In-Memory Computing SUMMIT NORTH AMERICA 2018

Ultimate Guide to Successful Cross-Platform Deployments with Apache Ignite

Pavel Petroshenko

Electric Imp

Agenda

- Apache Ignite Clients
- Data interoperability in Ignite
- Binary Client Protocol
- Cross-platform deployment demo

Clients and Servers

- Server nodes
  - Participate in caching
  - Compute jobs execution
  - Stream processing
  - etc.
- Client nodes
  - Provide ability to connect to server nodes
  - Client nodes are primarily used to run *Native (aka Thick)* clients

Native (aka Thick) Clients

- Have access to the full set of the Ignite APIs
  - Near caching
  - Transactions
  - Compute
  - Streaming
  - Services
  - etc.
- Require the server nodes to exist in the topology
  - Server mode discovery can weaken this requirement
- Other requirements to keep in mind
  - Client application should handle reconnects (can have a new ClusterNode id)
  - Server should manage the outbound traffic to the “slow” clients

Thin Clients

- Connect to clusters via a socket connection
  - Connects to a specific “proxy” server node
  - Sends all cache requests to the proxy, which re-routes data to the right server
- Does not require a client node to be run
- Does not become a part of the topology
- Can’t run the compute jobs
- Use the Binary Client Protocol for communication with server nodes
- Can be implemented in any programming language
- Thin Clients to be released in Apache Ignite v2.7
  - C++
  - Node.js
  - Python
  - PHP

Data Interoperability
BinaryObject format

- BinaryObject – cross-platform format for objects serialization
  - Allows for arbitrary field access from serialized form
    - *No need to have key/value type implementations on the server side*
  - Allows to dynamically modify object structures
    - *Might be useful to support multiple object type versions*
  - Allows to construct objects based on type name
    - *Dynamic type construction*
  - Supports SQL queries
- BinaryObject Limitations
  - Fields or types with the same name hash are not allowed
    - *Applicable to all the levels of class hierarchy*
  - Only default binary marshaller can be used

Binary Objects: Best Practices

- Zero Downtime Principle
  - No need to keep object classes on the server nodes thanks to the binary format
  - Use BinaryObjectBuilder and BinaryObject wrappers to access data on the servers
- Reduce Space Consumption With Serialization Tuning
  - Use BinaryRawWriter for more compact fields serialization, if you don’t need them in the SQL requests – footprint optimization

Binary Thin Client Protocol

- Allows Thin Client applications to interact with a cluster
- Application connects to a “proxy” server node via a socket connection
  - Connectivity issues should be handled on the client side
- Defines the format of client-server connection handshake
  - Verification that client and server versions match
  - Credentials exchange and authentication (optional)
- Defines the format of data/messages
  - Little-endian byte ordering
  - Header and body format
  - Request and response format
- Is based on the BinaryObject format for data representation
  - Keys, values, complex objects

