Internet Engineering Task Force
INTERNET-DRAFT
draft-schulzrinne-nsis-casp-00.ps

H. Schulzrinne, H. Tschofenig, X. Fu, J. Eisl, R. Hancock
Columbia U., Siemens,TU Berlin,Siemens,Siemens-Roke Manor

September 15, 2002
Expires: January 2003

CASP - Cross-Application Signaling Protocol

Status of this Memo

Copyright Notice

Abstract

The Cross-Application Signaling Protocol (CASP) is a general-purpose protocol for managing state in routers and other on-path network devices. It can be used for QoS signaling, middlebox control, topology discovery, measurement data collection, active network instantiation and any other application where state needs to be established along a data path. CASP consists of a set of building blocks that can be used to construct protocol behavior suited for a particular application. It is transport-neutral, network-friendly and securable. This document describes the usage-independent components of CASP.

Contents

1 Introduction

1.1 Protocol Overview

The Cross-Application Signaling Protocol (CASP) provides a generic *signaling* service by establishing state along the data path from a sender to one receiver, for unicast data, or multiple receivers, for multicast data. CASP sessions can be initiated by the sender or the receiver.

CASP is not restricted to sender or receiver-initiated reservations; it can be used for a variety of signaling purposes. Examples include resource reservation in both in-path and out-of-path modes, configuration of middleboxes [1] such as firewalls and NATs, distribution of code segments in active networks, network diagnostics, and MPLS label distribution.

CASP does not place restrictions on the location of signaling initiators and receivers. They can be the same as the data sources or sinks, or can be separate hosts (“proxies”).

CASP consists of two layers, the client (C) and messaging (M) layer, as shown in Fig. 1.

```
+-----------------------------+ +-----------------------------+
|                             | |                             |
|    CASP client (C) layer    | |      Discovery protocol     |
|                             | |        (Scout protocol)     |
|                             | |                             |
+-------------------------------------------------------------+
|                                                             |
|                   CASP messaging (M) layer                  |
|                                                             |
+-------------------------------------------------------------+
|                             | |                             |
|  reliable transport layer   | |            UDP              |
|      (TCP, SCTP, ...)       | |                             |
+-----------------------------+ +-----------------------------+
```

Figure 1: CASP Protocol Layering

The motivation for a separate client (application) and messaging layer is given by [2]. For the reasons discussed in Section 5, CASP uses existing transport protocols.

CASP establishes *sessions*. A CASP session consists of all CASP messages that refer to the same state, traverse the same path or parts of it and share the same session identifier, independent of which direction messages travel.

The origin and destination of the C and M layers have to be the same for each CASP session.

Each CASP message consists of two parts: an application-independent part that handles message routing, discovery and feature negotiation (i.e., the messaging layer), and an application-dependent part that is carried as a payload inside the client layer. We refer to the protocol elements carried in the payload of CASP messages as the *client protocol*. A sample client protocol for resource reservation is described in [3]. This memo describes the messaging and transport layer common to all CASP client protocols, the common attributes required of client protocols, and a specialized client protocol, namely the scout protocol for next-peer discovery.

CASP can establish state for its own use in forwarding messages; the client layer can establish its own state. Expiration of the message-layer state triggers removal of the client-layer state, but the converse may not be true. We assume that client layers within the same messaging layer session share fate and trust.

Network nodes that process CASP messages are called CASP nodes. CASP nodes are divided into *omnivorous* and *selective* nodes. An omnivorous nodes processes all CASP messages, even if it does not understand the client protocol and thus ignores the client layer object. For example, a CASP node residing at a firewall or NAT may need to see each CASP message, so that it can inspect and modify the traffic selector.

A selective CASP node is only interested in messages that carry one of the client protocols it supports. It is not visited by any other CASP message.

We call the sequence of omnivorous and selective CASP nodes traversed by a CASP message between the NSIS Initiatior (NI) and the NSIS responder (NR) [4, 5] *CASP chain*.

