Framework for Use of ECA in Network Self-Management

draft-bwd-netmod-eca-framework-00

Mohamed Boucadair
Daniel King
Michale Wang
Qin Wu
Chongfeng Xie
IETF 106

Why ECA?

- Event condition action (ECA) provides a structure for active rules in an event driven environment, traditionally consisting of three parts:
  - The **Event** part specifies the signal that triggers the invocation of the rule
  - The **Condition** is a logical test that, if satisfied, causes the action to be carried out
  - The **Action** part consists of updates or invocations on the local data
- IETF SUPA WG: datatracker.ietf.org/wg/supa was created in 2015 to provide approaches to express high-level, possibly network-wide policies to a network management function and classify policy into imperative and declarative policy model.
  - The WG concluded in 2017 as it failed to agree and derive a data model
- Recently (at IETF 105), two drafts both propose ECA-based solutions:
  - draft-bryskin-netconf-automation-framework-00
  - draft-wwx-netmod-event-yang
  - Authors were encouraged to merge discussions
- Its clear ECA will play an important role in event-driven networking
  - The above drafts have common complex use cases and propose models for event, condition and actions

The Motivation for this Work

- Given the suitability of ECA, it seems logical to develop a complimentary document to outline use cases, key issues and an architecture in parallel to the ECA-based solution work
- **Framework for Use of ECA in Network Self-Management**
  - draft-bwd-netmod-eca-framework-00
  - This would form the foundation and mechanism to sanity check the development of ECA-based data models for Network Self-Management
  - It investigates the problem space for network-self management
  - It identifies key issues and challenges that need to be addressed, including:
    - Limited Use Cases
    - Defining Event and Control Logic
    - State Management (see following slides)
      - Centralized and Distributed State Management
      - Delegation of Logic to Devices for Self-Management
    - Execution of Logic
    - Notification Handling (see following slides)
    - Conflicting Policy Resolution (see following slides)
    - Important Security Considerations

State Management

- State applies to
  - Managed object changes, this could be network level or device level
  - The time when Events are triggered
  - the occurrence of an Event

    {event name; start time; end time; threshold value; occurrence times}
- How much sate is this?
  - How long event-based management is prepared?
  - How often event-based management is scheduled?
  - How many start time do we need to support?
  - Do we need to keep state each time when event is triggered?
- State management issues may be mitigated if we:
  - Limit the state that need to be stored
  - Reduce frequency of event-based management being scheduled

Where do we store State?

- It depends
  - Architecture dependent, and who will need to consume the State?
- We have a range of options
  - App could monitor instantaneous network states of managed objects and provide service assurance based on some threshold value
  - App can provide rapid autonomic responses and enable self-management based on historical data of data object
  - Centralized control of system behavior across the whole network based on variables
    - Accumulation/computation thereof over periods of time (e.g. min/max/mean leaf values, history data, threshold value)
- Therefore:
  - State management is needed where time-based policy management is done
  - State management is needed where self-management is done
  - State management is needed where network control logic is delegated
  - State management is needed where network level policy control is done
- The question of state management creates substantial changes, based on
  - What functions do we need to provide?
  - What protocol changes may be required?

Suitable Architectures for State Management?

- Do we need centralized or distributed state management?
  - Is it only dependent only on the service architecture?
  - What about speed, scale, and security of ECA functions?
- Centralized ECA management
  - Central control of network-wide policy behavior:
  - State is stored in controller or the management system, and controlled centrally
  - Requires a searchable repository of all network information
    - Provides diagnostics, service assurance, maintenance and audit capabilities
  - However, responding to network events may take “time”
- Distributed ECA management
  - Delegates policy behavior types to allow autonomic behavior
  - State options are defined in the controller or the management system, but behavior is delegated to the network device
  - Network-wide changes or decision making on App flow information is limited

Conflicting Policy Resolution

- Detecting and Resolving Policy Conflict
  - Conflict between device level ECA policies
  - Conflict between network level ECA policy and device level ECA policy
  - A need for policy conflict detection and policy validation mechanism
- Chain Reaction of Coordinated Events
  - Execute Events in a coordinated manner by the same network devices
  - Execute Events in a coordinated manner by the different network devices
- Do we need to model ECA scripts?
  - Generate script from model
  - Include script in the model
  - Allow global variable shared by multiple script
- What actions can we support?
  - Log
  - Reconfiguration
  - Invoke another event,
- Policy Variables and ECA targets

Securing ECA-based Operations

- Operational and Security considerations discussed in the document, include:
  - Authentication of ECA programming requests
  - Application of suitable authorization methods when enabling ECA functions
  - Securing ECA communication channels
  - Locking ECA device config and state databases
  - Mitigation, and negation, of ECA functional component attacks
  - Logging and auditing of ECA transactions
  - Maintaining ECA device confidentially

Questions for the working group

- Is a framework for ECA-based self-management useful?
  - Describing key use cases, requirements, issues and operational and security considerations
  - Review the centralized state and distributed state, advantages and disadvantages
  - Continue to explore state management and policy conflict resolution
- If not…
  - We could move the key issue discussion on the list only, agree problems, discuss and move on