

Distributed Decisions: Hierarchical Network Policy Control

Table of Contents

Executive Summary	1
Policy Decision-Making Defined	2
Duelling Approaches: Centralized and Distributed Models	2
Centralized Network Policy Control	2
Distributed Network Policy Control	3
Deployment Considerations	5
Use Case-Based Comparison	6
URL Access Control	6
Centralized Architecture	6
Distributed Architecture	7
Observations	8
Traffic Management	9
Centralized Architecture	9
Distributed Architecture	11
Observations	12
Pre-Paid Quota Management	12
Centralized Architecture	12
Distributed Architecture	13
Observations	14
Analysis Conclusions	14
Scalability: Transaction Overload	14
Decision Latency: Microseconds Matter	16
Putting it into Practice: Sandvine's Distributed Network Policy Control	17

Executive Summary

This whitepaper demonstrates the advantages of distributed, coordinated network policy control architecture over traditional centralized models. Specifically, the analysis demonstrates that a distributed model is superior based on two primary factors, which both contribute to lower deployment and operating costs:

- Greatly reduced policy-signaling overhead
- Optimized network performance

A distributed network policy control architecture is characterized by a hierarchy of elements (minimally split between the data plane, control plane and provisioning nodes) that arbitrate, decide and communicate policy actions up and down the chain as required.

First, atomically self-contained policy rule packages that are on the data plane nodes greatly reduce (by orders of magnitude) the policy signaling rate between the data plane and control plane (this can be thought of as a reduction in signaling between the PCEF and PCRF).

Second, by making decisions close to the data plane and source information, the decision latency is reduced to the point where it is on the same order as typical flow life-times, leading to much more effective application of network policy control techniques. Ultimately, better policy control means better network performance.

This paper illustrates the benefits of distributed network policy control architecture by exploring the policy decision-making process, latency and signaling overhead between a distributed model and a centralized model through common policy use-cases.

The analysis leads to the conclusion that network policy decisions should be determined by the lowest element in the network which has access to all of the required information, consistent with a distributed architecture.

Policy Decision-Making Defined

First, let us define what is meant by a 'policy decision'. 3GPP¹ defines the relevant terms as follows:

- **Policy Decisions** - "Policy Control and Charging (PCC) rules and IP-Connectivity Access Network (IP-CAN) bearer attributes, which is provided by the Policy and Charging Rules Function (PCRF) to the Policy and Charging Enforcement Function (PCEF) for policy and charging control."
- **PCC Rules** - "A set of information enabling the detection of a service data flow and providing parameters for policy control and/or charging control."
- **Policy Control** - "The process whereby the PCRF indicates to the PCEF how to control the IP-CAN bearer. Policy control includes QoS control and/or gating control²."
- **Charging Control** - "The process of associating packets, belonging to a service data flow, to a charging key and applying online charging and/or offline charging, as appropriate." This definition can and frequently is extended to support a growing library of use cases - sometimes being standardized by 3GPP.³

In other words, policy decision equates to classifying subscriber service flows, managing (deciding and acting upon) the service flow for QoS or other attributes (policy control)⁴ and charging.

3GPP currently only defines a 5-tuple set of classifiers for a service flow of source IP address, destination IP address, source port number, destination port number, and protocol ID of the protocol above IP. However it can be extended to other classifications such as application, location (for mobility) and network congestion.

In addition, the management actions can also be extended beyond the QoS to diverting to a caching node, a subscriber portal or any other actions desired by a service provider.

Finally, 3GPP is extending the data plane management nodes initially from only a PCEF to also having a Traffic Detection Function (TDF) (being defined in R11) which would not only identify the traffic application but also detect details such as the video or audio encoding being used on the data path.

Duelling Approaches: Centralized and Distributed Models

Centralized Network Policy Control

Centralized policy decision-making is defined as policy control and charging being managed by a centralized policy management node (frequently referred to as a policy decision point, or PDP) which, in mobile standards, instructs PCEF (and TDF) nodes on required policy actions once a decision has been made based on the data plane conditions reported by the PCEF/TDF nodes. In Figure 1, which shows a generalized centralized architecture, policy enforcement is made at the policy enforcement point (PEP).

¹ 3GPP TS 23.203 V9.6.0, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Policy and charging control architecture (Release 9).

² This may include content filtering and diverting

³ Examples include the introduction of the Traffic Detection Function (TDF) in R11 and vendor-specific extensions.

⁴ This may include measurements for reporting on the QoS and other attributes.

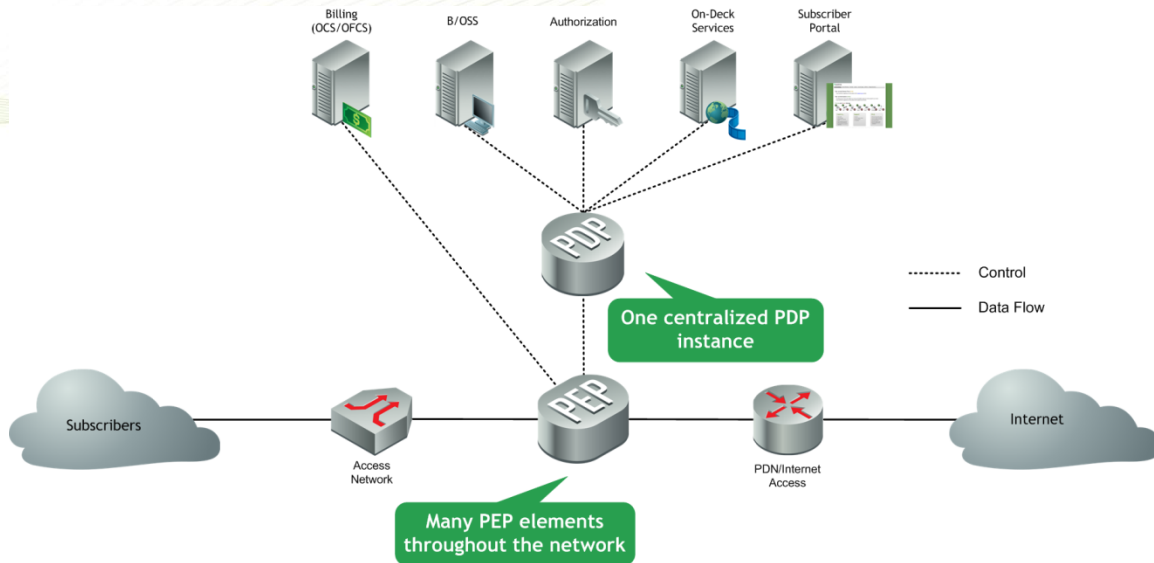


Figure 1: Generalized Centralized Architecture

The current architecture defined by 3GPP standards is a centralized architecture. The 3GPP standards also assume an IP Multimedia System-based service model, which differs markedly from the over-the-top (OTT) model (including services like Facebook and YouTube) that characterizes actual network traffic.

