A Secure Architecture for the Range-Level Command and Control System of a National Cyber Range Testbed

Michael Rosenstein / Frank Corvese

Michael Rosenstein
michael.rosenstein@securedecisions.com
631.759.3910

avi.com
securedecisions.com

What is the National Cyber Range (NCR)?

- DARPA program to develop a secure, self contained cyber testing and experimentation range[1]
- Among its goals:
  - Support rapid and automated reconfiguration of resources
  - Provide broad scalability
  - Allow for simultaneous experiments running at different security levels

[1] **DARPA.** National Cyber Range. *DARPA - STO.* www.darpa.mil/Our_Work/STO/Programs/National_Cyber_Range_(NCR).aspx

michael.rosenstein@securedecisions.com

Range-Level Command & Control System (RangeC2)

- Three primary functions of RangeC2 in the NCR:
  - Management of all range resources
    (resources are bound sets of physical nodes and their interconnectivity)
  - Management of numerous concurrent experiments
  - Enforcement of each experiment's isolation
- Built by Secure Decisions as part of the Johns Hopkins University - Applied Physics Laboratory's (JHU/APL) Phase 2 NCR program
  - Developed between January 2010 and April 2011

michael.rosenstein@securedecisions.com

Operational services provided by the RangeC2

- Inventory management
  - Bringing resources into service or moving them offline
- Experiment administration
  - Scheduling and resource requirements import
- Experiment container management

  [containers are logical groupings of resources that an experiment runs in]
  - Creation, inter-resource topology reconfiguration, assignment swapping, incremental resource recovery, and destruction
- Range monitoring
  - Resource state (available, assigned, or resetting)
  - Container state (building, running, destroying, or failed)

michael.rosenstein@securedecisions.com

Architectural considerations [1/3]

- Design goals
  - Container isolation had to be inherent throughout the design
  - Application should be highly available and survive hard failures without any risk to container isolation or security
  - Application should easily support certification & accreditation
- Management of range state

  [resource interconnectivity, assignment, and power; container and resource state]
  - Basis for all decisions made by the RangeC2
  - Knowledge of range configuration required at all times, even during state transitions or after failures

michael.rosenstein@securedecisions.com

Architectural considerations [2/3]

- Separation into four (4) subcomponents
  - RangeC2 Client
  - Range Controller
  - Transaction System
  - Action Processor

Switch Controller

Level of isolation (and physical security) increases with the criticality of the subcomponent

- Division of responsibility
  - Subcomponents managing operations differ from those handling state changes and ensuring container isolation
  - No single subcomponent could initiate, plan, and execute a state change on its own
  - Most subcomponents are self-contained, preventing errors or security breaches in one from cascading to others

michael.rosenstein@securedecisions.com

Architectural considerations [3/3]

- Overall data flow:
  - Tasks Arrive (from the *RangeC2 Client* or an experiment)
    [Tasks are logical functions of the Range C2]
  - Operationally Processed (*Range Controller*)

  - and if they required changes to the range's state -

  - Verified & Planned (*Transaction System*)
  - Verified & Executed (*Action Processor*)
- The *planning* and *execution* mechanisms follow the ***Action-Transaction*** processing model designed specifically for the needs of this system to:
  - Maintain integrity of range state at all times
  - Easily support code-level analysis and C&A

michael.rosenstein@securedecisions.com

I'm done, now it's your turn

- Questions for you:
  - Has anyone here designed a cyber-range capable of simultaneous experiments at different security levels?
    - How was physical separation achieved, enforced?
- Which would you like to know more about:
  - How the range operated and where the RangeC2 fit into that operational model?
  - Threats the RangeC2 faced?
    - How are they mitigated?
  - The Action-Transaction processing model?
    - How does it maintain range state during hard failures, transition?
    - How does it provide ease of code review and support C&A?