1.2 Protocol Properties

CASP has the following properties:

Layered: CASP is a layered protocol with two layers, the client and messaging layer. Each can be changed without affecting the other component. A separate *discovery* component determines the next CASP peer, which can be either the next router, following the data flow direction, or some other node. One example of the discovery component is the *scout* protocol, a CASP client protocol, that discovers the next CASP peer.

CASP uses the services of a reliable transport protocol that provides sequenced, reliable, flow- and congestion-controlled message transport between two CASP nodes. The *messaging* layer provides state identification, peer-to-peer message routing and other functionality common across all client layers. The *client* layer contains the application-specific components.

A single CASP messaging layer session can be used by multiple client states, to ensure that all client states are removed at the same time. As an example, active networking or firewall traversal state may be bound to QoS state. C state can only exist as long as the underlying M state exists.

Network-friendly: While most signaling messages for classical signaling applications are likely to be small and the overall data volume modest, CASP recognizes that there are potential applications that may need to deliver larger volumes of data or larger packets. For example, instantiating active network nodes may require payloads that are significantly larger than typical network MTUs. Similarly, cryptographic signatures may cause even common signaling messages to exceed MTU size. Thus, CASP would have to deal with fragmentation if it were to implement its own reliability mechanism.

Also, we believe that the total volume of signaling information between two nodes can be substantial, even if each signaling flow only contributes a message every few tens of seconds. For example, if we assume resource reservation for a VoIP application with 3-minute calls, four 500 byte signaling messages (for establishment and teardown and the responses), a 45 Mb/s access link could see about 64 kb/s of signaling traffic, which is a modest overhead relative to the useful application data (about 700 simultaneous calls), but still larger than many applications. Also, during overload situations, user applications will be tempted to retry their reservation requests frequently, so that congestion and flow control is desirable.

Thus, instead of using its own retransmission mechanism for each session within the messaging layer, CASP establishes a soft-state peer session. Unlike in BGP, these are established on demand and can be torn down after periods of inactivity. Such sessions are likely to result in significantly improved performance for each signaling flow, since retransmission can use better estimates for round-trip times and can reduce the time to loss discovery to multiple packet spacings plus a one-way delay rather than a conservative estimate of the round-trip time. Given that almost all nodes will already have support for a transport protocol, this approach is likely to greatly reduce the complexity of protocol implementations and avoid subtle interoperability problems.

Re-using peer sessions also reduces the number of cryptographic computations. Reusing an already established security association at the transport layer and possibly at the client layer avoids expensive security association establishment when a new connection is set up. TLS with session resuming (RFC 2246 [6]) can further reduce the impact of establishing a new transport association. In case of IPsec, SAs are valid for a given time period (if the SA lifetime is bound to a time duration and not to the number of transmitted bytes) and operates at a lower layer uneffected by TCP and SCTP connections.

Thus, new CASP sessions will only very rarely suffer from delays caused by setting up a transport connection. (We expect that frequent session setup will only be necessary if a CASP node exchanges messages with several thousand or more peer nodes with equal frequency, so that maintaining transport sessions becomes infeasible.)

Transport-neutral: CASP is transport-protocol neutral. Each peer could use a different transport protocol, with TCP and SCTP [7, 8] as RECOMMENDED protocols. Using TCP and SCTP also allows it to use channel security for protection of messaging and client layer messages in a peer-to-peer mode and mutual-authentication via TLS. Note that IPsec can be deployed independent of any upper layer transport protocol. The messaging layer only assumes that the transport layer offers reliable, sequenced message or byte stream delivery with flow and congestion control.

Use of TCP or SCTP does not necessarily make CASP NAT-friendly [9], since it carries network addresses by necessity. It may, however, simplify traversal of firewalls.

Policy-neutral: CASP does not impose a particular AAA or usage policy, but can carry necessary information for AAA protocols such as DIAMETER [10] or public-key credentials in the messaging or client layer. (We make no claim that CASP can support all AAA architectures or support them equally well.)