Distributed Network Policy Control

Distributed policy decision-making is defined as allowing nodes within the data plane (PCEF/TDF nodes in the 3GPP model) to make policy decisions locally when sufficient information is available, and otherwise escalating the policy decision responsibility. When escalation is required, the data plane nodes pass information describing the data plane conditions to policy decision capabilities in the control plane. Ultimately, in the rare circumstances that it is necessary to do so, the control plane nodes can pass information up to the provisioning layer (typically Billing and Operations Support Systems). A generalized description of these functional layers is shown below in Figure 2.

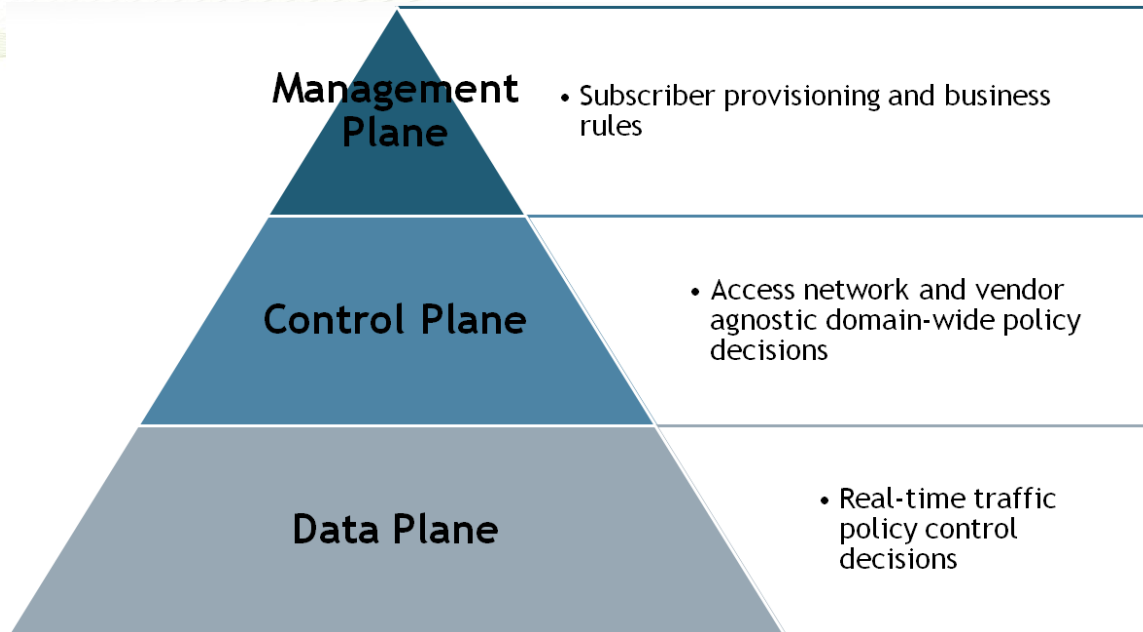


Figure 2: Generalized functional layers within a distributed network policy control architecture

The control plane addresses items like obtaining subscriber profiles and active session information, interaction with external systems, and storing and manipulating state information spanning multiple data plane elements or access technologies.

The data plane performs functions like application identification, traffic management through bandwidth-limiting and other techniques, real-time usage and event counting, and subscriber interaction (through redirection to a captive portal).

In practice, the nodes deployed in the data plane will include both data measurement and control decision-making capabilities.

In general, and this is true regardless of whether or not the architecture is distributed or centralized, crossing the boundary between the data plane and the control plane is extremely expensive in terms of performance and latency.

In complex network environments characterized by multiple access technologies, vendors, and standards, a distributed network policy control architecture will also include a policy control node that performs network-wide arbitration (including applying uniformity of decisions to enforcement nodes) when decisions require inputs from a span of sources such as subscriber-specific conditions, business intelligence analysis and multiple disparate network attributes.

A generalized distributed model is presented in Figure 3. In this model, distributed elements containing both PDP and PEP functionality move decision-making into the network's data plane.

Additionally, a node that performs central policy arbitration is now deployed in the control plane, and coordinates information and policy uniformly regardless of network technology, interface standards, or vendor differentiation. For instance, this arbitrator can coordinate policy within a converged network (say, Cable and HSPA), in which enforcement is provided using PCMM (for the Cable components) and

PCEF (for High Speed Packet Access). The arbitrator also serves as a hub for a wide collection of interfaces, enabling rapid distribution of contextual information and coordination of different network elements.

A key differentiating feature of the distributed network policy control system is that the vast majority of decisions are made in the combined PDP/PEP elements; only in the (relatively) rare cases that complete information is not available do these elements escalate responsibility to the central arbitrator.

