



A Virtual Conference presented by AIM & RAIN
9 - 10 December 2020



RAIN[®]
ALLIANCE



A Virtual Conference presented by AIM & RAIN
9 - 10 December 2020

Thank you to our sponsors

Platinum



Gold



Silver



SIMPLYRFID



RAIN in Action

RCI Introduction

S1 Reader Introduction

Demonstration

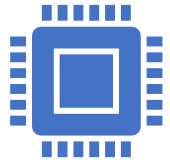


***Clairvoyant
Technology***

CLARIFY YOUR VISION



RAIN[®]
R F I D



RAIN Communication Interface (RCI)

Introduction

- ASCII JSON based protocol
 - Command/Response
 - Asynchronous Events
- Base RCI High Level Operations
 - Reader Configuration
 - DHCP, Event fields, Heartbeat, ...
 - Reader Info
 - Version, Model, Serial Number, ...
 - Read Zones (akin to LLRP AISpec)
 - Antennas, Triggers, Duty Cycle, Power, ...
 - Spot Profiles (akin to LLRP AccessSpec)
 - Tag Operations, Read Zone Targets, ...

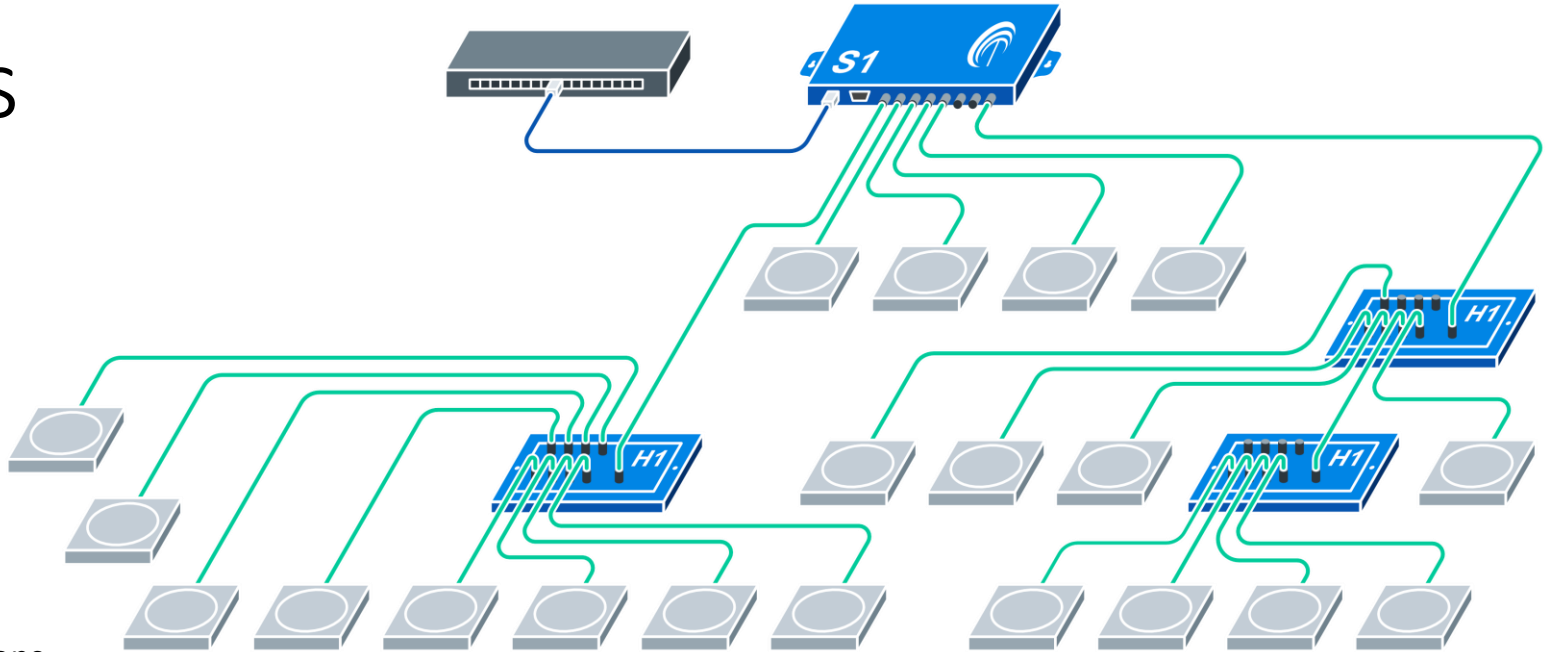
S1 Interfaces

- RCI v4
- Web / Web Sockets
- Discovery
 - Bonjour and S1 Specific Multicast
- MQTT Publisher
- Reader Management SNMP
- Other Licensable Interfaces / Capabilities
 - LLRP
 - Binary applications running on S1
 - ...
- Contact
 - info@clair-tech.com



S1 RCI v4 Features

- Out of Box
 - 2 Read Zones
 - 2 Spot Profiles
 - 1 MQTT or Publish Point Connection