Soft state: CASP provides a generic soft-state mechanism that can be used by all client protocols. Soft state is only used for logical state, not to deal with packet loss. To maintain soft state, requests are simply resent periodically by each node. Refresh periods can vary among CASP nodes. Nodes can also remove state explicitly.

> Peer-to-Peer refresh was chosen to allow different choices by each administrator, e.g., to enforce uniform values within an autonomous system or to control resource usage. Non-uniform refresh intervals mean, however, that islands of state can persist. Also, dead applications need to be detected at the client layer.

Extensible: CASP is extensible. It consists of a sequence of, possibly nested, type-length-value (TLV) objects. Extension objects can be added at any time. Protocol features are negotiated via feature negotiation, not as individual objects. This allows semantic negotiation such as "node X does not support mandatory feature Y" rather than low-level indications that some combination of objects is not supported. Nodes can add and modify objects. Cryptographically protected client-layer objects must not be modified or reordered. These digitally signed or encrypted objects can be recognized easily by their object identification to prevent accidental modification or reordering as described in Section 16.

Signaling message security: CASP integrates security protection. The security mechanisms provide means to protect the different signaling messages in different portions of the network, including first-peer, intra and inter domain. They accomodate environments with varying security and performance requirements.

Flow splitting: Some networks route packets differently depending on their flow labels or DSCP. Operators may want to treat signaling packets differently than the corresponding data packets. These two objectives may conflict since it may cause signaling packets to diverge from the data path.

Because CASP is split into a signaling protocol and a discovery mechanism, CASP only needs to label the scout (discovery) packets in the same manner as the data packets, but can assign labels to signaling packets based on the handling needed for them.

Topology hiding: Even in record-routing mode (Section 6), nodes can hide the addresses of nodes already visited by the message. A more detailed description of network topology hiding is given in 16.5.

Light-weight: CASP is light-weight in terms of implementation complexity and message forwarding overhead. CASP is designed so that a forwarding implementation can be implemented with minimal effort, consisting of a socket listening for TCP connections, a next-peer lookup table and a state timer. In some cases, the messaging layer can be implemented in user space by non-privileged (non-“root”) processes. Depending on the details of the traffic operation needed, the client layer may, however, require access to protected resources.

The proposed security mechanisms try to reuse existing protocols to the extent possible.

> CASP has not been assigned a port yet; we assume that the port number will be above 512. Also, the M layer may require access to kernel data structures to determine the current network address, particularly for mobile hosts.

Due to the re-use of transport connections, session setup latency is, on average, low. Once the authentication and key exchange protocol is finished signaling messages at the M layer can be protected efficiently.

Mobility transparent: CASP interfaces with route change detection mechanisms; IP mobility is also treated as a route change case.

CASP attempts to satisfy the NSIS requirements [4] and framework [5]. However, it also adds additional functionality, such as support for source-specific multicast (SSM) [11] and multiple discovery modes including edge-to-edge and AS routes.

2 Terminology

In this document, the key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “NOT RECOMMENDED”, “MAY”, and “OPTIONAL” are to be interpreted as described in RFC 2119 [12] and indicate requirement levels for compliant CASP implementations.

3 Definitions

CASP node: Application that supports at least the CASP message layer.

CASP chain: The collection of CASP nodes traversed by a CASP message from originator to destination.

CASP session: The set of CASP messages that refer to the same state record, along a path of CASP nodes. All messages with the same session identifier belong to the same CASP session, regardless of the direction of travel.

Downstream: Downstream refers to the direction of the data flow associated with the CASP session.

Originator: The CASP node that sends the CASP message.

State record: State (data) that is managed by CASP, located on each CASP node. There is both CASP state and client state.

4 Message Delivery

Messages within a CASP session can be generated by the end points or by any intermediate point. An intermediate CASP node can sent a message when triggered by internal state changes, such as a routing change, or a timer expiration. In either case, messages traverse the remainder of the CASP nodes, unless they exhaust their node counter or reach the target IP address.