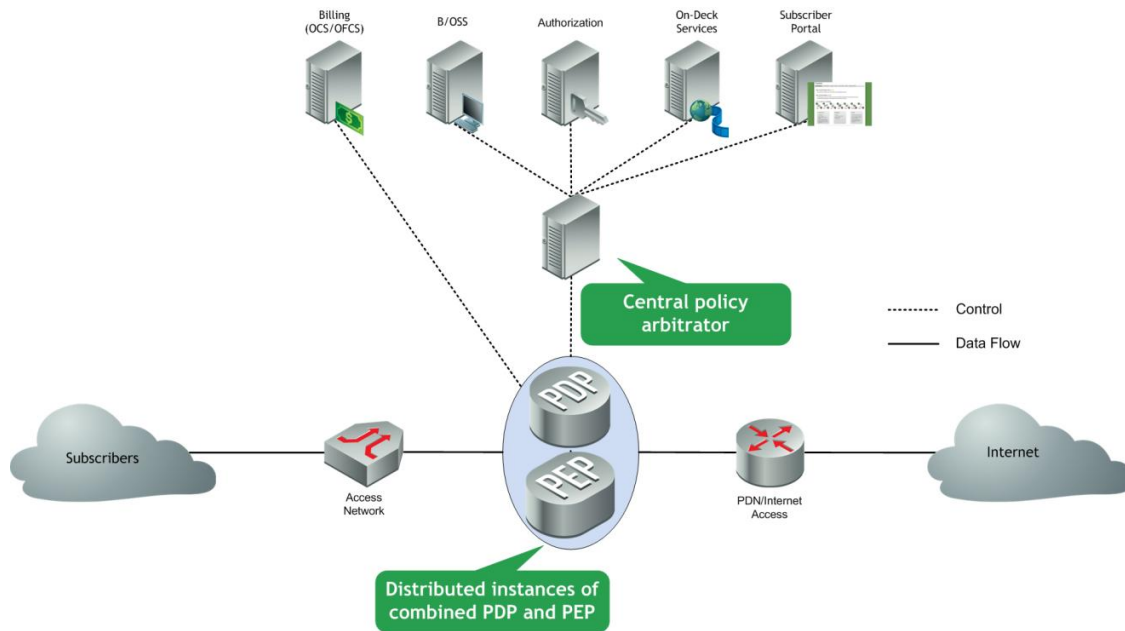


Figure 3: Generalized Distributed Architecture

Deployment Considerations

Operator networks are in a constant state of evolution, with technology advances, shifting subscriber usage habits and billing preferences, new and updated standards, and innovative policy control mechanisms contributing to an operational environment featuring a diverse mix of architectures, technologies, vendors and standards.

Efficiency demands that whatever policy decision model is deployed, it must be able to operate within the operational realities facing service providers. It is simply pragmatic to consider the actual state of affairs when examining alternatives, rather than basing examination upon an idealized environment in which services politely request network resources and behave altruistically for the greater good.

Actual networks feature a countless number of over-the-top services competing with each other and with on-deck offerings for limited network resources. Some applications generate a small number of long-lived flows, while others create hundreds of short-lived flows. Different applications and protocols rely on different optimization mechanisms, and rarely (if ever) consider wider network conditions as factors in these optimization decisions.

Essentially, the Internet today is an every-application-for-itself Wild West, and it is dangerous to assume any cooperation on the part of network traffic towards the goal of effective policy control.

Use Case-Based Comparison

Let us now examine the performance of distributed and centralized network policy control architectures within a set of widely-deployed PCC use cases. We will analyze the information required for the use case policy decisions, where the information for the decision resides, and the decision path flows and PCC actions. Key criteria for evaluating the results of this examination will be decision latency and signaling overhead. The four use cases to be considered are:

1. **URL Access Control (UAC)** - In this implementation of content filtering, which is consistent with most regulatory filtering requirements, certain URLs are blacklisted and access to these URLs is to be prevented. Instead of landing on the blacklisted URL, subscribers shall land on a notification portal. A report of the request is made to an audit system.
2. **Traffic Management (TM)** - In this version, there is no absolute limit on usage; however, recent heavy users are subject to bandwidth-limiting when there is congestion on the access resource.
3. **Pre-paid Quota Management** - Exceeding a defined usage quota subjects users to bandwidth-limiting. Quota thresholds originate in an Online Charging System (OCS), and usage is recorded in OCS and PCC for audit purposes.

Note that we will only assume the most basic versions of these use cases. It is easily possible to extend these examples to include interactions with subscriber self-management portals (for tops-ups, usage status, etc), subscriber notification (for instance, triggered by a subscriber nearing or exceeding usage limits) or other enhancements.

It is helpful to compare the flow diagrams with Figure 1 and Figure 3 to visualize the data and control flow in centralized and distributed architectures.

URL Access Control

Table 1 shows the parameters for the URL Access Control use case.

Information Required	List of restricted URLs, URL requested by subscriber, redirection landing page
Information Location	Data Plane: URL Requested by subscriber Control Plane and B/OSS: List of restricted URLs, redirection landing page
Decisions Needed	Is a requested URL on the restricted list?
Actions	Allow connection to continue or redirect to landing page Report request of restricted URL

Table 1: Use Case Parameters (URL Access Control/Content Filtering)

Centralized Architecture

Figure 4 presents the message flows for the URL Access Control use case in a centralized decision-making environment. For the sake of this analysis, it is assumed that the PDP receives the list of restricted URLs from a central authority. In this scenario, whenever the PEP sees a URL request, it must request instructions from the PDP. Upon receiving information about the request being made, the PDP compares the URL against the blacklist, and then responds with a “Block” or “Do Not Block” decision.

