame automatically prepended)

"alice"	"bob"	"VIP1"		"0001"	"ping"
---------	-------	--------	--	--------	--------

- ▶ *Router* swaps sender and recipient, adds User ID, and resends

"bob"	"alice"	"VIP1"	"1234abcd"	"0001"	"ping"
-------	---------	--------	------------	--------	--------

- ▶ Bob receives Alice's message from *router*

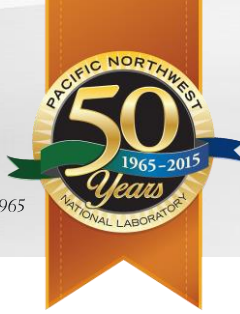
"alice"	"VIP1"	"1234abcd"	"0001"	"ping"
---------	--------	------------	--------	--------

VIP Subsystem: pubsub

- ▶ Platform pub/sub service
 - Global service allows for discovery and platform-level messaging
- ▶ Message format:
 - Topic
 - UTF-8 encoded string
 - /-separated components
 - Headers
 - JSON serialized dictionary (mapping)
 - Body
 - Zero or more ZeroMQ frames
- ▶ Improvements with 3.0
 - Source attribution (not spoofable)
 - Unlimited per-agent buses
 - Decentralized



VIP Subsystem: RPC



- ▶ Remote procedure calls via JSON-RPC 2.0
 - Specification at <http://www.jsonrpc.org/specification>
 - Safe, expressive, simple, well-supported, etc.
 - Supports one-way notifications
- ▶ Extended to support simultaneous use of list (*args) and keyword (**kwargs) arguments
- ▶ Export agent methods with export() decorator
- ▶ Calls handled asynchronously (spawned in own greenlet)
- ▶ Calling remote procedure returns AsyncResult
 - Wait for results
 - Set callback to handle results
- ▶ Discover exported methods with inspect()
 - Also used to query parameters, return value, documentation, etc.

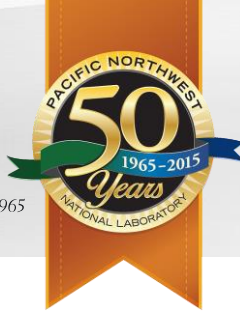


Other VIP Subsystems



- ▶ error
 - Protocol for communicating routing errors
 - EHOSTUNREACH: no route to peer (peer not connected)
 - EAGAIN: temporary failure because of full buffers
- ▶ hello
 - Get version and identity (router and peer) information from router
- ▶ ping
 - Send ping requests to any agent
- ▶ query
 - Query router for properties (e.g. TCP addresses)
- ▶ channel
 - Tunnel ZeroMQ frames between agents

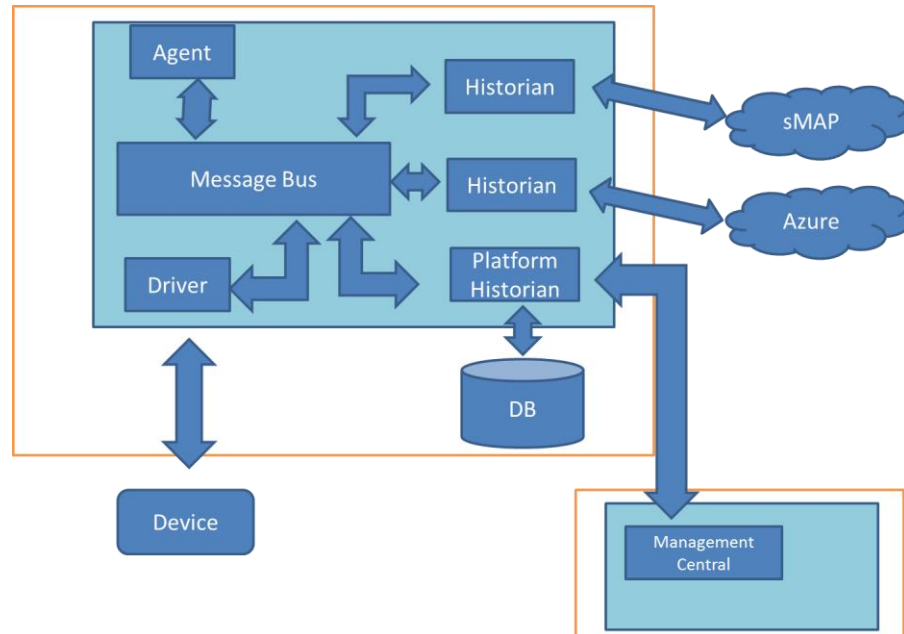
VIP Compatible with 2.x Agents



- ▶ Compatibility layer
 - Relays 2.x pub/sub messages via VIP
 - Completely modular
 - Can be easily removed
- ▶ 2.x agents work without modification
- ▶ 2.x legacy support will be removed in subsequent release

