DISTRIBUTED RESOURCE MANAGEMENT FOR ON DEMAND SERVICES

Bruce Thompson
Cisco Systems Inc

Abstract

Today's video-on-demand (VoD) systems are centralized in many respects – resource management, session management, and storage. However, there are many benefits from distributed architectures for some or all of the above. Many of the lessons that have been learned from widespread deployments of VoIP and other content-based networks can be applied to VoD networks. Today's video servers in many cases actually perform several logical functions, including session based resource management, storage, video streaming, etc. Separating, at least at a logical level, many of these components can provide significant benefits in terms of scalability, cost, and performance.

One candidate for breaking out from the components that actually store and stream the video is session and resource management. And beyond simply breaking out the different logical functions from each other, each individual function can be also done in either a centralized or distributed way themselves. It's the "all knowing database in the sky and dumb edge boxes" vs. "smarter edge devices that manage their own capabilities" debate that has raged on for years in networking, most recently manifesting itself in the VoIP debates.

To enable distributed resource management, network components responsible for allocating different classes of session-based resources can be separated from the session and resource manager. A component called the session manager is then responsible for determining the classes of resources required for a session request and communicating with the resource managers responsible for allocating those resources. There are two predominate schools of thought on how the session manager interacts with resource managers.

The first has the session manager responsible for the final decision on all resources. This model is a logical outgrowth of session resource management vendors who currently implement both session and resource management in the same node. In this model, the session manager would receive a session setup request, and then send individual messages to various resource managers to ask for the potential resources that are available for that session. Then having all of the information locally in one place, the session manager decides which resources should be used for that session. This model is similar to the centralized call manager model used in some VoIP networks.

Alternatively, session resource allocation can be distributed in more of a flow-through. In this model, the session manager would receive a session setup request, determine the classes of resources needed for the session and then let the resource managers themselves determine the actual resources to be allocated for the session. Instead of the session manager communicating directly with each resource manager, it embeds a list of resource managers that must be used into a resource request and sends the resource request to the first resource manager on the list. Resource managers then communicate with each other to determine the optimal set of resources for the session. This threaded model of signaling, takes advantage of the concept of a signaling proxy which is common in Internet protocols such as HTTP, RTSP, SIP, and others. Internet deployments

of signaling proxies have shown that they are very adaptable to changes in services and that they scale to very large networks since functionality is distributed.

This paper will go into detail on the pros and cons of distributed versus centralized control architectures for session and resources management. It will cover specific call flow examples, processing requirements, scalability concerns, and where possible real world examples of how this has been done in existing networks today.

OVERALL FUNCTIONAL BREAK DOWN

The DSM-CC defines a general architectural model and signaling protocol for client server applications. One of the architectural components the DSM-CC specification defines is the session and resource manager (SRM). The SRM is responsible for allocating resources for sessions between a client and a server. To enable distributed resource management, the function of the SRM can be broken into a component that is responsible for session management (the session manager) and multiple components called resource managers.

In this architecture, each resource manager is responsible for allocating a class of resources for a portion of the video topology. For example, a resource manager called the edge resource manager is responsible for allocating QAM resources for a population of set-top boxes connected to the HFC plant. Another resource manager called the encryption resource manager is responsible for allocating real time encryption resources for a population of set-top boxes. The on demand resource manager is responsible for determining the best video server to use for streaming a particular asset to a set –top box.

The session manager provides the interface between DSM-CC session components and resource allocation components. In the session space, the session manager interfaces to the on demand client running on a set-top box and to service application which is typically bundled in the VoD server. In the DSM-CC architectural model, the session manager appears as the DSM-CC SRM, the on demand client appears as the DSM-CC client, and the service application appears as the DSM-CC server. The on demand client initiates session requests on behalf of subscribers and receives tuning information as the result of those requests. The service application implements service and business logic. It uses the service and business logic to specify the classes of resources required for a particular session. The session manager takes the classes of resources specified by the service application and interfaces with the resource managers to obtain the resources required for the session.