michael.rosenstein@securedecisions.com

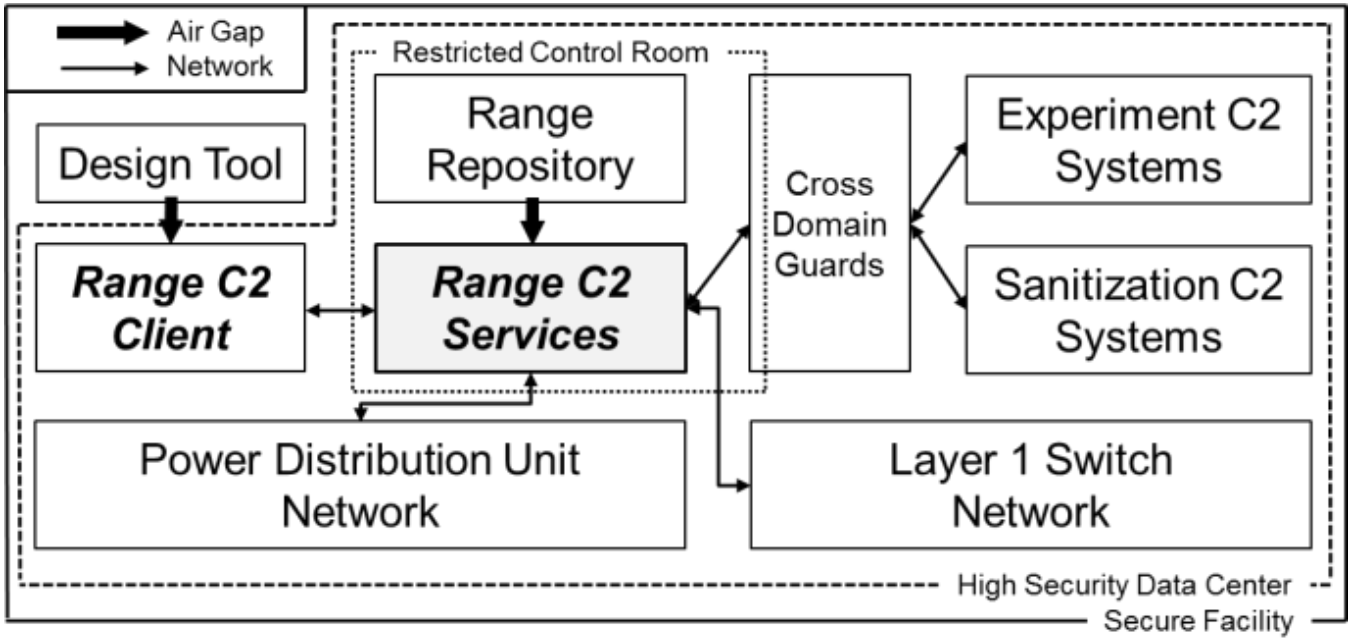
Additional Information

michael.rosenstein@securedecisions.com

RangeC2 Context

Communication and physical security

- The RangeC2 was designed to be located behind several layers of physical security.
- It communicated with other components over an air-gap or private network and entities within a container through cross domain guards (CDG)