- Licensable Max (Arbitrary):
 - 16 Read Zones
 - 8 Spot Profiles
 - 8 MQTT or Publish Point Connections
- SetConnection Object
 - Simultaneous connections with permissions
 - Events are filterable
- Custom MQTT / Publish Point Object
- Custom Crypto Reader / Tag Security Object
 - Supports “Sames” (tags with “Same” observable EPC because of Untraceable)
- Custom Tag Database
- Custom Logging
- Custom Events
 - _ReaderLogEvent, _ChangeEvent, _StatusEvent, _UserEvent



S1 RCI Features

Connectivity

RCI Ports (Configurable)

- TCP Port 51000 (Insecure), 51100 (TLS)
- Web Sockets
 - http://<reader_ip>/rci (Insecure)
 - https://<reader_ip>/rci (TLS)

Import File (Configurable)

- TCP Port 3034 (Insecure), 30034 (TLS)
- Imports key files, license files, certificate files, root CA files

MQTT or Publish Point

- Reader Starts Connection (Not Listening Port)

Java Host GUI with RCI Command Builder

- Uses Secure Reader Port (Or Selectable)
- RCI Learning Menus
 - Command Builder
 - Menus for Profiles, Read Zones, Info, Configuration, MQTT/Publish Point
 - Menus display JSON used by the GUI to configure and get information from the reader to help learn

RCI SDKs

- Java, .NET, Python, C (intended for licensable ability to run binary applications on S1)