Figure 1 illustrates the classes of components used to implement distributed resource management for on demand services.

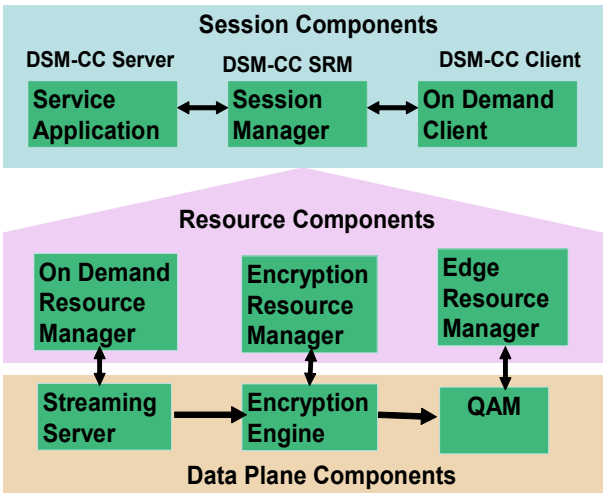

Figure 1: Distributed resource management components

Session & Resource Management Functional Break Down

There are 3 major functions that need to be implemented as part of session and resource management for on demand services. They are:

1. Business Logic

Business logic is used to tie business rules to a particular service and to a particular session/resource request. Rules associated with business logic can be translated into parameters of a resource request.

For example, resource requests for a pay per view service may have a higher priority associated with them than resource requests for a free on demand service. This priority can be carried as a parameter within a resource request. Rules associated with business logic can also be used to determine which resource managers should be used for a particular session request.

2. Video topology routing

The HFC network that set-top boxes and edge devices are connected to is divided into multiple broadcast segments that are combined with RF splitters and combiners. In the resulting network, the combination of the RF segment that an edge device is connected to and the RFC frequency that the edge device uses as a carrier define which set- top boxes can be reached by that edge device. Resources allocated for on demand session requests generated by set-top boxes must take into account the connectivity of the HFC network. In addition, the IP transport network may not be fully connected or may have paths between components that are not suitable for carrying video.

HFC and IP network connectivity will affect the selection of edge resources and may also affect the selection of encryption and streaming server resources. The session and resource management architecture must take the HFC and IP network topology into account when determining which resource managers as well as which resources should be used for a particular session.

3. Resource Allocation

Once the classes of resources required for a particular session are known and a set of resource managers are chosen to allocate those resources, the actual resources to be used for a session need to be chosen.

The resources selected for a particular session need be optimized to take into account individual resource availability as well as the connectivity/availability of bandwidth in both the HFC and IP networks

In both of the resource allocation architectures described in this paper, the session manager implements both business logic and video topology routing. The difference between the architectures is in how the actual resources to be used for the session are negotiated between the session

manager and the resource managers the session manager has selected for the session.

The paper will also briefly describe a method by which the tables used for video topology routing can be populated dynamically using protocols similar to those used to implement dynamic routing in IP networks.

Centralized Resource Allocation

In centralized model for resource allocation, the logic for deciding which resources should be used for a particular session is homed on the session manager. In this architecture, the resource managers themselves simply report available resources to the session manager while the session manager looks at all resources from the resource managers and decides which resources should be allocated for a session.

With centralized resource allocation, all resource requests for a session originate from the session manager. The session manager sends a resource request message to each resource manager. Each resource manager then responds with a list of potential resources (QAMs, Encryption Engines, etc.) that may be used for the session. The IP transport address of each potential resource is included in the response.

Based on the responses, the session manager picks a set of resources to be used for the session. In order to make an optimal choice of resources, the session manager should take connectivity and available bandwidth in the IP network into account as part of the algorithm for selecting the resources to use for the session. Once the session manager has picked a set of resources to use for the session, it informs each resource manager which resource was selected for the session. This second message exchange is also required to exchange IP transport addresses between data plane components selected for the session. Figure 2 illustrates the resource signaling flow for centralized resource allocation.