At the messaging layer, CASP is not a request-response protocol, but rather a messaging protocol, i.e., it delivers messages along a path. Client layer applications can, however, send back responses that allow the originator to confirm that the initial message was delivered and to determine whether the operation was successful or encountered an error. This offers end-to-end reliability. The response message uses the existing transport associations in the reverse direction. At the message layer, such responses look the same as requests.

CASP messages using a reliable transport protocol are not constrained in length (except by the CASP length field). They make use of the reliable delivery services offered by the transport layer. CASP scout messages are restricted to the path MTU.

Soft-state refresh is performed by each node, using the algorithm described in Section 11. Refresh intervals are randomized around a nominal value provided by the originator.

5 Transport Protocol Usage

For regular (non-scout) messages, CASP requires reliable, sequenced message delivery with flow and congestion control and can use any transport protocol that supports such a service. (Sequencing is only required for messages within a single CASP session.) Currently, SCTP [8] and TCP provide such services.

> If we had used a new protocol running directly on top of IP or UDP, we would have had to add much of the same functionality, essentially replicating a full-fledged transport protocol. We believe it is not safe to assume that all signaling messages are small and infrequent. Re-use of connections allows improved round-trip time estimation and amortizes the cost of establishing the connection and security association over many CASP sessions.

There generally is only one lower-layer CASP transport connection between any two CASP nodes, regardless of the message and client layer. A CASP node MAY maintain a transport association with another node even if there is no current CASP session. The holding time of these transport connections is an implementation choice.

A CASP node discovers the next CASP node for message delivery using any of the methods in Section 9 and then checks if there is an existing transport connection between these two nodes. If so, it sends the CASP message on that transport connection. If not, it establishes a new connection. It is assumed that transport connections are bidirectional, so that response messages can reuse existing transport connections. However, a response message may also establish a new connection if there is no existing one (e.g., after a reboot of the node to be contacted).

CASP scout messages can, by their nature, not use an existing transport connection. They are transmitted by the origin towards the network-layer destination, marked with the IP router alert option [13, 14]. The originator of a scout message retransmits the message with exponentially increasing time intervals until a response is received, a maximum number of retransmission attempts is reached or an ICMP error message indicates that the destination host or network is unreachable. For each destination and client layer, there can only be one scout message outstanding.

Note that a scout message typically does not reach the IP destination address contained in the IP header.

6 Message Forwarding

CASP messages can be routed either statefully or statelessly. This generality avoids the strict request-response mechanisms found in other protocols. A message has two routing and state fields that determine forwarding behavior:

Destination: The destination flag can have five values: “address” (A), “address+record’ (AR), “route” (R), “state forward” (SF) and “state backward” (SB). In “address” mode, each node determines the next CASP node within the chain by looking at the destination address object. In “address+route” mode, it also records the local IP address, transport mechanism and port for future CASP messages, using RecordRoute objects. In “route” mode, the node uses the embedded Route object to find the next location. In “state forward” mode, the node forwards the message in the direction of the initial message that established the CASP session. In “state backward” mode, the message is sent towards the previous CASP node.

For the “state forward” and “state backward” modes, nodes fall back to “address” mode if there is no state record. This can occur after route changes.

State: The state flag can indicate three operations: “no-op” (NOOP), “add” (ADD), “delete” (DEL). The operations manipulate message layer state. With “no-op”, the node does not establish any state. ADD establishes state and DEL deletes the state record. All messages within a CASP session should use the ADD operation. Deleting the state record deletes all client states. The NOOP message is meant for messages that should not establish state.

In SF and SB modes, the CASP node only inspects the CASP session identifier and then routes the message according to the stored next-node or previous-node information.

Error responses typically contain the “delete” flag and one or more error description objects (see below). They can be routed in “state backward” mode.

The origination of CASP messages does not imply that the client protocol has to follow the same direction. For example, in a receiver-oriented QoS protocol, the recipient of the CASP request, sent in “SF/ADD” mode, would return a “SB/ADD” message containing the reservation data.

