Genoma: Distributed Provenance as a Service for IoT-based Systems

Authors: Nanjungud C Narendra, Anshu Shukla, Sambit Nayak (Ericsson Research Bangalore, India); Asha Jagadish, Rachana Kalkur (Manipal University, Bangalore, India)
Agenda
— Introduction
— Background
— *Genoma* Data Model
— *Genoma* Architecture
— *Genoma* Implementation
— Key Research Challenges and Future Work
IoT systems produce voluminous streaming data
- Streaming data is processed on the edge (e.g., in a laptop in a factory) and on the cloud
  - This processing yields insights into actuations, reconfigurations, etc., in the factory
- But this processing needs to be tracked to determine how the data was used, and also to what extent the data itself is useful
  - This is called *Provenance*
- Typical provenance solutions suffer from two drawbacks making them unsuitable for IoT systems:
  - Embedded into systems; not possible to delink from them
  - Centralized in nature
- Hence our system *Genoma*:
  - Provenance as a Service; agnostic of underlying IoT system
  - Works on both Edge and Central Cloud
    - Works even with intermittent connectivity between Edge and Cloud
Provenance can be represented as per the lifecycle shown:

- First model the provenance data to be captured
- Data is captured and then recorded & stored
- Then retrieved by the user, who then performs inferencing on it in order to understand and extract useful insights.
- In addition, since provenance data is expected to be an immutable record of usage, it is expected to be read-only; hence it can only be stored, archived or deleted, not overwritten.
Two kinds of distributed provenance: workflow and data
  - Workflow provenance builds on data provenance: records the workflows that were created and
eexecuted to act on the data, e.g., streaming, processing, transformation, etc.
  - *Genoma* is a directed acyclic graph (DAG)
    - Vertices represent entities; will have ID, type and version number
    - Edges represent relationships among them
  - Vertices in data provenance model
    - Parameter: represents a specific sensory input from a sensor
    - Measurement: represents a specific value of a parameter. But only measurements at specific state
      changes stored, viz. workflow start/stop/suspend/resume
    - Job: represents a type of action on the underlying streaming data
    - Workflow: represents a collection of jobs in a predefined sequence
    - Workflow Instance: represents an instance of a workflow
    - Task: represents an instance of a job; hence a workflow instance is a collection of tasks
    - User: represents an entity (human or automated) responsible for running the workflow instance
Genoma Data Model (2)

— Vertices in workflow provenance model
  — Source: represents the source of a measurement; this could be either the sensor itself or an entity (e.g., a machine in a factory) where the sensor is installed
  — Location: represents a location. By location we mean a combination of a logical (e.g., edge1) and physical (e.g., GPS coordinates) location. A single location can contain one or more sources.
  — Environment: represents an individual computation environment (computing nodes, etc.) at a location. Provides details of the compute and storage capabilities available at the location.
  — Namespace: refers to the namespace under which the parameter is defined
  — Topic: represents a topic on a broker such as MQTT; would represent a parameter or group of parameters which are being subscribed to via the broker.
  — Storage: represents a resource either on edge or cloud where the data parameter in question is stored, either permanently or temporarily. Belongs to a particular Environment.
### Edges in the *Genoma* Data Model*

*To be read from row to column, e.g., “Task BelongsTo Job”

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Parameter</th>
<th>Measurement</th>
<th>Job</th>
<th>WF</th>
<th>WF Instance</th>
<th>Task</th>
<th>User</th>
<th>Source</th>
<th>Location</th>
<th>Environment</th>
<th>Namespace</th>
<th>Topic</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement</td>
<td>BelongsTo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Job</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WF Instance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Topic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*To be read from row to column, e.g., “Task BelongsTo Job”*
**Genoma Architecture**

— **On Edge**
  — Captures and records provenance data via subscriptions to brokers (e.g., MQTT) on edge devices such as low-end computers or Raspberry PIs.
  — Capture and recording separate, in order to ensure “as a Service”
  — After storage on the edge, provenance data transmitted to the cloud based on policies set by the Policy Modeler

— **On Cloud**
  — Provenance data collected either via Provenance Collector, and/or directly from the data stream processing system
  — Provenance Data storage
    — Cloud: Apache Atlas with modified type system
    — TinkerGraph, whose underlying JanusGraph has same model as in Apache Atlas
  — Separate viewing and processing on Edge as well as Cloud
Genoma Implementation

- On Edge
  - Low-end device such as low storage (within 80 GB) laptop or Raspberry Pi
  - Mini-provenance engine being built due to lack of provenance systems for low-end devices: Provenance Capture, Provenance Recording, Provenance Visualization, Provenance Transmitter
  - Provenance data transmission based on policies set by the Policy Modeler on Edge and Cloud
  - Diagram below shows provenance visualization on Edge using TinkerGraph
**Genoma Implementation (2)**

- On Cloud – following enhancements being built
  - Enhanced Type System for Apache Atlas: enhancement of Apache Atlas’s type system to incorporate the following
    - Additional constructs from our data model
    - Links between these constructs to represent edges
    - Version attributes to represent versioning of nodes and edges in the data model, represent lineage
  - Provenance Collector: receives the provenance data sent by the Provenance Transmitter; and also (based on policies specified via the Provenance Policy Modeler on the cloud) sends requests for provenance data.
  - Diagram depicts a sample lineage graph in Apache Atlas
  - Provenance Policy Modeler under development:
    - On Edge: ensure optimal resource utilization and transmit provenance data to cloud as per user defined policies
    - On Cloud: send requests for provenance data as per user requirements
Key Research Challenges and Future Work

- Real-time Provenance Capture and Recording
  - Should be low overhead so as not to disturb underlying data stream processing system
  - Provenance data should also not be missed
- Provenance Data Storage and Transmission
  - Efficient provenance storage on resource-constrained edge devices
- Efficient Provenance Data Transmission from Edge to Cloud
  - Optimal data transmission policies balancing storage availability on the edge and network overhead of provenance data transmission
  - Data compression techniques to facilitate faster provenance data transmission
  - Data decompression at the cloud, with techniques for correlating recently received provenance data with already stored provenance data
- Securing Provenance Data
  - Lattice-based role based access control model for provenance graphs in *Genoma*
  - Will provide administrators to specify varying levels of access to Genoma nodes and/or edges to a user based on the user’s level in the organizational hierarchy and based on the extent to which they are authorized to view (subsets of) stored provenance data