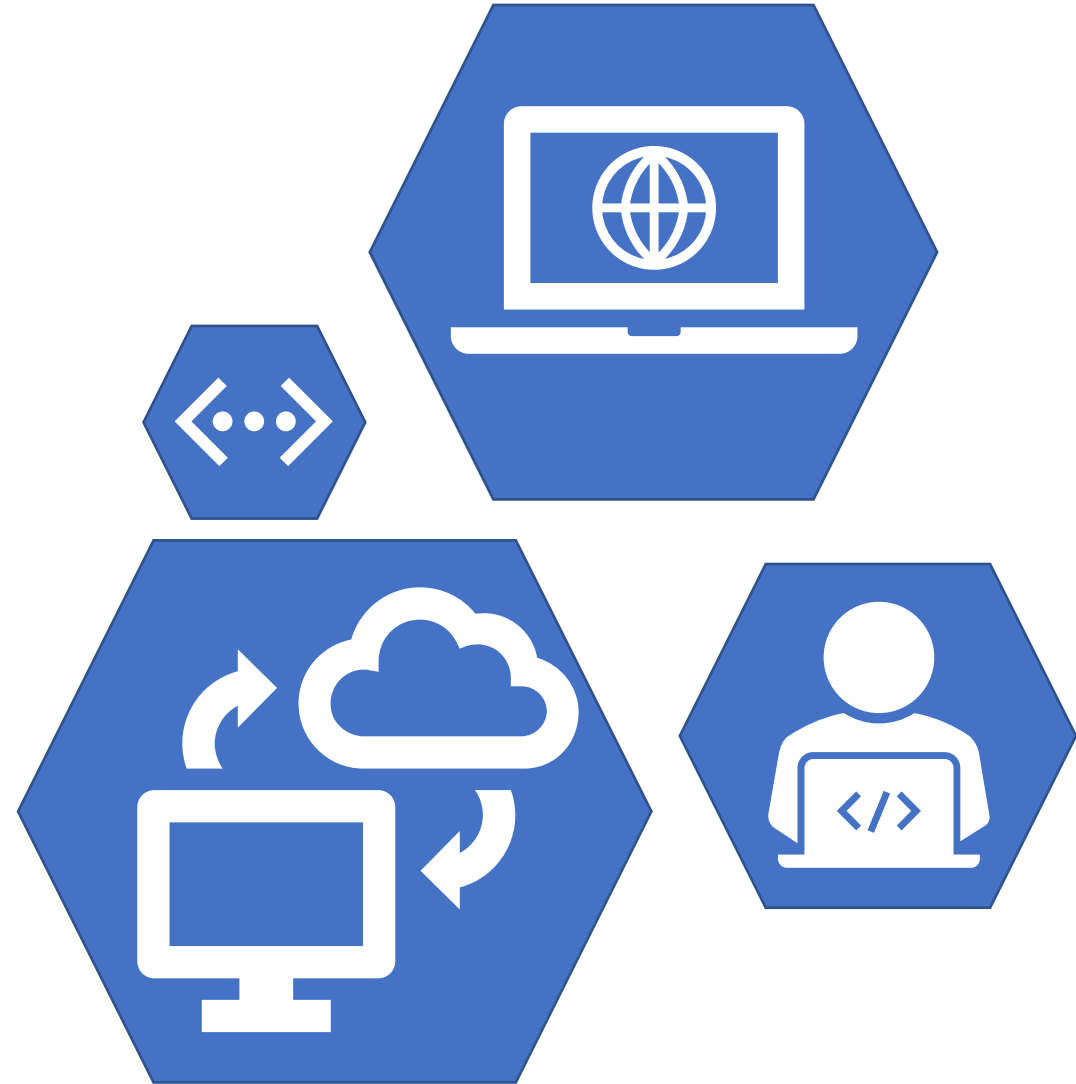
S1 Custom RCI Features

MQTT

- Setup S1 to publish to MQTT broker
- AWS and Google IoT tutorials available
- Choose types of events to publish and names to publish to
- Supports 8 simultaneous MQTT or Simple Publish connections