7 Message Format

Unlike most other protocols, CASP does not have requests and responses. Rather, it is based on messages that can make use of state established or paths discovered earlier, traversing such paths in either the original sequence of nodes or the reverse.

The CASP message and its components are type-length-value objects, making it possible to use the messages directly with stream-based protocols such as TCP, without having to add an encapsulation layer. The client protocol is encapsulated in one such object, so that no IPv6-style ”next protocol” identifier is needed in the common message part.

The objects within a CASP message can appear in any order, except that the first object is the message identifier and the final object in each CASP message is the CASP client payload. The order of objects has no semantic signficance. There can be at most one client payload in each message; the client payload is optional. (For example, a CASP node discovery or ”traceroute” message may not need a client payload.)

CASP nodes can insert and modify certain objects. The design of objects should separate data that is modifiable from end-to-end constant data, to simplify object signing. CASP nodes do not reorder objects; new objects are added at the end of the message.

> Restricting CASP messages to one client layer message simplifies error reporting and reduces the number of failure scenarios.

Destination flag: Governs the determination of the next node. See Section 6.

State flag: Governs state establishment and teardown, with the values ADD, DEL and NOOP. See Section 6.

Session identifier: The session identifier describes a CASP session, a set of CASP messages that belong together and refer to the same state. It allows subsequent messages that belong together to add, modify or delete existing state information.

The special value of zero indicates that this message neither refers to existing state nor establishes state. The session identifier is a random number with 128-bits in length. The length of this value is motivated to prevent collisions since it has to be globally unique for a given path. Thus, messages that have different CASP origin and destination addresses still belong to the same CASP session if they share a session identifier. Security issues related with this session identifier are described in Section 11 and in 16. As motivated in Section 11 it is important to have some sort of identifier which is not based on a network layer address or a combination of it.

> A globally unique session identifier that is independent of the origin and destination addresses makes it easy for nodes in the middle of the CASP chain to generate messages that only traverse part of the chain.

Flow identification: The flow identification (or descriptor) contains fields that assist in the identification of traffic (data packets) which receive treatment by the client-layer protocol. The client layer determines what action is taken for packets matching the flow identification. In case of a QoS reservation client layer the flow identification determines which data packets experience preferential treatment. It therefore maps individual incoming packets to a given QoS class. Depending on the given flow identification values the effect could be a per-flow treatment or a more broad selection of data traffic. Currently, the following fields should be supported: source, destination IP address, port numbers, transport protocol, flow label (for example described in [15]), destination address options (such as the home address), the SPI (which is motivated in [16]), DiffServ code point (DSCP) and VPN tunnel identifiers.

A CASP message can contain multiple flow descriptors which might be useful in case of SCTP or for specifying individual flow identifiers which cannot be combined into a single one (e.g. traffic of several non-continuous port regions). A flow identifier should allow ranges of ports to be specified as described in Section 7.13.1 of [17]. The usage of flow identifier stacking was considered but requires further investigations. An application for TSEL stacking would allow reservations for tunnels where the ingress node adds a new flow identifier to a stack knowing that the new traffic selector is useful only for a particular region. The original traffic selector is restored at the egress node. IPv4-to-IPv6 translation is an example of such a flow selector stacking.

> The flow identifier is included in the common part of a CASP message so that NATs and firewalls can inspect and possibly modify this data.

Flow identifiers can change mid-session and mid-chain.

Message sequence number: A 32-bit integer that uniquely identifies the message. (Retransmissions are not seen at the message layer.) Each origin issues messages with sequence numbers that increase by one for each message. Message sequence numbers are assigned consecutively by each origin within a CASP session. Sequence numbers are not reset if the transport connection is re-established in mid-session.

> Sequence numbers are relative to each origin so that intermediate nodes can issue messages without conflicting with the sequence numbers chosen by other nodes in the chain.