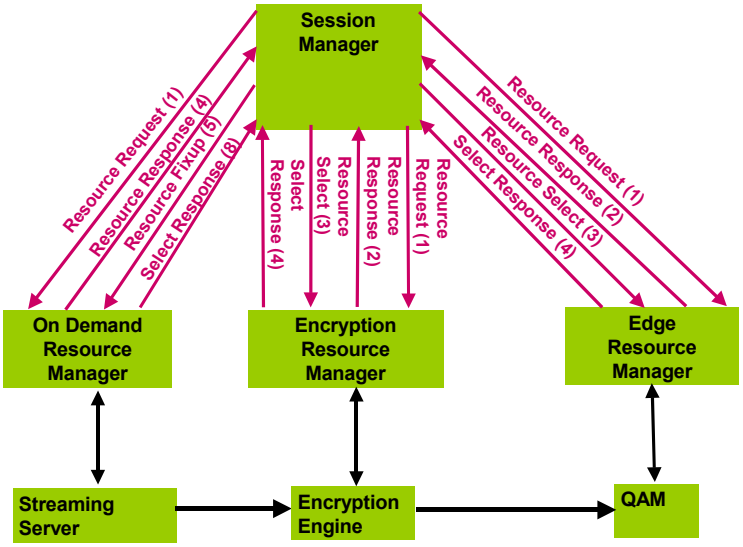

Figure 2: Signaling flow for centralized resource allocation

Pros and Cons of Centralized Resource Allocation

Since the session manager makes the final decision for resource selection, the centralized resource allocation architecture provides the most flexibility in making tradeoffs between between business logic and resource allocation logic.

While centralized resource management allows flexibility, there are a number of limitations associated with it.

With centralized resource allocation, the session manager must implement all of the logic of resource allocation. In addition, it must coordinate all of the individual resource allocation transactions between each resource manager. Because of this, the session manager becomes the most complex component of the system to build. Also, since most of the per session processing falls on the session manager, it will be the performance bottleneck of the system.

With centralized resource allocation, both session and resource state are shared between the session manager and resource managers. The amount of state that is shared between components complicates fail over and redundancy methods for the session manager and resource managers. When either the session manager or a resource manager fails over to a redundant system, both session and resource state must be resynchronized between the session manager and resource managers.

Since the session manager must coordinate transactions between multiple resource managers, the process of cleaning up after failed resource requests can be quite complex. If any request fails, it must abort all resource requests that have successfully completed along with all outstanding resource requests for that session. This greatly increases the complexity of the implementation of the session manager and may lead to software errors where the resources of sessions that have failed end up getting "orphaned". Orphaned resources must be cleaned up by either rebooting the entire system or by running special processes that check for resource consistency between the components of the system.

There are a large number of messages that must be exchanged between the session manager and resource managers in order to allocate resource for a session. At least 2 message exchanges (4 messages) are required between the session manager and each resource manager to allocate resources for a session. The first message exchange is needed to obtain the potential resources from each resource manager while the second message exchange specifies the selected resource and also provides IP transport address information for the data plane components to be used for the session.

Distributed Resource Allocation

The distribution of functionality and as well as the signaling model used for distributed resource allocation is similar to the model used for the Web as well as streaming and interactive audio and video services on the Internet.

With distributed resource management, the session manager directs the resource managers to communicate between themselves to select the optimal set of resources to use for the session. Delegating

resource allocation to the resource managers off loads this function from the session manager and results in a more even distribution of functionality/processing load between the session manager and resource managers.

Each resource manager appears as a signaling proxy in the signaling model used for distributed resource allocation. Each resource manager examines the parameters of an incoming resource allocation request and allocates resources based on the set of parameters it understands. Each resource manager then modifies the parameters of the request based on the resources that were allocated. Finally, each resource manager passes the request to the next resource manager based on a list of resource manager addresses that the session manager embeds in the initial resource allocation request. The proxy signaling model enables resource managers to efficiently negotiate the optimal set of resources to be used for a session using a minimal number of message exchanges.