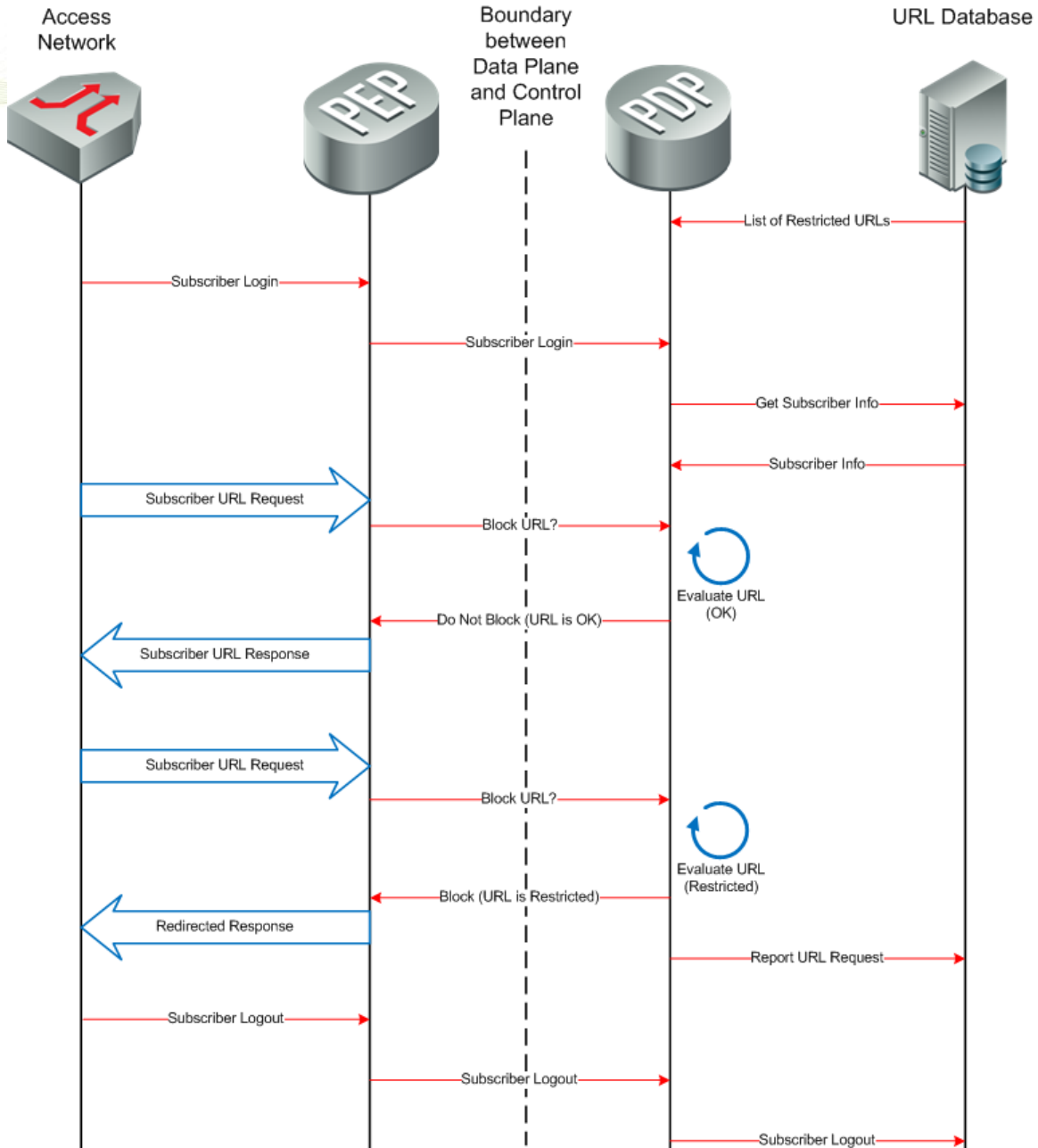


Figure 4: Message Flows with Centralized Architecture (URL Access Control)

Distributed Architecture

Figure 5 presents the message flows for the URL Access Control use case in a distributed decision-making environment. Once again, it is assumed that the list of restricted URLs has been communicated to the PDP function (in this case through the Policy Arbitrator).

Contrary to the centralized environment, the joint PEP/PDP elements have sufficient information to make the “Block”/“Do Not Block” decision locally, outright eliminating a huge number of control messages.

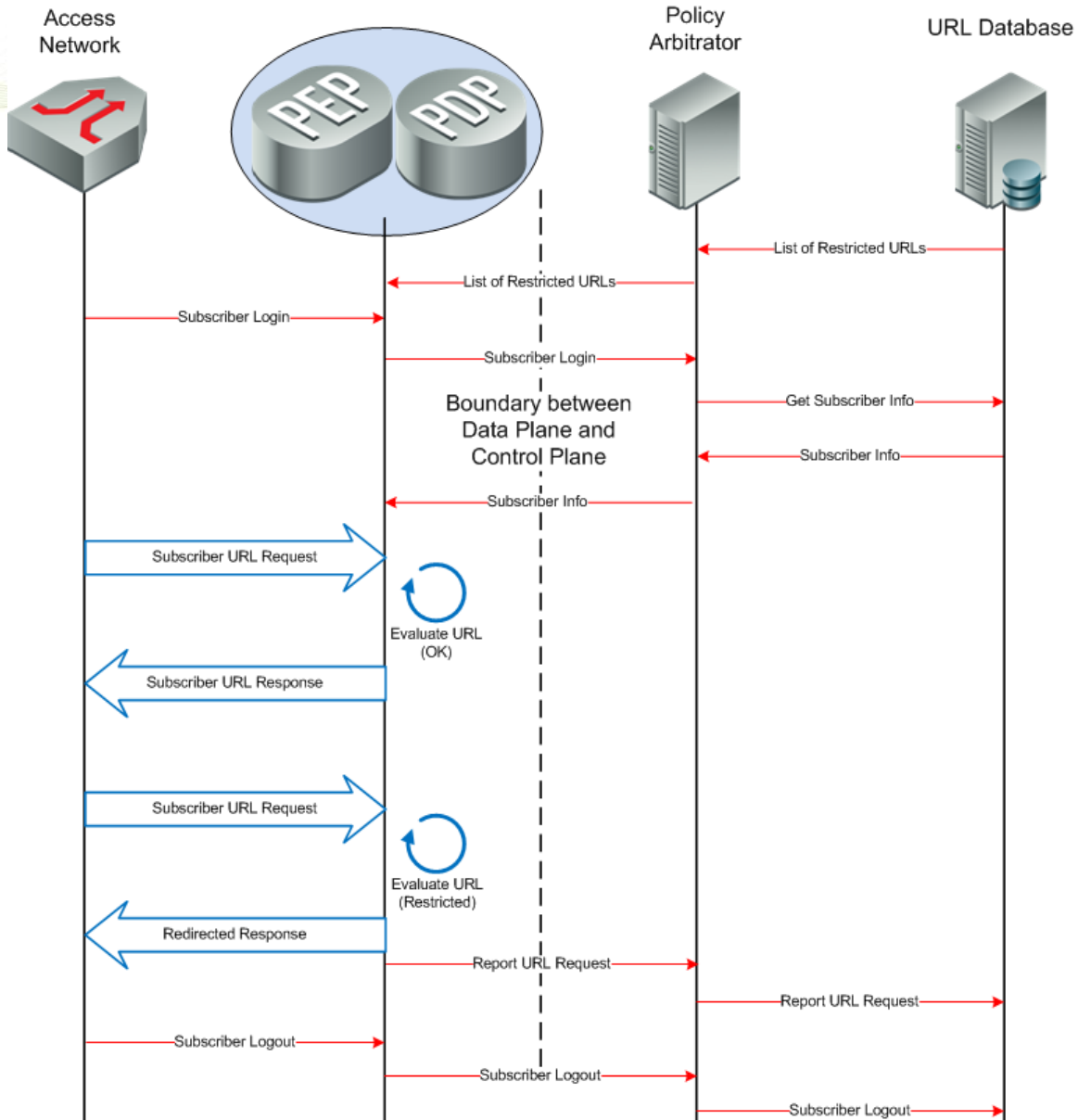


Figure 5: Message Flows with distributed Architecture (URL Access Control)

Observations

While the message flows presented above represent simplified scenarios, they nevertheless lead to some useful observations. Primarily the distributed approach has two obvious benefits over the centralized design:

1. **Signaling Overhead:** within the distributed architecture, control traffic is cut down by several orders of magnitude since the “Block”/“Do Not Block” decision is made locally, without needing to generate two control messages.
2. **Decision Latency:** within the distributed architecture, the decision latency is dramatically reduced since it is made locally and is therefore not subject to network latency.