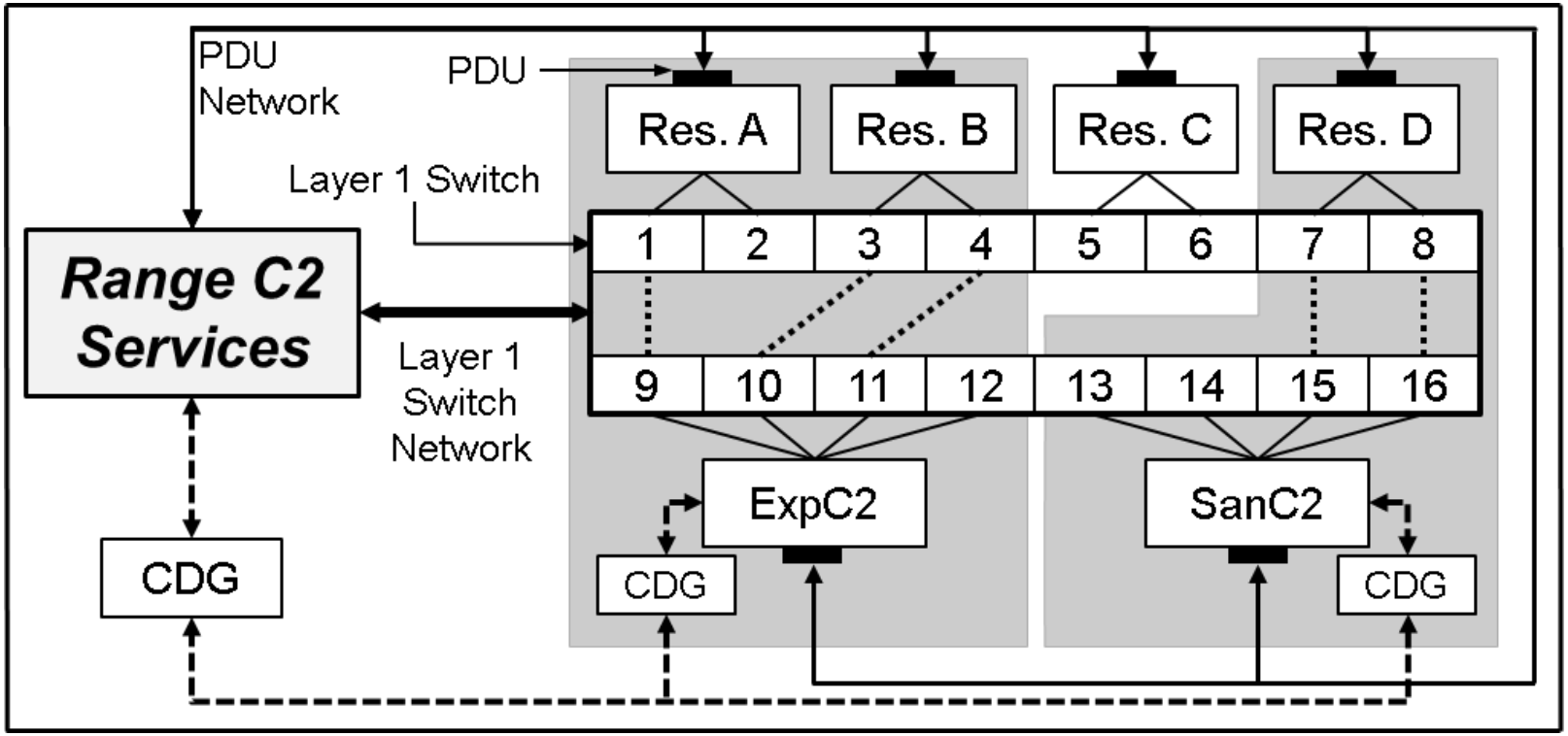


michael.rosenstein@securedecisions.com

RangeC2 Context

Range's state under normal operations

- Experiments operated within containers (gray areas) that contained assigned resources; the RangeC2 used a Layer-1 Switch to configure physical connectivity between them (dotted lines)