The session manager includes an ordered list of resource managers to be contacted for a particular session in the initial resource request. The session manager determines the order that resource managers must be contacted based on the order that the MPEG stream must flow through the data plane components that each resource manager controls. The order that resource managers must be contacted is opposite of the order that the MPEG stream flows through the data plane components. For example, the MPEG stream associated with a VoD session that requires encryption will originate from a streaming server and pass through an encryption engine on the way to an edge device. The resource managers associated with these data plane components are the on demand resource manager, encryption resource manager, and edge resource manager. Because the video stream originates from the streaming server, its associated resource manager (the on demand resource manager) must be contacted last. Likewise,

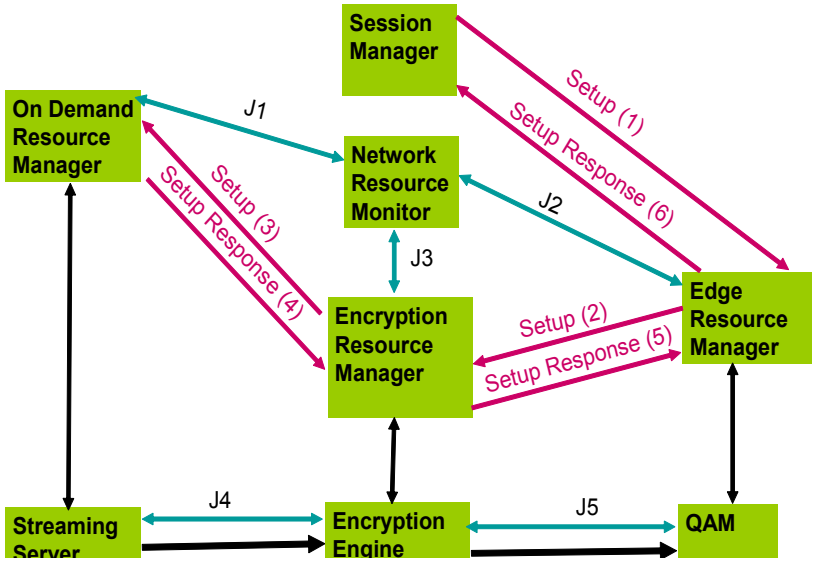

Figure 3: Signaling with distributed resource allocation

since the edge device is the last component that the video stream passes through, its associated resource manager (the edge resource manager) must be contacted first.

This ordering results in resource managers whose data plane components are adjacent being able to exchange IP transport parameters needed for the MPEG media stream very efficiently.

IP transport network resources are also allocated in a distributed fashion. With distributed resource allocation, network resource allocation is broken into two logical functions. They are: network resource monitoring and network resource allocation. Network resource monitoring is a function that provides information on the available network bandwidth between data plane components while network resource allocation reserves network bandwidth between data plane components. The threaded architecture implements the network resource monitoring function in a component called the network resource monitor and implements network resource allocation in the IP transport network itself.

RTSP can be used as the resource allocation protocol used between the session manager and the resource managers. RTSP is well suited for use in this architecture since RTSP was designed to support proxies as part of the signaling architecture. Figure 3 shows the order of signaling flow for distributed resource allocation using RTSP as the signaling protocol. In Figure 3, interfaces J1, J2, and J3 are interfaces the resource managers use to request information on available network bandwidth from the network resource monitor. Interfaces J3 and J4 are used to request resources from the transport network.

RTSP enables each resource manager to implement a 2 stage reserve/commit process for allocating resources. Figure 3 will be used to illustrate the 2 stage process. The first stage of the resource allocation process is triggered by a resource manager receiving an incoming RTSP setup(1) request from its down stream neighbor. The resource manager examines the parameters of the RTSP setup message it received to determine which resources it needs to allocate for the session. In Figure 3, when the RTSP setup(1) message is received by the edge resource manager, it allocates resources from one or more QAMs that can be used to service the session. Once the resource manager reserves resources it passes the data plane transport parameters associated with the resources it reserved to the next resource manager in an RTSP setup(2) message.

