

## **SALSA-NetAuth**

### **Architecture for Automating Network Policy**

Kevin Amarin, Editor  
Harvard University

Eric Gauthier, Editor  
Boston University

Created on March 25<sup>th</sup>, 2005  
Last Updated on August 10<sup>th</sup>, 2005

Comments to: [salsa-netauth AT internet2 DOT edu](#)

Copyright © 2005 by Internet2 and/or the respective authors - July 2005

#### ***Overview***

This document, which is a follow-up to SALSA-NetAuth's first document titled "Strategies for Automating Network Policy Enforcement", is intended to detail a policy enforcement architecture for network access. This architecture is intended as a framework to develop standardized mechanisms and detailed descriptions of how to directly implement policy enforcement using existing devices and as a guide for the development of new interoperable solutions. This framework is intended to be flexible, extensible, interoperable with existing infrastructure, and provide the necessary hooks to accommodate upcoming technologies such as federated authentication and authorization schemes.

#### ***General Architecture***

The primary goal of this document is to show how networks can implement network access policies even when network configurations and policies change dynamically. At a very high level, network usage translates into allowing or blocking various sets of network flows. This filtering can be relatively simple, such as allowing all flows, or can be extraordinarily complex, such as the case of inline application proxies which make per-flow decisions based on application layer content. Ultimately, all hosts have some class of policy applied to them. We define a policy class as a set of rules that share the same underlying network access policy. Though a network could have a different policy class for each host, basic networks have only two primary policy classes:

- **Compliant**: Network traffic for the host is restricted according to the overall, general network policy.
- **Non-Compliant**: Network access is restricted to certain remediation tools, registration systems, etc.

When a host comes online, both it and the network cycle through various states as the host's network stack is brought online and ports, vLANs, and flows are established. In terms of policy enforcement, these states can be organized into two categories, either "compliance" states or "policy determination" states. Compliance states are those where the network has settled and is no longer altering itself or modifying the current network access policy. In a compliant state, all network access policies for a particular host have been applied to the various network components. Policy determination states are where the network determines whether or not a host is in compliance with network access policy. The network typically remains in a given policy determination state long enough for it to determine compliance and take certain enforcement actions (e.g. dynamic ACLs), before the network transitions to a new state.

Transitions between network states may be triggered by various events, including external events or policy rules themselves. These triggers can also cause the network to make certain policy look-ups to determine if additional actions are required.

These triggers fall into these general categories:

- **Connections:** A layer2 connection or disconnection attempt occurs. This begins the policy determination process or transitions network access to the offline state.
- **Network Disruption.** This may occur when part of the network changes state. This may require a new DHCP interaction, ACL setup, VPN connection, etc. These events can trigger the network to move into a policy determination state.
- **Host Stack Change:** Host attempts to acquire or renew its DHCP lease after the state of the network has changed. When these events occur, the host and network should transition to the L3INIT/L3NEGOTIATION policy determination states.
- **Scanning:** Active/Passive Scanning detects a state change. These events can trigger a transition into a policy determination state.
- **Agent:** Host-based Agent detects a state change. These events can trigger a transition into a policy determination state.
- **Flows:** A new flow to or from a host is initiated. These events can trigger a transition into a policy determination state.
- **Services:** Application servers detect a state change. This may occur due to account expiration or timeout, receipt of a security notice (e.g. DMCA violation), any manual updates, or an end-user interaction (e.g. registration), etc. This should update the required back-end systems and, potentially, trigger a transition to a state other than Compliant or non-Compliant.
- **Rules:** During the policy determination process, a network policy rule may cause a transition to a new state (see figure 1).

When a trigger event occurs, the network must go through a policy determination process as shown in figure 1. The steps in the policy determination phase are: query a policy repository; given the results of this query, determine what actions are required; initiate the required action. The most basic action is that no action is required. Effectively, the trigger is ignored. The policy determination might also indicate that the current state of the network is no longer valid and cause a transition directly into a new state. This might

occur if an administrative system, such as an intrusion detection system (IDS), determines that a host, though previously in a compliant state, is now violating a particular network policy and needs to be moved into a non-compliant state.

