

Local Policy in CUPS

- Revision History, on page 1
- Feature Description, on page 1
- How It Works, on page 2
- Configuring Local Policy in CUPS, on page 2

Revision History

Note Revision history details are not provided for features introduced before release 21.24.

Revision Details	Release
First introduced	Pre 21.24

Feature Description

The local policies are used to control different aspects of a sessions such as - QoS, Data Usage, Subscription profiles, Server Usage, and so on, by means of locally defined policies. It is intended as a replacement or enhancement to PCRF-based policy control. The local policies are triggered during certain events and the associated conditions.

The Local Policy functionality has the following advantages:

- Reusability: Reusable rules engine as a common infrastructure for PCRF-based policies.
- Resource Consumption: Lower memory usage, CPU usage and response time.
- Extensibility: Extensible to handle new events and attributes with minimal effort.
- Execution speed: Shorter reaction time for network events.
- Integration: Seamless integration with the existing policy infrastructure - IMSA and PCEF with a minimal impact on existing services. In case of unreachable events, a mechanism to fallback to PCRF is implemented.

Local policies are useful in various scenarios. For example:

- A Local Policy operates as a fallback mechanism when PCRF is unavailable or when an operator has not deployed PCRF in the infrastructure.
- As an enhancer to PCRF triggers, handling certain triggers locally or to handle triggers unsupported by 3GPP Standards or PCRF.
- Deployments where the subscription policies are static and tiered or has well defined subscriber groups.
- When the response time required is less.

Note The working of the Local Policy feature in the CUPS environment is similar to the non-CUPS P-GW and SAEGW nodes.

When the Gx session operates under local policy, the CP retrieves usage data from the UP and reports it to the PCRF via CCR-U during retry attempts after timer expiration. To fetch the usage report from UP, you must configure the *fetch-usage-from-up* command in the *Local Policy Actiondef configuration* mode. For more information on the command, see the *Local Policy Actiondef Configuration Mode Commands* chapter in *CLI Command Reference* Guide.

How It Works

Local Policy feature is implemented based on the following concepts:

- Event driven rules engine. For example, RAT change event.
- On a registered Event Trigger occurrence, series of registered rules are evaluated based on the Type of Event and the current State.
- On a successful rule match, series of actions are executed.

Configuring Local Policy in CUPS

Note The CLI commands available for non-CUPS Local Policy feature are also applicable in CUPS environment.

Sample Configuration 1:

Following is a sample Local Policy configuration in Control Plane node:

```
configure
  local-policy-service service_name
    ruledef ruledef_name
      condition priority priority radio-access-technology eq eutran
    ruledef ruledef_name
      condition priority priority apn eqcompare_string
```

```
      actiondef actiondef_name
        action priority priority default-qos qci qci_value arp arp_value
      actiondef actiondef_name
       action priority priority activate-rulebase name rulebase_name eventbase
 eventbase_name
        rule priority priority event new-call ruledef ruledef_name actiondef
actiondef_name
        rule priority priority event location-change ruledef ruledef_name
actiondef actiondef_name
        end
```



Note No configuration is required in User Plane node.

Sample Configuration 2:

Following is a sample configuration for configuring **fetch-usage-from-up** in the local policy:

```
config
    failure-handling-template gxfhtemplate
        msg-type credit-control-initial failure-type diameter result-code 3002 to 3002
action continue local-fallback
       msg-type credit-control-update failure-type diameter result-code 3002 to 3002 action
 continue local-fallback
    local-policy-service localpolicygp
      ruledef match_apn
       condition priority 1 apn match *
         actiondef reconnect_to_PCRF
             action priority 1 reconnect-to-server send-usage-report fetch-usage-from-up
                actiondef time
               action priority 1 start-timer timer duration 60 retry-count 10
eventbase default
        rule priority 1 event fallback ruledef match_apn actiondef time continue
        rule priority 2 event timer-expiry ruledef match_apn actiondef reconnect_to_PCRF
```