The next resource manager in the chain allocates its resources based on the parameters of the RTSP setup(2) request. The resource manager may check to see if transport bandwidth is available between the potential data plane components it has selected and the data plane transport parameters it received in the RTSP setup message from its down stream neighbor. This check can be used to trim the list of resources that are reserved. In Figure 3, the encryption resource manager reserves resources associated with one or more encryption engines when it receives the incoming setup(2) message from the edge resource manager. The encryption resource manager may check to see if there is available network bandwidth between the rncryption rngines it has selected and the transport addresses it received from the edge resource manager in the setup(2) request.

Once the resource managers have reserved resources in response to the RTSP setup message, the resource commit process starts. The last resource manager in the thread commits a specific resource to be used for the session. This resource manager then requests the transport network to reserve network bandwidth between itself and its down stream neighbor and passes the data plane transport parameters for the chosen resource in the RTSP setup response(4). In Figure 3, the on demand resource manager picks a specific streaming server and output port to use for the session. The on demand resource manager passes the source IP address / port number that was selected for the session in the RTSP setup response(4).

As the RTSP setup response is passed back through the thread, each resource manager examines the resources that were committed by its up stream neighbor. It uses this information to commit its own resource. In the process of committing its own resource, each resource manager requests the transport network to reserve network bandwidth between itself and its down stream neighbor and passes the data plane transport parameters for the chosen resource in the RTSP setup response. In Figure 3, the edge resource manager commits a single QAM resource when it receives the RTSP setup response(5). It passes the HFC parameters such as the modulation frequency and MPEG program ID back to the session manager in the RSTP setup response(6).

Distributed IP Network Resource Allocation

With distributed resource allocation, each resource manager takes IP transport network resources into account as part of the decision of which resource will be used for a session. Each resource manager may use the services of a component called the network resource monitor to determine which components/ resources to choose for a particular resource request based on available network bandwidth to downstream components already selected for that session.

The network resource monitor (NRM) contains a map of the IP transport network topology and the amount of available bandwidth on every IP transport link in the VoD topology. The NRM is implemented as an internal routing protocol (IRP) listener (OSPF, ISIS, etc) and knows how much network bandwidth is currently available between any 2 IP endpoints. To allow the NRM to understand bandwidth availability, the IRP must include bandwidth availability as part of the link state attributes that are flooded. To obtain available bandwidth information on each link, the IRP must interface to the Network Resource Allocation subsystem in each router.

With distributed resource allocation, IP transport network bandwidth is allocated by the routers in the IP transport network themselves. IP networks have traditionally used a distributed approach to both path selection (IP routing protocols) and managing network bandwidth (RSVP).

Distributed resource allocation uses a combination of RSVP in the control plane and the DiffServ architecture in the data plane to allocate bandwidth and minimize jitter and latency for MPEG video in the IP transport network.

RSVP messages are passed between the data plane components responsible for originating and terminating the MPEG video stream. The components from Figure 3 that originate/terminate RSVP requests are the streaming server, encryption engine, and edge device. RSVP messages pass through the routers of the transport network along the same path as the MPEG video stream. As

RSVP messages pass through the transport network, they are examined by every router along the path. When a router receives an RSVP request message, it first looks up the outbound interface that the packet will be routed to and then checks to see if that interface and its associated transport link have enough bandwidth to service the request. If it does, available bandwidth for that interface is decremented and the RSVP message is forwarded to the next router along the path. If any router along the path does not have enough bandwidth to service an RSVP request, the router fails the request and the message is returned along the same path that it was originated from, tearing down existing state long the way.

When RSVP and DiffServ are used together in the network, RSVP is used to provide admission control functionality while DiffServ provides the per packet scheduling characteristics required for MPEG video. RSVP works together with DiffServ at the edge of the network to ensure that MPEG video flows admitted by RSVP are marked with the proper DSCP value to ensure proper QoS treatment through the rest of the network.

Pros and Cons of Distributed Resource Allocation

Distributed resource allocation resolves many of the issues associated with centralized resource allocation.