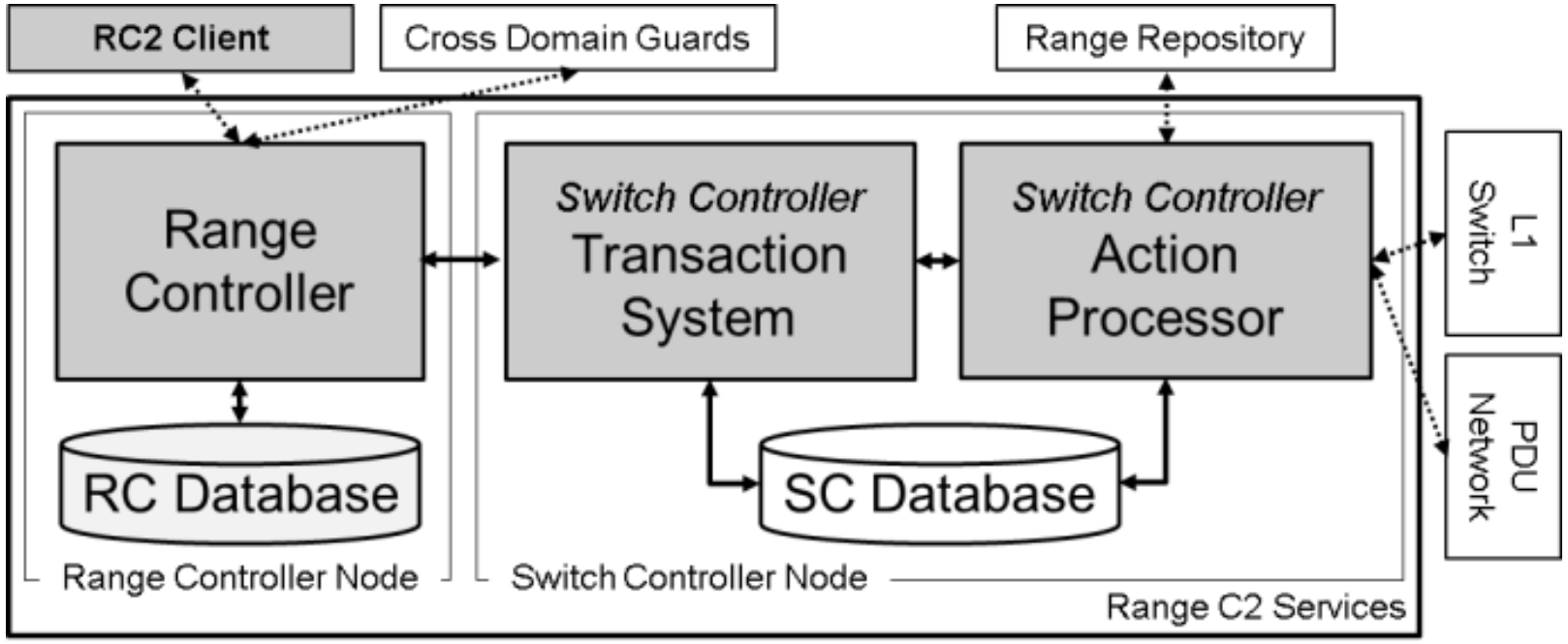
PDU = Power Distribution Unit; ExpC2 = ExperimentC2; SanC2 = SanitizationC2; Res. = Resource

michael.rosenstein@securedecisions.com

RangeC2 Architecture

Four subcomponents

- The Range Controller made decisions affecting how the range behaved at an operational level
- The Switch Controller affected change upon the range's state; its primary concern was container isolation – all else was secondary