Reduced signaling overhead has direct benefits in terms of deployment and operations cost of the control plane architecture; by limiting the number of transactions per second, the overall deployment can be reduced by orders of magnitude.

Consider that a study by Google showed that a simple webpage download of 320 KB resulted in an average of 45 GET requests to around 7 unique hosts per webpage download⁵ due to the various sources that comprise a single webpage. If each of those GETs corresponded to a unique URL, then there would be almost 18,000 GET requests (and 36,000 control flows) per Gbps of HTTP traffic.

Also, by making the “Block” or “Do Not Block” decision locally, network management policies become more effective. For instance, in the centralized model, there are two ways in which blocking and redirection can be accomplished:

1. Race Condition: allow the GET request to proceed and rely on the PDP response to arrive at the subscriber before the actual response
2. Wait-and-See: delay the GET request until the PDP response is received

Neither approach is desirable: in the first case, it is possible that subscribers will gain access to restricted websites; in the second, the subscriber quality of experience is impacted by enormous latency (recall that a single web page generates on average 45 requests, each of which is subjected to the wait).

Traffic Management

Table 2 shows the parameters for the Traffic Management use case.

Information Required	Recent subscriber usage information, local resource usage, defined congestion limit, bandwidth limit, applications to be limited
Information Location	Data Plane: subscriber usage, resource usage Control Plane and B/OSS: defined congestion limit, bandwidth limit, applications to be limited
Decisions Needed	Identification of heaviest recent users, determination of congestion state
Actions	Limit bandwidth of particular applications for heaviest recent users

Table 2: Use Case Parameters (Traffic Management)

Centralized Architecture

Within a centralized architecture, there are a number of control messages to measure resource utilization (information needed to determine whether or not a resource is congested). These can be seen in Figure 6. Additionally, subscriber usage must be reported so that the PDP can determine which subscribers are recent heavy users.

The PDP must constantly evaluate the state of congestion of every network access resource, and must maintain a list of all recent heavy users per resource so that the PEP is aware of which users should be

⁵ <http://code.google.com/speed/articles/web-metrics.html>

subjected to bandwidth limiting.

The number of control flows required to provide the PDP with all of the measurements and usage reports that are required for decision-making is immense.

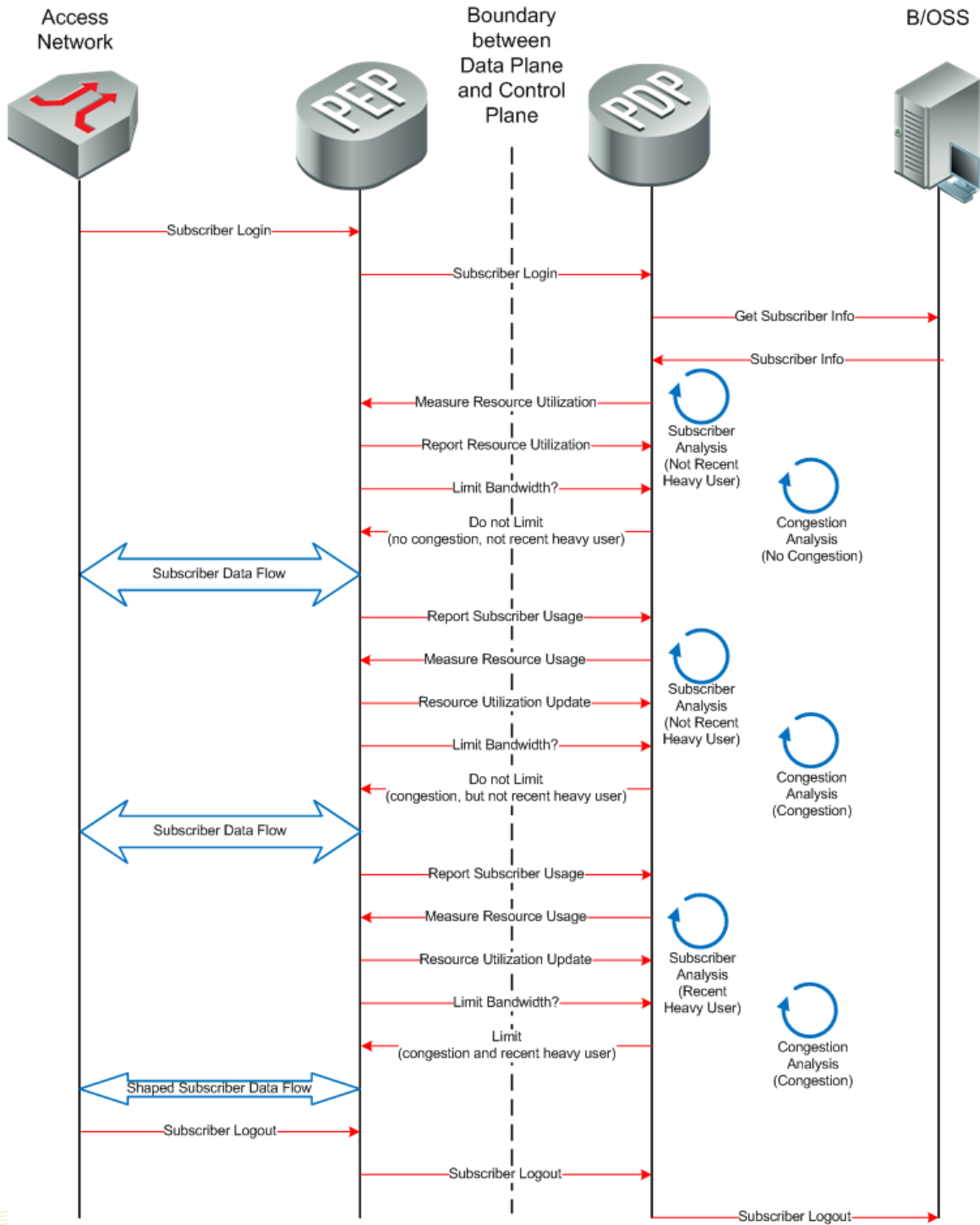


Figure 6: Message Flows with Centralized Architecture (Traffic Management)

Distributed Architecture

In a distributed architecture, the congestion evaluation and determination of recent heavy users is made locally, in the data plane. Contrasting Figure 7 with Figure 6 gives an indication as to the profound positive impact this architecture has on the number of control flows.