Many of the performance issues and complexities associated with the session manager in centralized resource allocation are dealt with by moving the function selecting an optimal resource set of resources for an on demand session to the resource managers themselves. With distributed resource allocation, the session manager simply creates a list of resource managers that need to be involved in the session and forwards a resource allocation request to the first resource manager in the list. The rest of the resource management processing is done by the resource management components specified in the list. This distributed model of signaling and synchronization more evenly distributes the processing load between the session manager and the resource managers involved in a session.

Since the resource managers negotiate and communicate allocated resource information between them selves, the only state that is shared between the resource managers and the session manager is active session information. This simplifies failure / recovery mechanisms for the session manager and resource managers since the session manager and resource managers only need to have a consistent view of active session state.

With distributed resource allocation, the ordering of resource requests greatly simplifies error processing when a request fails for any reason. When a resource request to a resource manager fails for any reason, it responds with an error to the resource manager that sent the request. Each resource manager then de-allocates the resources it allocated on the resource allocation request and forwards the error to its neighbor. This synchronous behavior greatly simplifies error processing logic.

With distributed resource allocation, a single resource allocation protocol is used between the session manager and the resource managers. This allows each resource manager to be implemented using the same resource signaling state machine. It also enables new classes of resource managers can be added without having to modify any of the existing resource managers or the signaling state machine of the session manager.

The signaling model used for distributed resource allocation uses a minimal number of message exchanges to allocate resources for a session. This is because the proxy signaling mdoel used for distributed resource allocation only exchanges messages between the components that must exchange information in order to set up the session. Most of the information that needs to be exchanged in order to setup a session needs to be exchanged between data plane components that are adjacent in the flow of the media stream. .For example, to set up the media stream between an encryption engine and an edge device, the edge and encryption resource managers must exchange the IP source and destination addresses along with the UDP port numbers to be used for the session. The proxy signaling model uses a single message exchange between adjacent components to exchange transport information. With centralized resource allocation, 2 message exchanges are required.

Dynamic Routing of Video Sessions

When resource management functionality is distributed to multiple resource managers, each resource manager is responsible for allocating a class of resources for a portion of the video topology. One of the functions of the session manager that is common to both the centralized and distributed resource allocation models is the ability for the session manager to select a set of resource managers to use for aparticular session based on the constraints of the HFC and IP topologies. This function is called video topology routing.

This section describes a method by which the tables used for video topology routing can be populated dynamically using protocols similar to those used to implement dynamic routing in IP networks. Using a dynamic method for populating video routing tables allows the session manager to re route video sessions when resource managers fail or when new resource managers are added to a system. It also allows the session manager to choose the best set of resource managers to use for a particular session based on both static and dynamic cost metrics associated with individual resource managers.

When a DSM-CC session request is sent from an on demand client to the session manager, it typically includes 1 or more identifiers that can be used by the session manager to determine where the clients Set Top is located in the HFC plant. The identifier may be statically provisioned in or it may be dynamically learned. One method to enable a set-top box to obtain topology identifiers dynamically is to configure each QAM with a unique identifier and then have the QAM send that identifier in the MPEG-2 transport stream it generates. The TSID field in the MPEG-2 transport stream is commonly used for this purpose. To obtain these identifiers, a set-top box periodically tunes to a set of pre defined “beacon” channels and stores the TSID information it receives on those channels.

When multiple QAMs are fed into an RF combining network, their RF output is broadcast to all of the set-top boxes connected to the same RF network. The combination of QAMs and set-top boxes that can be reached in this RF network is typically referred to as a serving group or serving area. edge resource managers manage QAM resources for one or more serving groups.

When an on demand client sends a DSM-CC session request to the session manager, it includes the list of TSIDs it has learned in the session request. When the session manager receives the session request, it must map the TSID information it receives in the session

request to the IP address(s) of one or more edge resource managers that can be used to service this request.