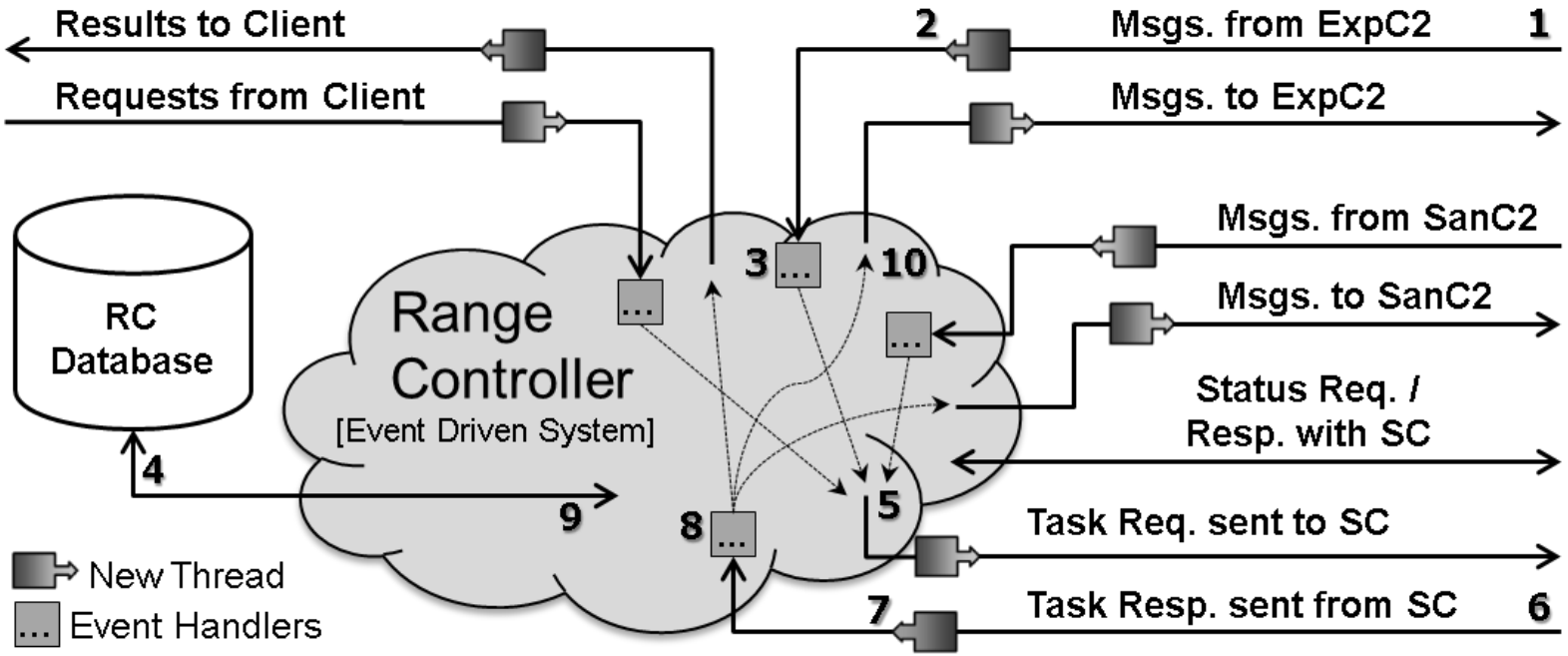


michael.rosenstein@securedecisions.com

RangeC2 Architecture

Range Controller

- The Range Controller was an event driven system that received external stimuli and processed them according to defined rules
- If the stimulus was a request that required a change to the range's state, it was routed to the Switch Controller

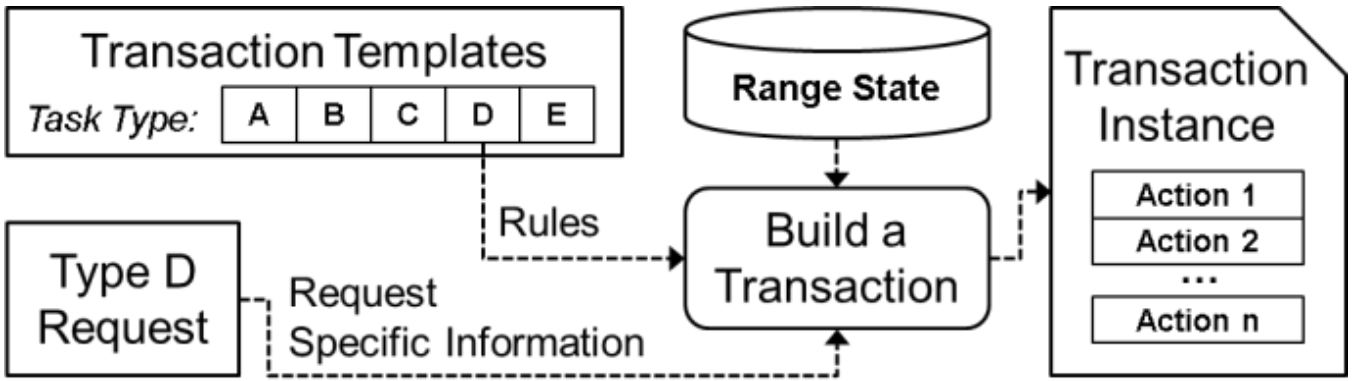


michael.rosenstein@securedecisions.com

RangeC2 Architecture

Switch Controller: Action - Transaction model

- Transactions were hardcoded templates for performing a task
- When a task was requested, information was combined with a template to generate the list of ordered steps, or actions, needed to complete the task
- This list of actions included all necessary security and container isolation checks as well as those needed to perform the task