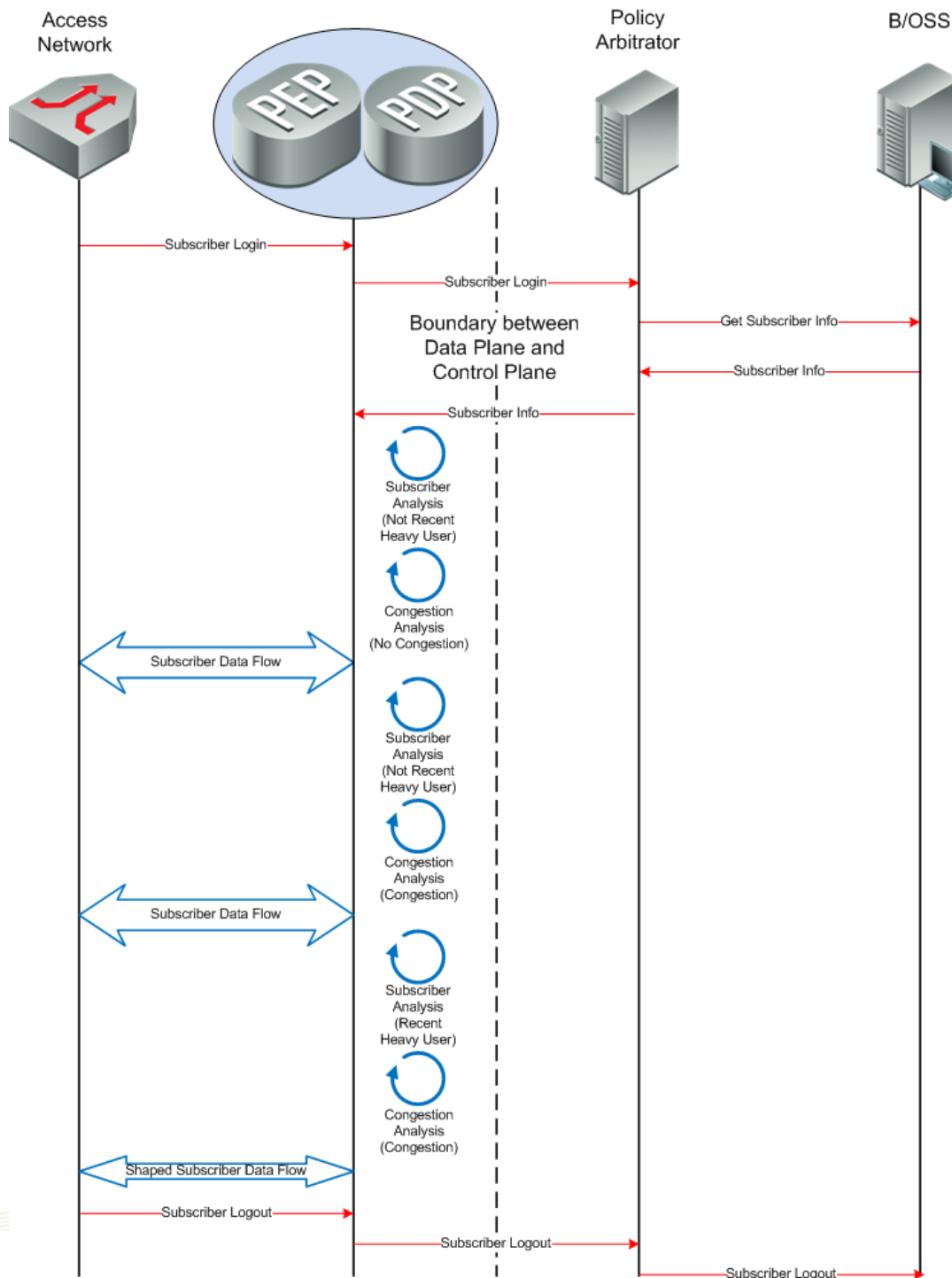


Figure 7: Message Flows with Distributed Architecture (Traffic Management)

Observations

The observations are consistent with those from the URL Access Control use-case analysis; that is, the distributed architecture offers significantly reduced control overhead and dramatically shorter decision latency, both without compromising the overall policy enforcement effectiveness.

Consider, as well, that this use case is simplified. In many real-world deployments, bandwidth limiting might be limited to certain classes of application, which could mean even more control flows in a centralized environment.

Pre-Paid Quota Management

Table 3 shows the parameters for the Fair Use Policy use case.

Information Required	Usage quota (time or bytes), subscriber usage (time or bytes), bandwidth limit
Information Location	Data Plane: subscriber usage Control Plane and B/OSS: usage quota, bandwidth limit
Decisions Needed	Determine when OCS-granted usage quota is exceeded
Actions	Report subscriber usage to OCS Limit bandwidth for subscribers who have exceeded prepaid usage quota

Table 3 - Use Case Parameters (Pre-Paid Quota Management)

Centralized Architecture

Figure 8 shows the data and control flows for the Pre-Paid Quota Management use case in a centralized decision-making environment. When a subscriber connects to the network, the login triggers a service request to the OCS, which grants the request when quota remains.

Subsequent subscriber data flows cause usage calculations, followed by service request updates, and grants for service when quota remains. When quota is exhausted, the OCS instructs the PEP to enforce traffic limiting.

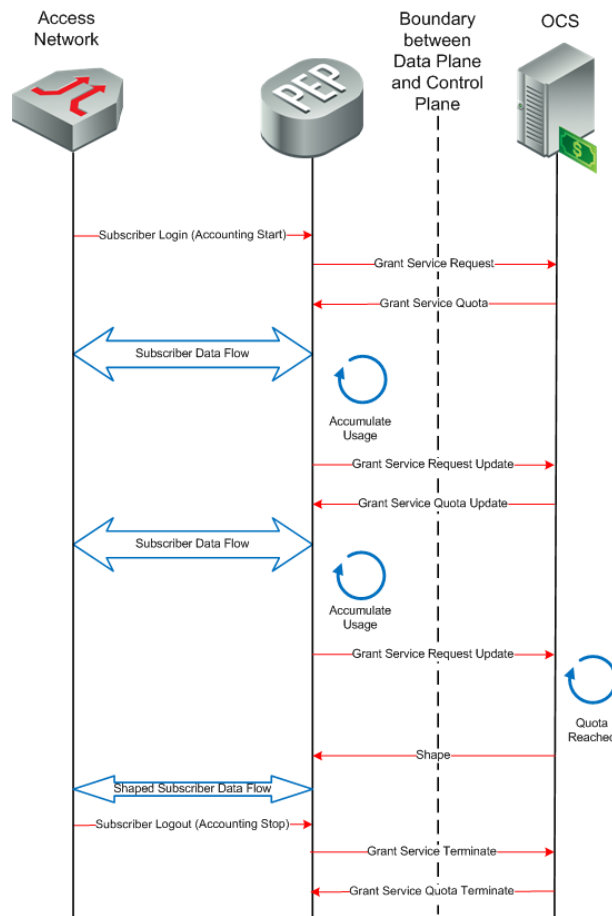


Figure 8: Message Flows with Centralized Architecture (Pre-Paid Quota Management)

In the case presented in Figure 8, the traffic limiting remains in place until the subscriber logs off from the network. In reality, the limiting would remain until the billing period rolls over (thereby resetting the quota), or the subscriber purchases additional quota credit.

