Outline

- Motivation for accountability
- Early stage work on implementing SDN accountability
- Ideas and feedback
Motivation

- SDNs provide flexibility, but also new opportunities for attacks
- What assurances do we have about previous system events?
- NIST definition of **accountability**: “actions of an [agent] [that can] be traced uniquely to that [agent]” that supports “nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action”
- Why accountability in SDN?
  - Attributing causal actions is difficult; needed for assigning blame fairly and to take appropriate response actions
  - Multiple (potentially distrusting) parties or **agents** with different incentives
Uses of Accountability

- *A posteriori* compliance control
  - Collect **relevant** data about agents’ actions in order to **blame** one or more agents based on agreed-upon *(a priori)* policies

- Forensics
  - Collect **all** data about agents’ actions under adversarial conditions in order to **blame** one or more agents

- Troubleshooting
  - Collect **relevant/all** data about agents’ actions for **testing** or **debugging** purposes under **non-adversarial** (but possibly faulty) conditions
RRE, Accountability, and Cyber Resiliency

RRE Concepts

Monitoring and fusion
- Data for monitoring inputs
- Structured semantic data for logical reasoning
- Responses based on who to blame
- Penalties for policy breaches

Response decisions

Response actions

Recovery

System actions by agents

Data provenance

Policy checking

Blame and attribution

Response actions

Recovery

Accountability Process

maps to

provides

Situational Awareness

Cyber Resiliency

Motivation
Approach

- Applying “accountability regime” design\(^1\) to SDNs based on CS and social science notions of accountability

<table>
<thead>
<tr>
<th>Who is accountable to whom</th>
<th>What one is accountable for</th>
<th>Assurance mechanisms</th>
<th>Standards</th>
<th>Effects of breach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software process level</td>
<td>- Forwarding / topology</td>
<td>- Data provenance</td>
<td>- Legal</td>
<td>- Deterrence</td>
</tr>
<tr>
<td>- Switch-switch</td>
<td>- Intent / policy</td>
<td>- Authenticated logging</td>
<td>- Regulatory</td>
<td>- Loss of money</td>
</tr>
<tr>
<td>- Controller-switch</td>
<td>- Network resources</td>
<td>- Tamper-proof</td>
<td>- Policy</td>
<td>- Loss of reputation</td>
</tr>
<tr>
<td>- Controller-application</td>
<td>- Constraints</td>
<td>- Non-repudiable</td>
<td>- Contractual</td>
<td>- Resiliency</td>
</tr>
<tr>
<td>- Controller–controller</td>
<td>- Criteria</td>
<td>- Fault tolerance</td>
<td></td>
<td>- Response</td>
</tr>
<tr>
<td>User level</td>
<td>- Instructions</td>
<td>- Byzantine fault tolerance</td>
<td></td>
<td>- Recovery</td>
</tr>
<tr>
<td>- Network administrators</td>
<td>- Configuration</td>
<td>- Graphical modeling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Security administrators</td>
<td>- Authorization / access</td>
<td>- Blockchains</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- End users</td>
<td>- Permissions and roles</td>
<td>- Roots of trust</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Organizational level</td>
<td>- Authentication and access</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Clients–providers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Peers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. Who is accountable to whom

- Notion of **agents** and their relationships among each other
- SDN “ecosystem” encompasses many interrelated agents that requires looking at the system holistically
2. What one is accountable for

- Notions of **entities** that store system state and **actions** that can be taken on the entities by the **agents**
- Broader view than just simply a representation of the network as a forwarding graph
3. Assurance mechanisms

- What assurances or guarantees can we make about the data that we collect?
- Important research areas:
  - Data provenance
  - Blockchains and cryptocurrencies
4. Standards

- Two views of accountability standards
  - Accountability by design to support (external) legal systems
  - Accountability by design to create a system for self-executing policy/compliance enforcement
- Automated enforcement of standards via smart contracts
5. Effects of breach

- Go beyond just collecting data for auditing; must use it somehow
- Deterrence and resiliency as complementary aspects
- Completes the RRE “loop”

<table>
<thead>
<tr>
<th>Standards</th>
<th>Effects of breach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legal, Regulatory, Policy, Contractual</td>
<td>Deterrence</td>
</tr>
<tr>
<td></td>
<td>- Loss of money</td>
</tr>
<tr>
<td></td>
<td>- Loss of reputation</td>
</tr>
<tr>
<td></td>
<td>Resiliency</td>
</tr>
<tr>
<td></td>
<td>- Response</td>
</tr>
<tr>
<td></td>
<td>- Recovery</td>
</tr>
</tbody>
</table>
Implementing Accountability

- **Goal**: Design and build a realized accountable SDN system

- **Major components**: 
  - **Data provenance / provenance language** for formally describing system state (i.e., how data came to be) in a structured way
  - **Blockchains** as replicated, fault tolerant distributed consensus ledgers to store commitments about past data provenance
  - **Smart contracts** to implement *a priori* policy agreements among (distrusting) agents for meeting system invariants/predicates and for defining consequences if invariants are breached
Components: Data Provenance

- RDF triples for building distributed provenance graph
- Ontology constrains language
- Extend W3C PROV ontology with SDN semantics
- Use provenance data model to form queries with networking/security semantics
  - E.g., “Was there a path between hosts A and B at this time?”

URL: https://www.w3.org/2011/prov/wiki/ISWCProvTutorial
Components: Blockchains

- Blockchain data stored and executed by all participating nodes for fault-tolerance and replication purposes
- Can store self-executing smart contracts
- Consensus through proof-of-work or BFT-like protocols
- Agents commit hashes of provenance data to blockchain’s smart contracts
- Auditing protocol

Diagram source: “Bitcoin Block Data”, Matthäus Wander, Wikimedia Commons. URL: https://commons.wikimedia.org/wiki/File:Bitcoin_Block_Data.png
Components: Smart Contracts

- Self-executing pieces of code and data that “live” on the blockchain
- In cryptocurrency context, can exchange financial value
- Store policy agreements among agents and relevant commitments related to provenance
- Translate high level network policies to executable code
Proposed Case Studies

- Multiple administrative domains
  - Different administrators
  - Different ownership/trust assumptions of equipment, processes, or data

- Network applications
  - Extensions to SDN controller functionality for providing services (e.g., IDSes, firewalls)
  - Proliferation of network applications makes it challenging to assign blame, especially with apps of equal permission levels
Ideas and Feedback

- Alignment with research goals
- Uses of accountable systems or networks with highly granular provenance metadata and/or automated penalties and responses
- Extension to end host application semantics
- Questions?
Thanks!