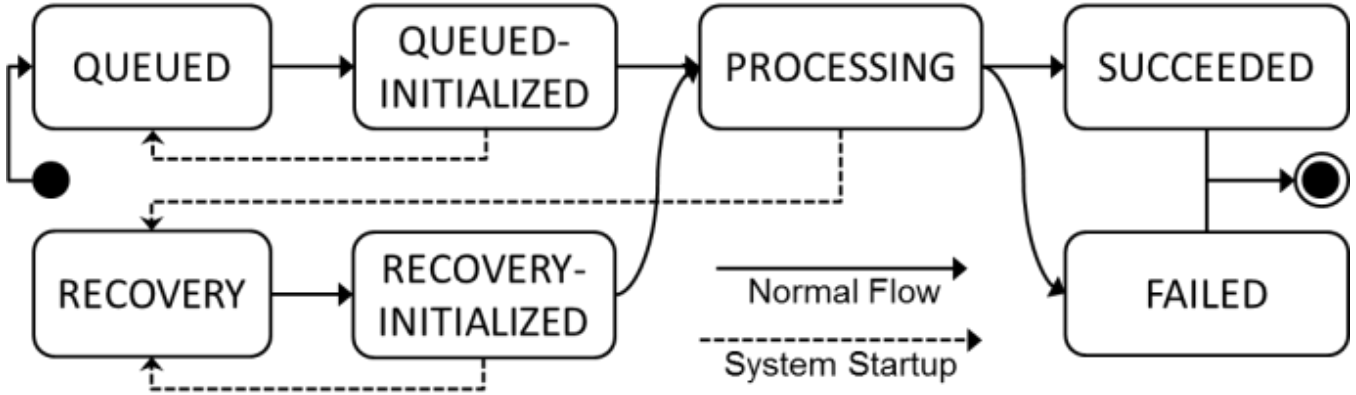


michael.rosenstein@securedecisions.com

RangeC2 Architecture

Switch Controller: Action - Transaction model

- Actions were the atomic units of work the system could perform; each action was completely independent of all others
- To complete a task, all actions in a transaction were processed, serially, in order, according to the state diagram below, and with potential exclusivity to other transactions



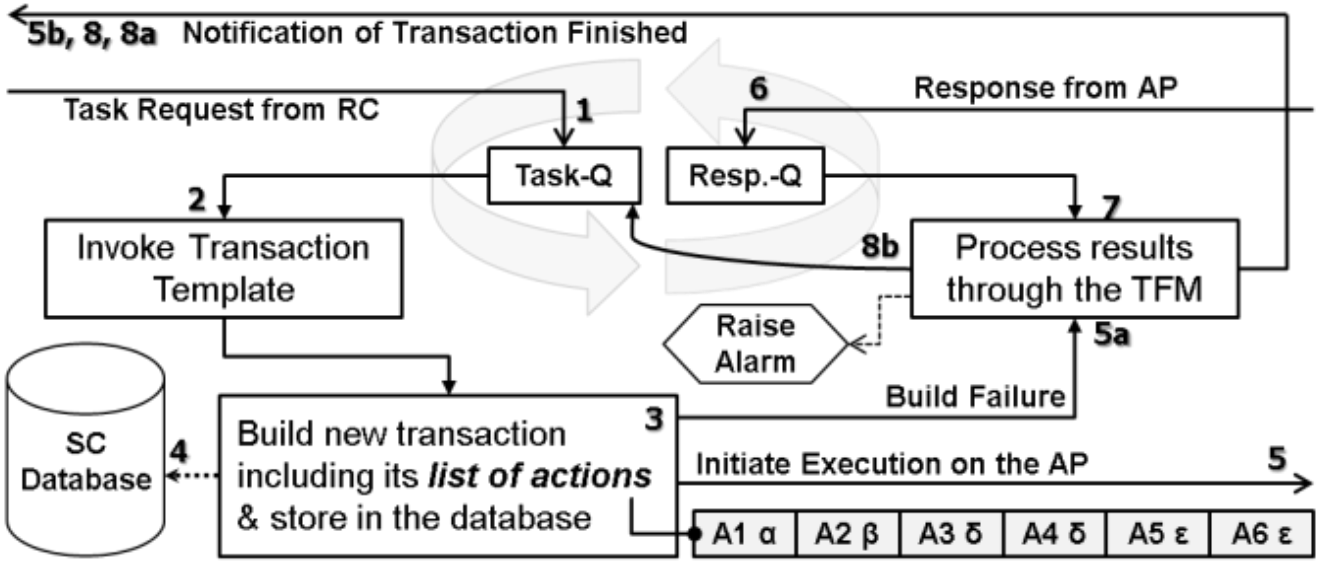
- Advantages: Compartmentalized code - Easy to review
  Inherent progress, logging, and forensics