The protocol described in this section enables the session manager to build a map that translates a set of TSIDs to the HFC serving group and the IP addresses of one or more resource managers responsible for allocating resources of that HFC serving group. This protocol is called the service registration protocol.

In addition to providing topological information, the service registration protocol can be used to provide other operational information about QAMs or other video nodes such as encryption engine. For example, the service registration protocol should be capable of providing information about the capabilities of an encryption engine. Another important requirement for the service registration protocol is to support the ability to detect the failure of an QAM or an encryption engine in a timely manner. This capability will allow the session manager to reroute video streams from a failed QAM or encryption engine to QAMs or encryption engines that are still operational. It will also allow the SM to route new video streams to the EQAMs/encryption engines that are still operational.

Finally, when more than one resource manager has resources capable of satisfying a session request, the service registration protocol can provide a static administrative cost associated with that resource manager as well as a dynamically updated cost that can be used to reflect information such as the amount of resources currently available from the resource manager.

RFC 3219[7] (TRIP) is an IETF protocol that deals with the problem of translating telephone numbers into the session signaling address of a telephony gateway that can be used to reach that telephone number. While TRIP was originally developed for telephony services, a subset of the protocol can be used to dynamically populate the tables the session manager uses to perform video topology routing.

TRIP allows a telephony gateway to advertise the telephone numbers it serves along with its session signaling protocol address to other telephony gateways. A telephony gateway that sends TRIP advertisements is called a TRIP speaker and a telephony gateway that receives TRIP advertisements is called a TRIP listener.

When TRIP is used distributed resource management, edge resource managers act as TRIP speakers and the session manager acts as a TRIP listener. Edge resource managers advertise the TSIDs and serving groups of the QAMs they manage as well as their IP address. The session manager uses this information to develop a the video topology routing map.

When TRIP is used with other types of resource managers such as the encryption resource manager it will be used to advertise the IP address of the resource manager along with other properties such as the conditional access protocols supported by the encryption engines. Since data plane components such as encryption engines may not be directly connected to the HFC network, TRIP can be used to advertise an administrative cost to the different HFC serving groups that can be

reached by that component. Figure 4 illustrates the role of in a distributed resource management environment.

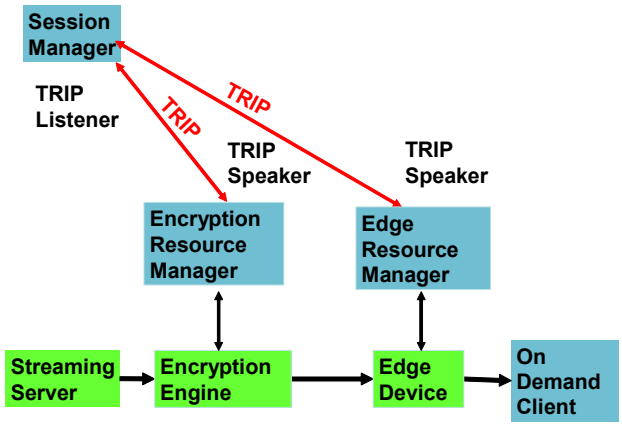

Figure 4: Role of TRIP in distributed resource management

SUMMARY

This paper described a few alternative methods for implementing distributed resource management in for On Demand Services. It described a model for breaking the Session and Resource Manager (SRM) used in current On Demand architectures into components responsible for session management and resource management.

Two alternative architectures for implementing distributed resource allocation were presented. In the centralized resource allocation model, the Session Manager is responsible for coordinating resource allocation between the resource managers selected for a particular session request. In the distributed resource management model, the the function of selecting an optimal set of resource for a particular session is distributed to the resource managers themselves. While the centralized resource allocation model is a natural fallout from existing SRM implementations, the distributed resource allocation model allows the implementation of the Session Manager to be greatly simplified and allows the network to scale by using protocols and architectural methods that have been widely deployed in the Internet.

Finally, the application of a telephony routing protocol (TRIP) for distributed resource management in On Demand networks is described. The use of dynamic protocols such as TRIP in On Demand networks will allow these networks to scale as On Demand services become more popular.