Simple Publish Protocol

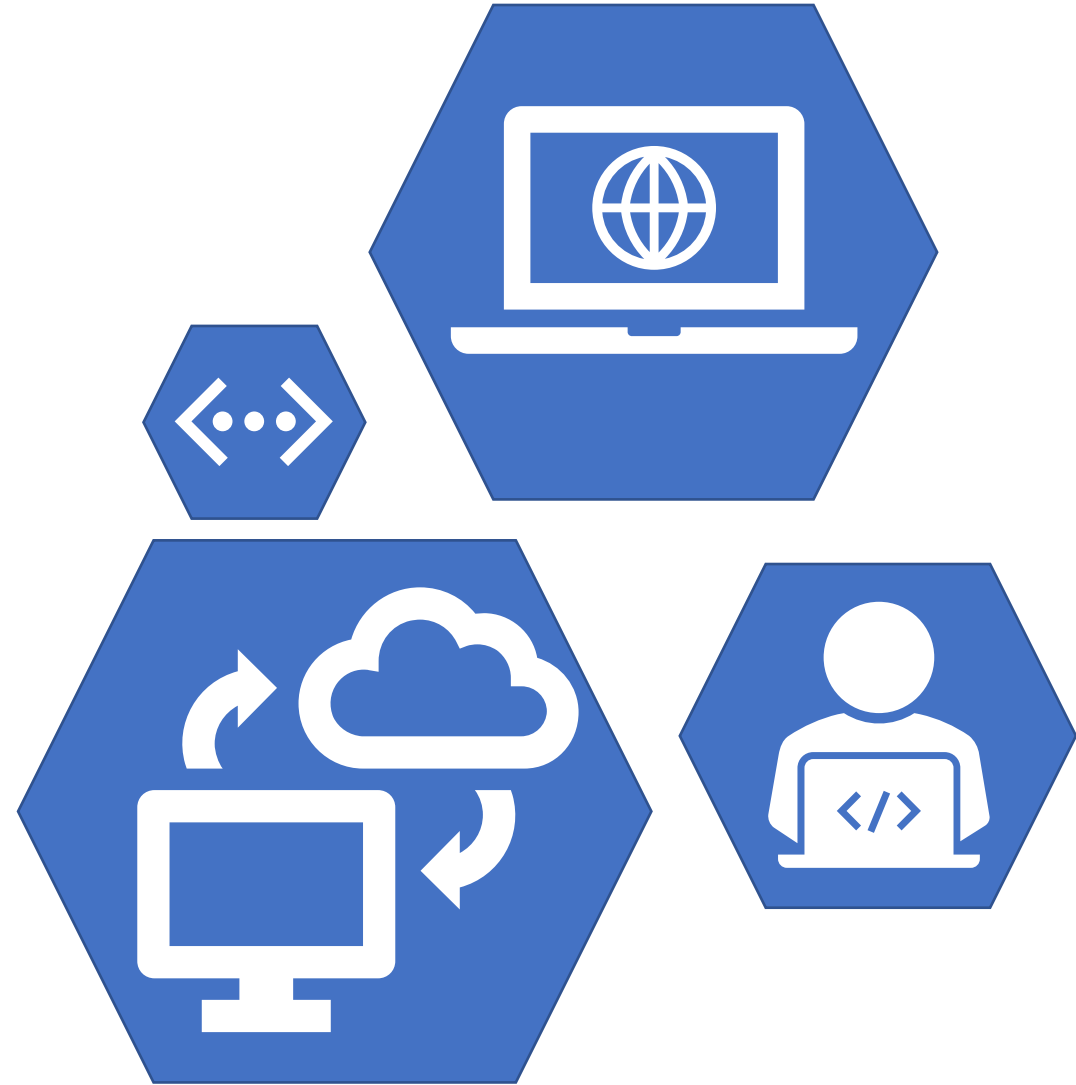
- Setup S1 to publish RCI events to TCP server
- Choose types of events to publish
- Specify “broker” via IP address and port (optional SSL/TLS)



S1 RCI Features

Crypto Reader / Tag Security

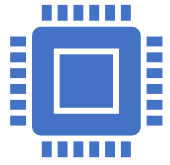
- Supports TAM1 and TAM2 Authentication
- Supports Handling Untraceable EPC
- Reports “Sames”



S1: Secure RFID Edge Server

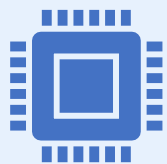
- Real-time, on-board, secure RAIN authentication and traceability
- S1 Chain of Trust
 - Hardware authenticated secure boot loader
 - Secure boot loader authenticates operating system before loading
 - Operating system authenticates applications & data before loading
- Key management (secure key insertion, use, and storage)
- Tamper-evident, tamper-detecting, and tamper-responsive design
- Based on Analog Devices' Lockbox™ and Microchip, Inc. Trust Platform™
- Current International Standards
 - FIPS 140-3
 - ANSI X9.24-1-2017
 - ISO 13491





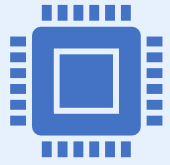
Crypto Reader Key Storage

- Keys never stored in nonvolatile or L3 memory unencrypted
 - Keys only available through on chip L1/L2 memory which is protected using Analog Devices Lockbox technology (<https://www.analog.com/en/design-center/processors-and-dsp/lockbox-secure-technology.html>)
- Keys generated using ECDH premaster secrets and random numbers
 - Customer holds their own private key and provides the reader with their public key
 - Reader holds its own private key and provides customer with public key
 - Private key held in Microchip secure crypto chip that the reader can only write to, but not read and can only write encrypted
 - <https://www.microchip.com/wwwproducts/en/ATECC608A>
- Tag specific key derivation through a SHA hash function with the EPC and secret master key
 - Use two keys (Key ID 0 and 1) generated from two master keys and EPC