michael.rosenstein@securedecisions.com

RangeC2 Architecture

Switch Controller – Transaction System

- The Transaction System managed the lifecycle of all transactions, including receiving requests, building transactions, and processing the results
- By design, requests could only be made at the task level; this prevented activities from occurring that might usurp security

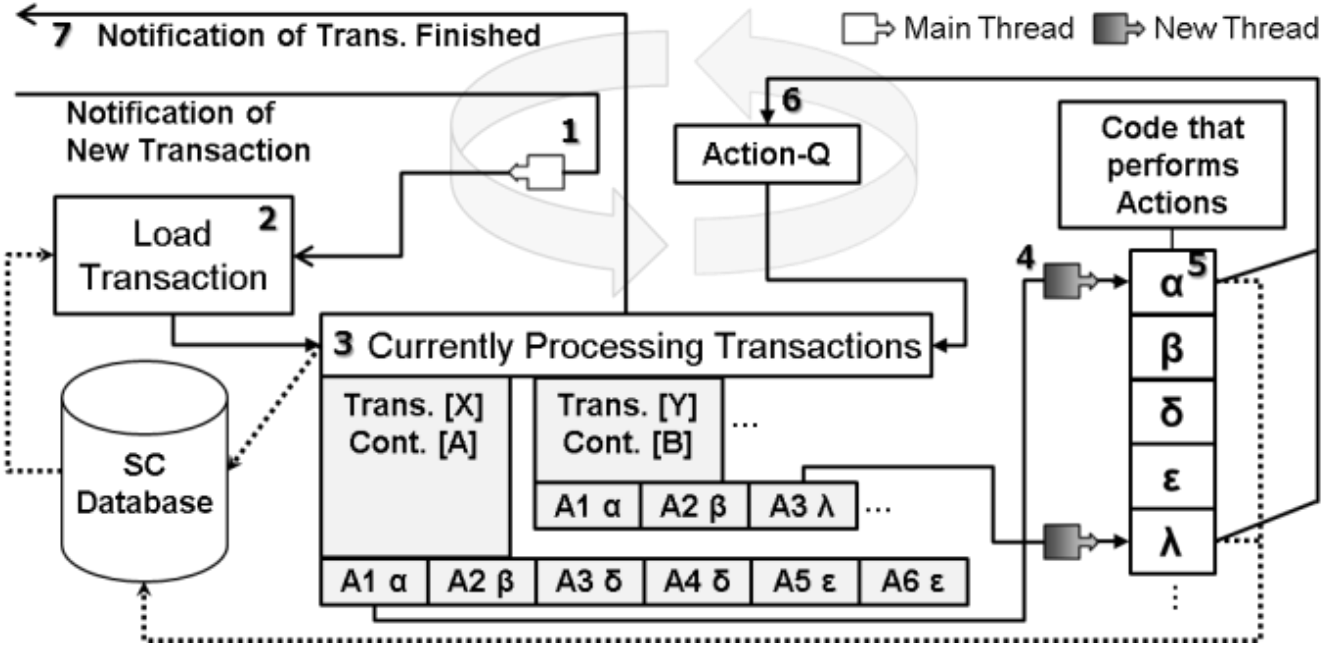


michael.rosenstein@securedecisions.com

RangeC2 Architecture

Switch Controller – Action Processor

- The Action Processor's single purpose was to execute the actions contained within transactions
- This was the only subcomponent capable of altering the range's state (assignment, layer-1 connectivity, power, etc.)