Distributed Architecture

Figure 9 shows the data and control flows for the Pre-Paid Quota Management use case in a distributed decision-making environment. When a subscriber connects to the network, the login triggers a service request to the OCS, which grants the request when quota remains and communicates the remaining quota.

Instead of causing additional control traffic for every subsequent subscriber data flow (or even batch of flows), subscriber usage is calculated locally on the combined PEP/PDP elements and remaining quota is evaluated in real-time based on the contextual information initially provided by the OCS at the time of log-in. When quota is exhausted, the bandwidth-limiting is triggered, and the quota exhaustion notification is sent up the hierarchy to the OCS (and optionally to the central arbitrator, if such information is of use to other control plane elements).

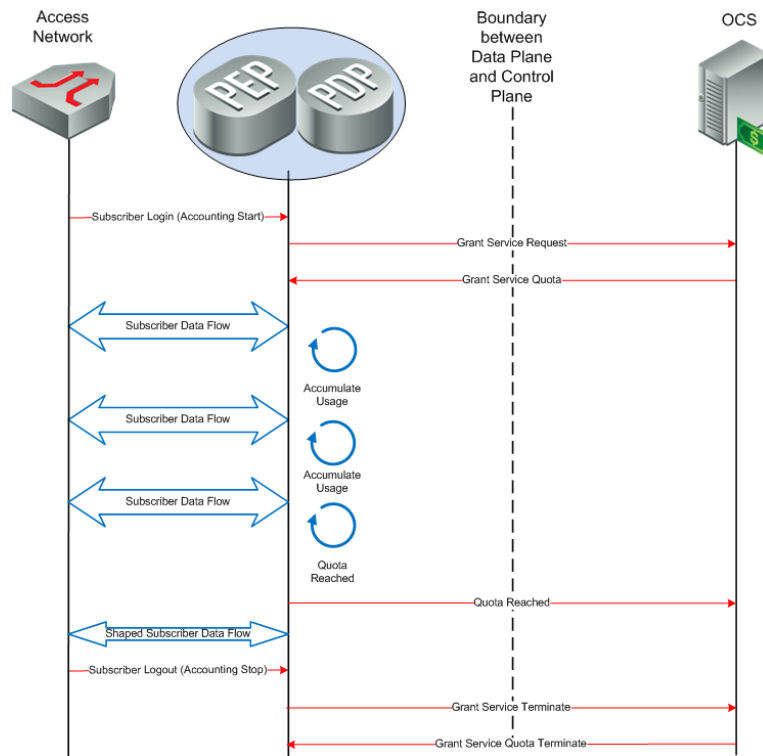


Figure 9: Message Flows with Distributed Architecture (Pre-Paid Quota Management)

Observations

Once again, it is clear that the distributed architecture involves much lower control plane traffic and reduced decision latency.

Analysis Conclusions

Scalability: Transaction Overload

Put simply, it is impossible to scale when there is a single point responsible for all policy decisions. The messaging intensity of a PCC architecture is determined by the location of the required data, the data's dynamism and the functional partitioning which then requires access to the data. It is most logical to allocate decision functionality to the point where the data is most dynamic - on the data plane.

Many operators are reporting a *signalling overload* upon deploying centralized decision-making architectures. Consider these real experiences from five different Tier-1 service providers, representing both fixed and mobile providers from three continents:

"...our transaction rates are so high that adding more elements to the admission chain has become prohibitive."

“...the message rate got so high that we needed a separate device to load-balance across multiple OCS nodes.”

“...the convergent service platform has caused us much grief due to limited scalability to deal with real-world transaction rates.”

“...our PCRF development has been delayed for almost a year due to its limited performance.”

“...in order to support enough Diameter transactions per second, we need to fill an entire rack of servers - just for roaming quota enforcement!”

But why is this the case? It often comes down to the sheer number of data flows combined with relatively short flow lifetimes. Recall the Google study described in the URL Access Control use case.

In terms of pure efficiency, it is appropriate to only have central decisions made remotely when a subscriber-specific condition has changed (thereby impacting entitlement such as QoS or usage quotas, and even then as described in the Pre-Paid Quota Management use case analysis this can also be distributed to a large extent). By extension, network-related conditions are handled locally by the distributed network element. Network conditions can be understood, and decisions based on network conditions can be made directly, by the intelligent enforcement layer on the data plane instead of asking for permission to implement an action that could have already been known.

Figure 10 shows how a decision-making hierarchy can cut the number of policy transactions by orders of magnitude at each functional layer (the absolute numbers in this figure are dependent upon deployment characteristics and are meant primarily to illustrate the relative ratios of messages between the planes). The data plane, which makes decisions on a session, flow and packet level, tackles millions of transactions per second. By distributing these decisions across many joint PDP/PEP elements, each instance has a manageable number of transactions and the model scales to actual network environments.

The control plane is only called into matters for a subset of the data plane's transactions and experiences anywhere from 100 transactions per second to 50,000 - still a much lower number than that facing the data plane. The control plane tends to focus on subscriber and session-level decisions.

The management plane addresses a much lower rate of transactions by virtue of only being called upon for subscriber-level decisions like provisioning.