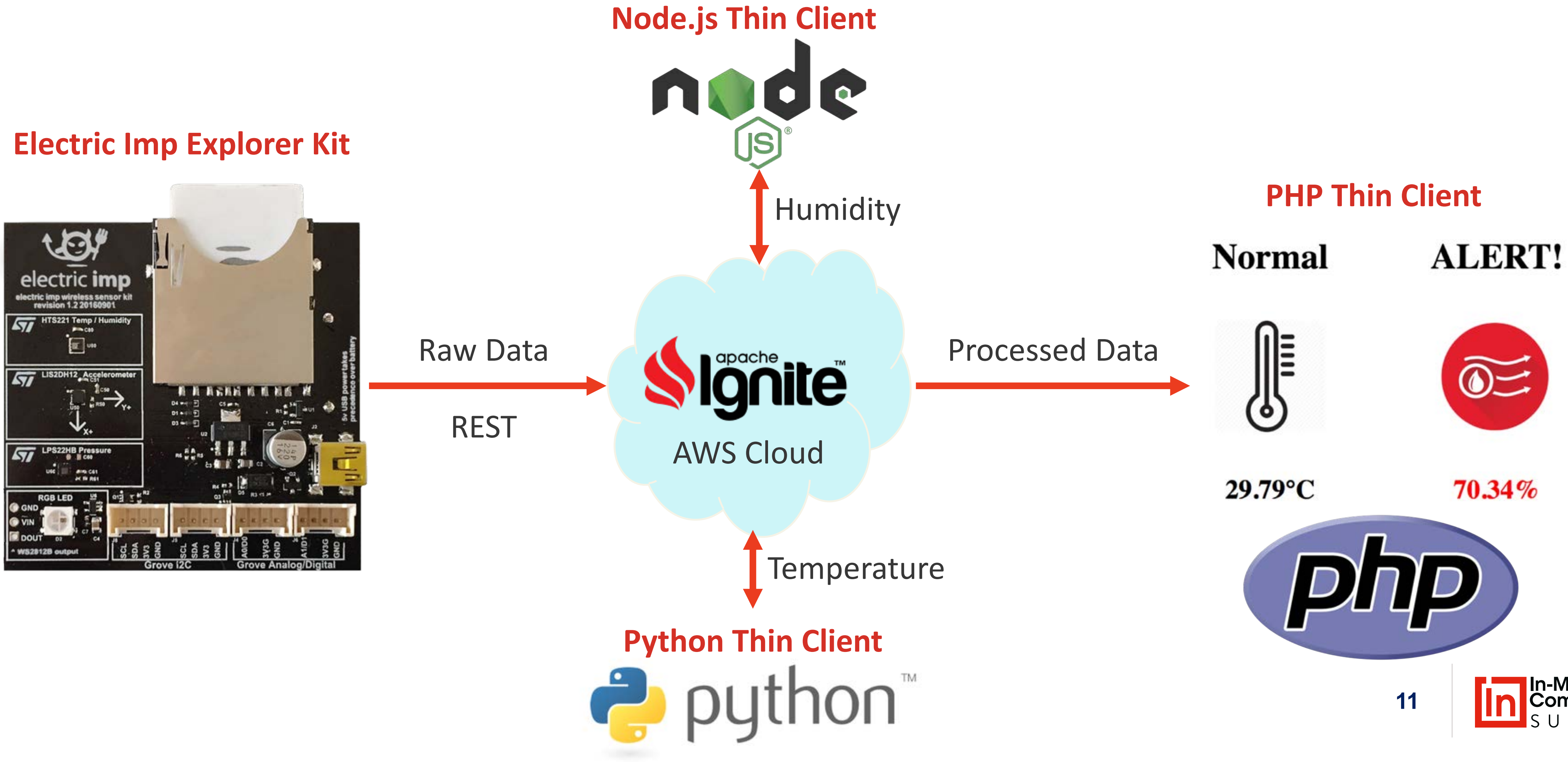
Binary Client Protocol (cont.)

- Client operations
  - Key-Value operations
  - SQL and scan queries
    - *Scan with filters may be limited at this point*
  - Binary-type operations
  - Cache configuration operations

Cross-Platform Deployment Demo

- Apache Ignite cluster running on an AWS instance
- 4 different platforms share the same data set
  - Real-time data pushed from the Electric Imp Explorer Kit
  - Data processing with sample PHP, Python and Node.js Thin Client applications
- Cross-platform and Cross-APIs operations
  - Key-value primitive type put/get
  - Complex binary objects manipulation

Demo Architecture



Node.js Thin Client

Humidity

Electric Imp Explorer Kit

Raw Data

REST

Apache Ignite

AWS Cloud

Processed Data

PHP Thin Client

Normal

ALERT!

29.79°C

70.34%

Temperature

Python Thin Client

Thank You!

Questions?

Useful Links

- https://apacheignite.readme.io/docs/clients-vs-servers
- https://apacheignite.readme.io/docs/binary-marshaller
- https://apacheignite.readme.io/docs/binary-client-protocol
- https://apacheignite.readme.io/docs/thin-clients
- https://apacheignite.readme.io/docs/rest-api
- https://developer.electricimp.com/hardware/resources/reference-designs/explorerkit