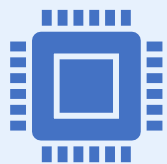
List of Examples

- Connect to RCI port using Teraterm and GUI
 - Send commands through teraterm
 - Receive events through teraterm
- Examples of Configuring Profiles, Read Zones and JSON send and received
 - Run different examples of Reading and Writing User Data
- Examples of Using MQTT
 - Send to public MQTT broker anyone watching can subscribe
 - Allow audience to observe events locally on their own devices
- Examples of Sames and Crypto Reader Tag Security
 - Anyone subscribed via MQTT can continue to observe
 - Show “Sames” with tag security turned off
 - Show “Sames” getting properly expanded to unique EPCs when tag security turned on



Tutorials Available

- Introduction to RCI Tutorial
- Introduction to RCI Publish Points Tutorial
- Introduction to MQTT Publish Points Tutorial
 - Amazon AWS Example
 - Google IoT Example
- Additional Tutorials (Help with Tutorials Used Above)
 - Host Software Tutorial
 - Setting Up TLS Security Tutorial



S1 RCI Java SDK

- Uses JSON library from org.json.simple (json-simple-1.1.1)
- Java Object For Every RCI Command and Report
 - E.g. SetCfgCmd, SetCfgReport, HBReport, GetInfoCmd, GetInfoReport
- Java Objects For Managing Connections / Asynchronous Events
 - RciConnect
 - RciEventHandling
 - RciEvent
- .NET, Python, and C Have Similar Behavior / Capabilities

S1 RCI SDK Event Handling Objects 1

```
public interface RciEventHandling
{
    public void RciEventHandlingCallback(RciEvent rciEvent);
    public List<String> RciGetReportFilter();
}
```


S1 RCI SDK Event Handling Objects 2

```
public class RciEvent
{
    public JSONObject reportObject; ← org.json.simple.JSONObject
    public String reportString;

    public RciEvent(JSONObject reportObject, String reportString) {
        this.reportObject = reportObject;
        this.reportString = reportString;
    }
}
```

S1 RCI SDK HB Event Handler Example 1

```
class HBEventHandler implements RciEventHandling {  
    boolean hbSeen = false;  
    HBReport report = null;  
    int hbCount = 0;  
  
    ArrayList<String> desiredReports = new ArrayList<String>() {  
        {  
            add("HB");  
        }  
    };  
  
    @Override  
    public List<String> RciGetReportFilter() {  
        return desiredReports;  
    }  
}
```



S1 RCI SDK HB Event Handler Example 2



```
@Override
public void RciEventHandlingCallback(RciEvent rciEvent) {
    String reportString = (String) rciEvent.reportObject.get("Report");
    if (reportString != null) {
        if (reportString.compareTo("HB") == 0) {
            report = new HBReport(rciEvent.reportObject);
            hbSeen = true;
            hbCount++;
        }
    }
}
```


S1 RCI SDK Connect Example

```
String ipAddress = "10.172.0.124";  
HBEventHandler hbHandler = new HBEventHandler();  
RciConnect testConn = new RciConnect( ipAddress, hbHandler);  
try {  
    testConn.Connect(5000);  
} catch (Exception e) {  
    fail("Can't connect");  
}
```

User Defined Event Handler



Connection Timeout (5000 ms)



S1 RCI SDK GetCfg Example

```
GetCfgCmd getCfgCmd = new GetCfgCmd(rciConnect);
getCfgCmd.setFields(getCfgCmd.FIELDS_HBFIELDS);
getCfgCmd.setFields(getCfgCmd.FIELDS_HBPeriod);
GetCfgReport getCfgReport = getCfgCmd.sendObject(5000);
if (getCfgReport.getErrorCode() != 0) {
    getCfgReport.printErrors("    ");
    fail("Failed sending cfg report");
}
```

```
Long hbPeriod = getCfgReport.getFieldsHBPeriod();
JSONArray hbFields = getCfgReport.getFieldsHBFields();
for (int i=0; i< hbFields.size(); i++) {
    String field = (String) hbFields.get(i);
}
```

Thank you for Attending



A Virtual Conference presented by AIM & RAIN
9 - 10 December 2020

Presentations will be available on-line soon. You will receive an email with a link when they are available.