In addition to taking no action and shifting the network to a new state, the network can also take one or more enforcement actions. These actions entail the network modifying either its own configuration or causing events in other administrative systems. For example, an event triggered by a network IDS may require that a hosts administrator be notified. In this case, the IDS detects the event and causes the network to make a policy decision. This results in an email being sent to an administrator followed by another policy decision that no further action is required.

In another case, the IDS event may cause the network to transition back to an earlier state via the policy action of a state transition. Once the network enters into this new state, it makes a policy determination that a dynamic filter is required. This causes an enforcement action of isolating the host's traffic. The policy then defines that no further enforcement action is required transitioning back to the final compliant state, which itself makes a policy decision that no further action is required. Notice that, whenever the network enters into a new state, the network should repeat the process outlined in figure 1.

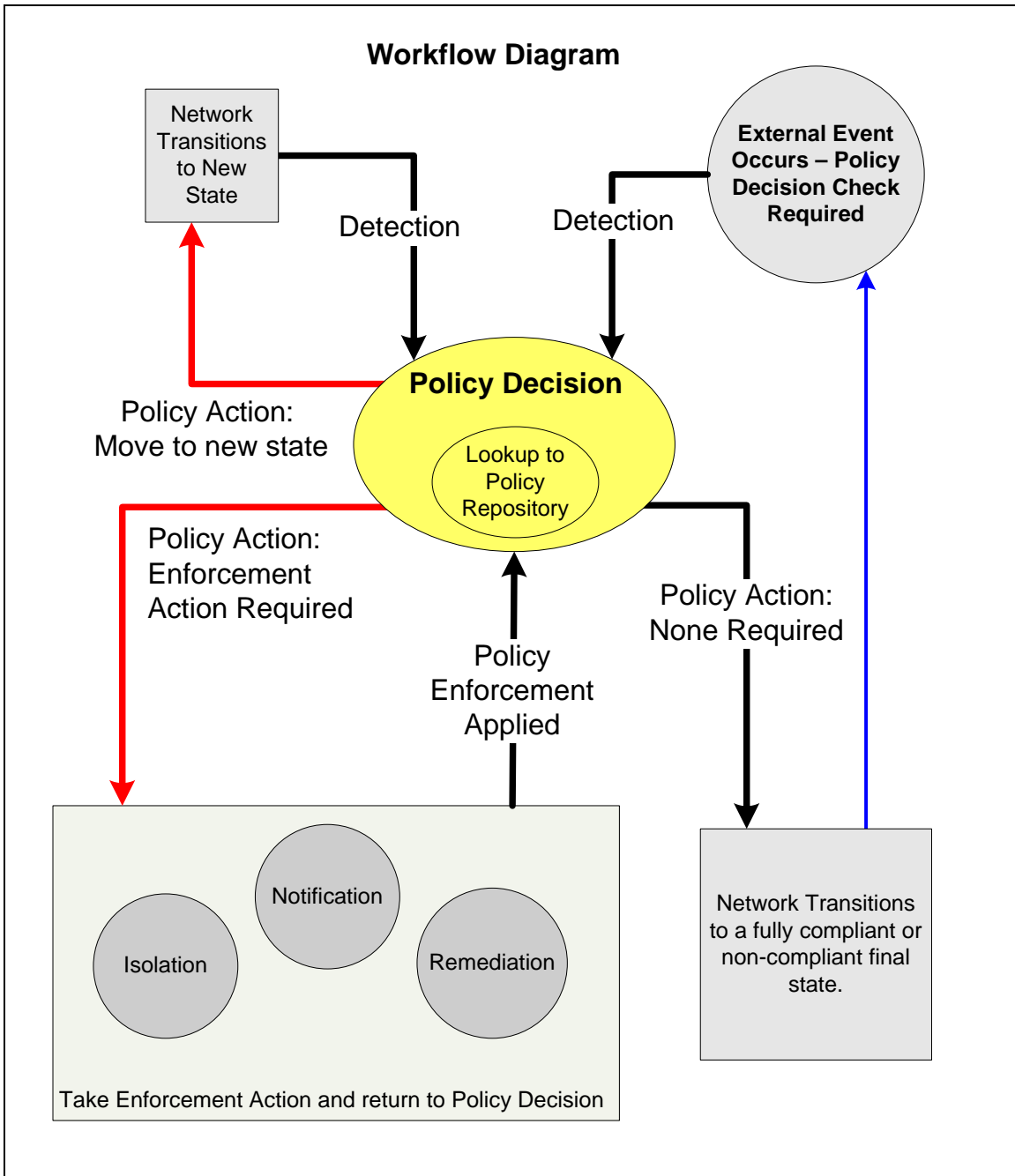


Figure 1 – Policy Determination Process

In order to implement network policy, hosts and networks should move through the states shown in figure 2. Each time that a new state is entered, the network should follow the policy determination process shown in figure 1. The outcome of this process should be that the network transitions through various policy determination states and ends up either with the host offline or as part of a compliance state representing the host's general policy class.