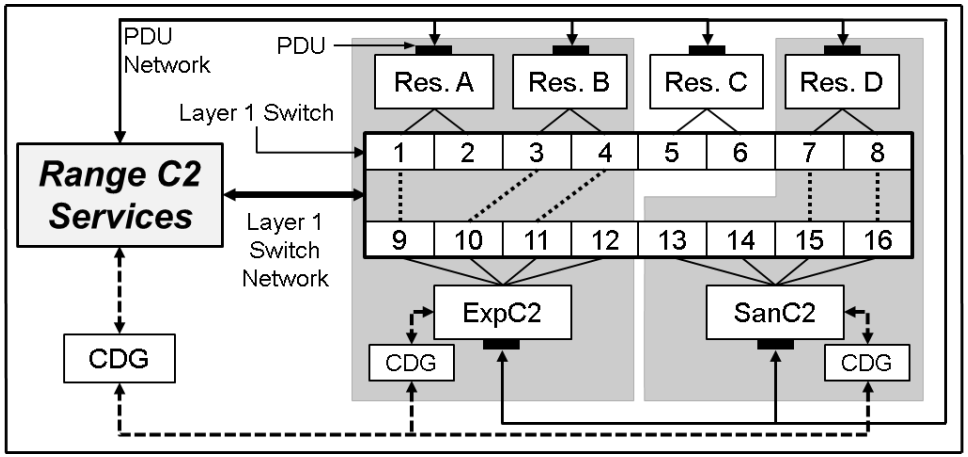
michael.rosenstein@securedecisions.com

Backup slides - Threats

michael.rosenstein@securedecisions.com

Discussion of Threats

- The RangeC2 did not manage experiments and had very little interaction with containers
  - The RangeC2 was not generally susceptible to attacks or malicious activities carried out by the subjects of experimentation



  - The ExperimentC2 was susceptible to these activities, so it was designed to be wholly encapsulated within a container
    - When containers were destroyed, their ExperimentC2s were as well
    - Each new container received a new ExperimentC2

michael.rosenstein@securedecisions.com

Discussion of Threats

- The RangeC2 managed physical resources, including their interconnectivity
  - Insider threats - range staff attempting to alter system functionality
    - Mitigations:
      - Highly restrictive physical and digital access to the Switch Controller
    - *Staff with sufficient access could have caused a data leak!*
      - Cross-container checks would spot this issue, but staff with access to alter these systems could also alter the logs
  - Malicious container activity – containers were assumed to be compromised & might send malicious commands to the RangeC2
    - Mitigations:
      - Cross domain guards only allowed very specific requests to proceed
      - Range Controller acted as a buffer between the Switch Controller and the other range components
      - Switch Controller had a limited API and hardcoded security checks

michael.rosenstein@securedecisions.com

Discussion of Threats

- Supply chain – malicious or error-prone vendors causing hardware to fail in unpredictable ways and with little or no reporting
  - Mitigations:
    - Use of only trusted vendors
    - Working with vendors to develop system unit testing capabilities
    - Extensive testing of new hardware and firmware prior to deployment
- Denial of Service – compromised containers flooding the RangeC2 with valid or invalid requests
  - Mitigations:
    - CDGs, one per container, would filter out nonsensical requests
    - Transaction System would fail to build transactions based on valid requests with invalid parameters (incl. those that would violate security)
  - *Containers requesting large numbers of valid requests could starve the system*
    - Action Processor could prioritize transactions from quiet containers
    - Range Controller could refuse task requests beyond a certain point

michael.rosenstein@securedecisions.com