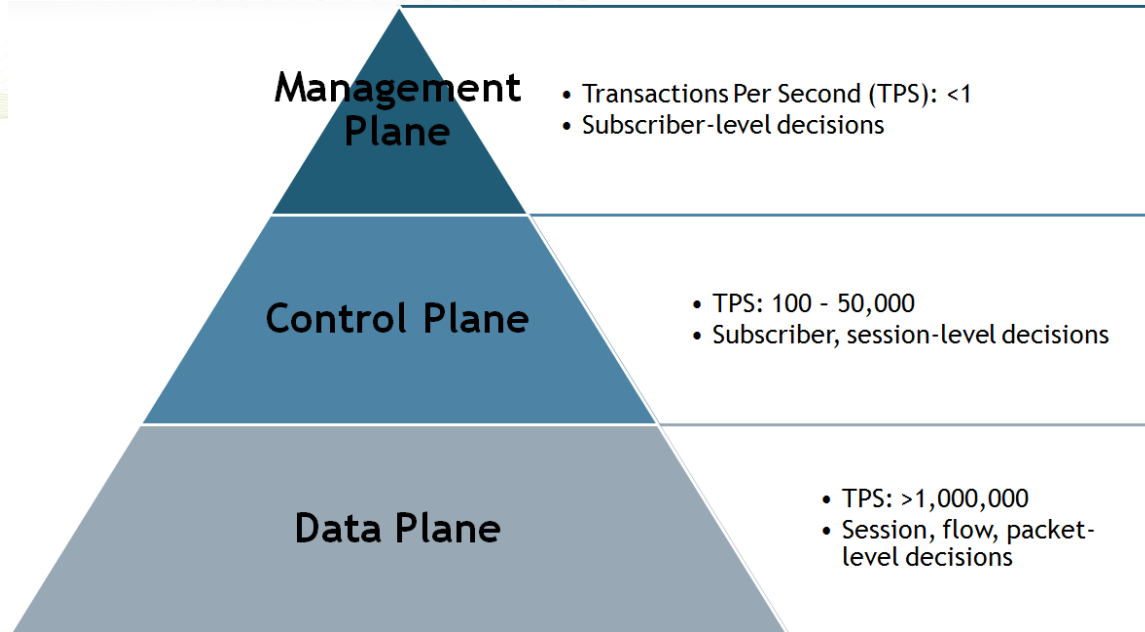


Figure 10: Relative transaction rates and focus of decisions for layers within a distributed network policy control architecture

Decision Latency: Microseconds Matter

Consider traffic characterized by short flow lifetimes, approximately 100ms, such as might be created by an email client checking mail or a smart-phone application querying for status updates. Assuming decision latency (to actually determine what to do) of 50ms and round trip signaling latency of 20ms between a distributed node and a central brain, then the centralized architecture introduces 70ms of latency.

During those 70ms, the majority (70 percent) of the flow in question has either already traversed the enforcement point before any action can be taken or the flow needs to be queued and delayed by 70 ms. In the latter case, network policies will not be functioning as designed, and in the former, the subscriber quality of experience suffers immensely. This centralized architecture therefore suffers from a fatal flaw in that it cannot adequately perform in a typical Internet environment.

By empowering distributed devices to perform local network element flow identification and service enforcement when conditions are well known (the vast majority of cases), then the 70ms latency is reduced to the order of microseconds. A classic example of empowering the distributed device to perform local service enforcement is that of real-time quota enforcement where the distributed device can immediately block or redirect the subscriber when the quota limit is reached versus asking the centralized policy device what to do and either delaying traffic (queuing up) or letting some traffic pass through.

Putting it into Practice: Sandvine's Distributed Network Policy Control

Sandvine's network policy control architecture is designed based on the principles outlined above - that network policy should be executed on the lowest element in the network that has access to all of the required information. This approach maximizes scalability by reducing centralization and maximizes performance and minimizes latency by removing unnecessary communication between elements.

The PTS identifies and measures traffic, makes local policy decisions and implements enforcement actions. The SPB acts as a subscriber profile repository (SPR) and maintains the active state of each subscriber. Additionally, the SPB serves as a database storage and reporting platform and hosts the rules defining the network's policy control functionality. The SDE is a policy decision and enforcement element that works together with the PTS to provide unified network policy control within various network environments. The SDE is also able to signal enforcement decisions to additional network elements.

Sandvine's architecture mirrors that of the distributed design discussed previously and Figure 11 shows the mapping of functions within this hierarchy to Sandvine's platforms:

- **Policy Traffic Switch (PTS)** - identifies and measures traffic, makes local policy decisions and implements enforcement actions
- **Service Delivery Engine (SDE)** - policy decision and enforcement (through signaling) element that provides unified network policy control across the entire access network and vendor domain (in the generalized distributed architecture examples in the use cases, the SDE would function as the policy arbitrator)

Since the Sandvine policy engine is present on both platforms, policy can be executed at the appropriate location in the network policy control hierarchy.

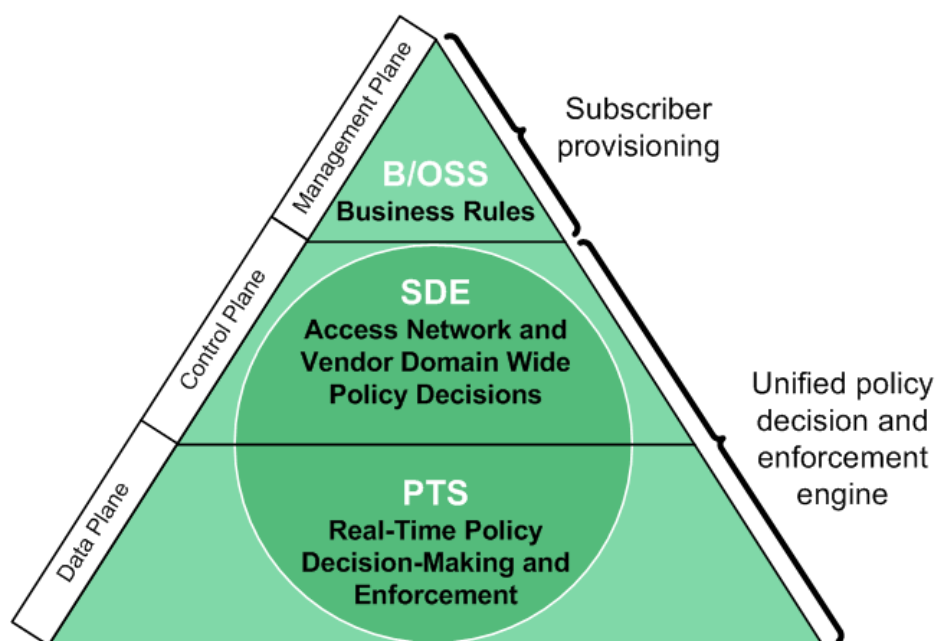


Figure 11: Sandvine Network Policy Control Pyramid