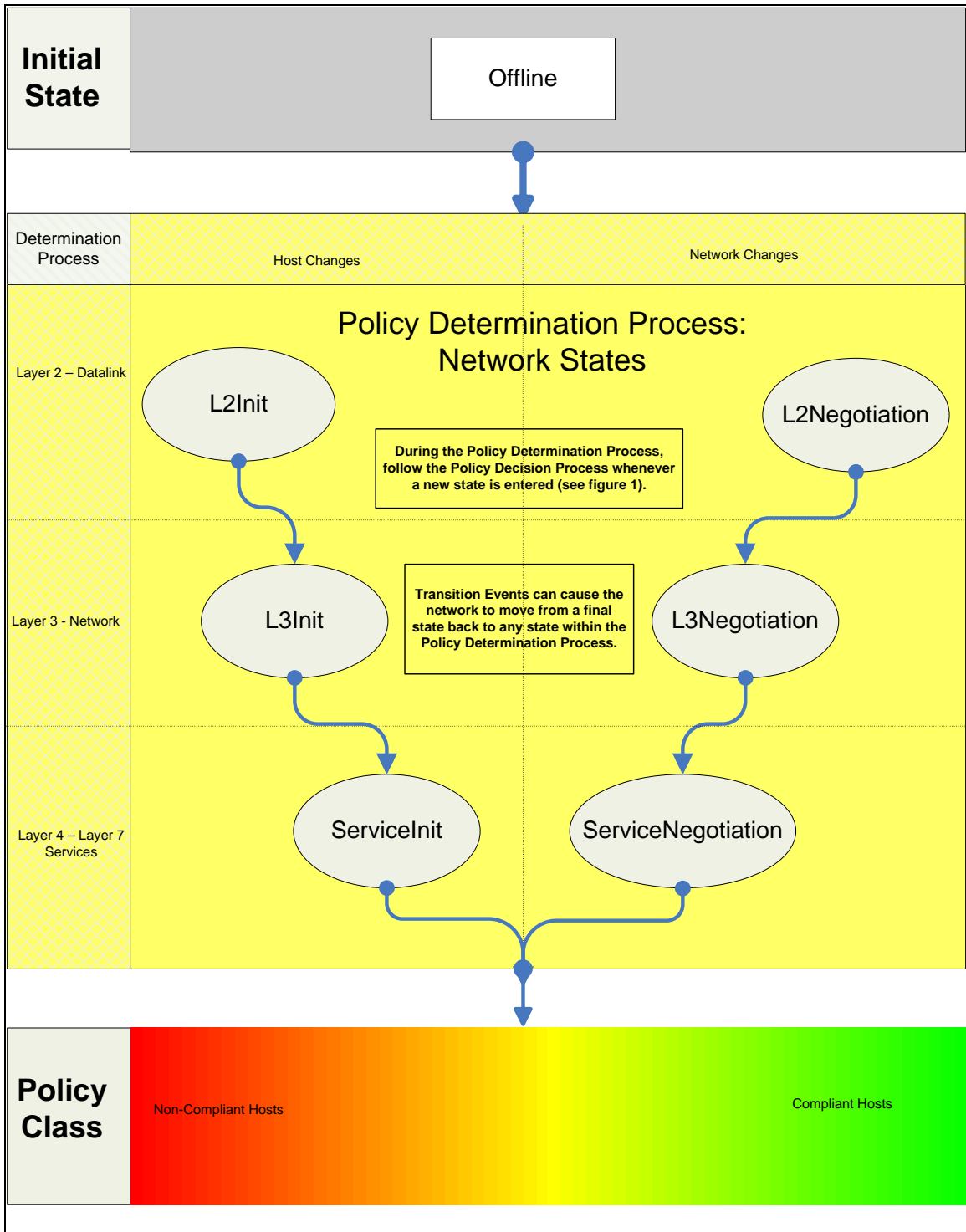


Figure 2 – Virtual Network States

The policy determination states are:

1. **L2INIT**: Host is attempting a layer2 connection. This could be bringing up an ethernet interface or associating to an 802.11 wireless network. In this stage the directly attached network device determines the DHCP IP scope.
2. **L2NEGOTIATION**: The directly attached network brings online the layer2 connection/association to the host and sets the connection into the appropriate VLAN. At this stage, the host has a layer2 link established.
3. **L3INIT**: The host attempts to determine its IP Address. This may be done internally (e.g. static addressing) or by communicating with the network (e.g. DHCP).
4. **L3NEGOTIATION**: The network configures this vlan/connection through any intermediary devices to the default gateway and/or first layer3 hop. Any additional state information that's required to be pushed into the local network occurs at this stage, including pre-populating ACLs, forwarding tables, etc.
5. **SERVICEINIT**: The host is now online with a configured network stack. Passive monitoring of the host's layer 3 traffic can begin here.
6. **SERVICENEGOTIATION**: The network makes any appropriate connections to provide IP services. This may include setting up a tunnel connection (e.g. MPLS or VPN), a set of NAT translations, or a flow state within an in-band device. Dynamic filters or state tables should be updated at this stage, including those devices that dynamically filter on a per-flow basis. The network then moves to Compliant or non-Compliant.

Once the network has completed its policy determination process, the network should be fully configured to enforce network access in conformance with the local institution's network policy.

## ***Acknowledgements***

The hard work and dedicated efforts of the SALSA-Netauth were essential to the foundation, evolution, and continual improvements to this document

## ***Change Log***

1. Version draft 3 to final internet2-salsa-netauth-architecture-200510.
  - A. Formatting updates to add conformance with Internet2 document templates.
2. Version 2a to Version 3
  - A. Updated figure 2 to include references to the host/network model and changed the "final state" portion at the bottom into a policy class box.
  - B. Updated final paragraph in section 3 about 'final states' and replaced it with a discussion of policy classes.

- C. Updated overview text and removed references to ‘final states’.
  - D. Replaces section 1 and 2 with overview paragraphs and removed section 4.
  - E. Various text updates to make terminology consistent
- 3. Version 2 to Version 2a**
- A. Redid figure 1
  - B. Updated paragraph on even triggers to account for new model flow
  - C. Added description of figure 1 enforcement boxes
- 4. Version 1 to Version 2:**
- A. Removed components diagram (formerly figure 2) and made the policy determination states figure into figure 2 (formerly figure 3).
  - B. Total rewrite of section 3.
  - C. Changed name to architecture document