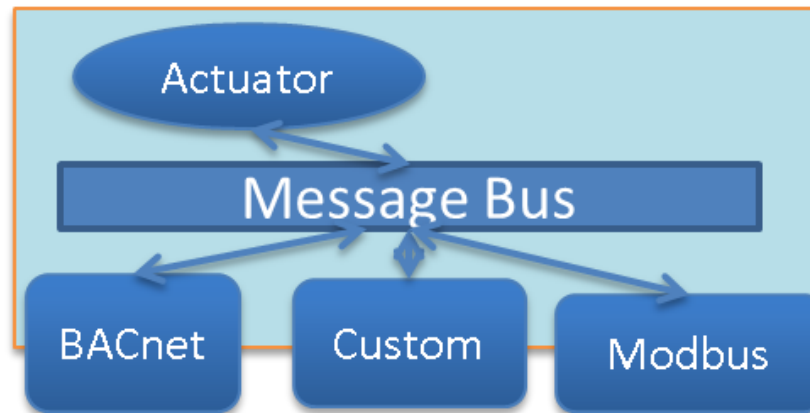
Generic Historian

- ▶ Enables support for many different methods of storing log and device data
 - Includes local storage
- ▶ Automatically preserves unarchived data until it can be stored successfully
- ▶ Allows alternatives to sMAP
- ▶ API implemented in abstract base class



Updated Driver Framework

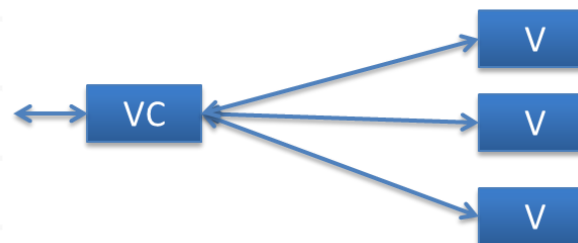
- ▶ Removes the twisted framework and sMap library requirements
- ▶ Should allow the use of drivers on lower powered devices
- ▶ Removes HTTP side channel communication
 - All inter-agent communication now happens via VIP
- ▶ Further simplifies the implementation of new drivers



VOLTTRON™ Management Central

- ▶ Web-based management console
- ▶ Simplifies managing a large number of deployed platforms
- ▶ All functions of volttron-ctl command-line control utility
 - Install, start, stop, status agents
- ▶ Plus central configuration, control, and communications
- ▶ Rendezvous point
- ▶ Dashboard to easily monitor health and status of agents

VOLTTRON™	
Platforms	
Beagle EV2	ab0c25fd-3718-4222-ad25-ae44d2a5cdaa Agents: 3 running, 0 stopped, 4 installed
Panda1 WH1	b8efd9f20-825c-4382-900d-003eaa1d1be Agents: 3 running, 0 stopped, 3 installed
Pandas 2 EV1	8785999d-e298-47b2-bb23-8a369240d220 Agents: 3 running, 0 stopped, 4 installed
Rock WH2	c4b3282e-7517-4a8b-b76d-cfe5833ed7a6 Agents: 3 running, 0 stopped, 4 installed
VC Platform	b5c34dfd-88ed-412b-8a4c-aa4d60ec7ae5 Agents: 3 running, 0 stopped, 4 installed



Status of 3.0 Features

- ▶ Currently working toward Beta release
 - Please submit issue requests through GitHub
 - <https://github.com/VOLTTRON/volttron/issues>
- ▶ New features are in master branch
 - <https://github.com/VOLTTRON/volttron/tree/master>
- ▶ Final release scheduled for September



QUESTIONS?

► VOLTTRON Resources

- Wiki: <https://github.com/VOLTTRON/volttron/wiki>
- Email: volttron@pnnl.gov
- Bi-weekly office hours