Origin: The network source address corresponding to the origin of the message; an IPv4 or IPv6 address. Each network source address has its own sequence number space. The network source address does not imply that the data traffic has this origin or destination.

Target : The target field indicates the destination IP address to which data packets are later sent. It is used to route the signaling messages along the same path as used by later data packets.

Message destination: This field contains either the destination address of the message or a scope flag. The address indicates how far the message should travel; if it reaches the destination named, the CASP chain ends. The address MAY, but does not have to, correspond to the target address. It will be different if the message traverse only part of the path for the CASP session. The message destination address does not have to correspond to the address contained in the flow identifier. For example, for proxied CASP session, they will differ.

The scope flag indicates that the CASP chain should terminate at the boundary of the (administrative) scope [18].

> For simplicity, only a single scope is supported, rather than, say, nested scopes. This reflects the typical intra/interdomain division or the division between a local network and ”the Internet”.

NodesLeft: To prevent message loops, the NodesLeft counter is decremented by each CASP node and the message is not forwarded if the counter reaches zero. A suitable “node count exceeded” message is returned to the originator if the counter reaches zero.

> This also allows messages to only traverse parts of a chain of CASP nodes. Other mechanisms are used to restrict the forwarding of CASP messages by scope or address.

Lifetime: If this message establishes state, the Lifetime field determines how long the message soft state is to be kept by each node if there is no additional CASP message. The life time is established in an ADD message and can be updated with ADD messages. (NOOP messages do not refer to state and thus there is no lifetime.) Any client state expires with the message state, but client lifetime can be smaller than the message layer session lifetime. State is refreshed node-by-node and may differ at each node. Thus, this value can be adjusted along the CASP chain.

Dead branch removal flag: Governs whether to remove the CASP states in the detected dead route. See Section 11.

Branch identifier: The branch identifier is a randomly chosen integer that identifies a particular branch. See Section 11 for details.

There is no message type, since each client protocol, including the scout protocol, identifies its own message types in the client object.

In addition, there are a number of optional objects:

Record route: The Record route object is filled with the network addresses of CASP nodes that this message has visited.

Route: The Route object enumerates the addresses of nodes that the message should visit, along with a pointer that indicates the next node. [TBD: Addresses could be simply removed by the visited nodes, simplifying network topology hiding, but making error diagnosis harder.]

8 Capability Negotiation

The CASP capability discovery model relies on named capabilities. Capabilities are named by 16-bit unsigned integers. The value 0 is reserved and not used for any capability. Client protocols are registered as capability values. Client protocols need to negotiate their own capabilities, possibly using the same mechanism and data structures.

The originator can discover capabilities by including a CapDiscovery object in the request. The object has a list of capabilities and counters. Each node that supports a capability increments the counter for that capability. In addition, an overall node count allows to estimate the fraction of nodes supporting a particular feature.

If more detail is desired, the CapRecord object records the address of each node and its list of capabilities.

A CapRequired object enumerates the capabilities that are required. These capabilities are used in the discovery phase. A scout message will traverse nodes that do not meet these capabilities and be reflected back to its source by the first node that can satisfy all requirements. If there are multiple CapRequired objects, it is sufficient if the node satisfies the conditions in one of them.

> Unlike other protocols, CASP does not label individual objects as being mandatory-to-understand or optional. Instead, it identifies certain behaviors that may well rely on a set of objects. Each behavior needs certain types of objects and ignores all others. This makes it easy to define behaviors that require one of N objects.

9 Next-Node Discovery

There are two basic types of CASP nodes, depending on how close they are to the data path. In next-in-path (Section 9.1), each CASP node attempts to discover the next router along the data path that is CASP-aware. In next-AS mode (Section 9.2), there is one (logical) CASP server for each autonomous system and data packets and CASP requests visit the same AS, but not necessarily the same routers. Other paths, such as scopes [19], are possible, but harder to define as a sequence and will not be considered here. (Scopes are, however, important to limit the propagation of CASP messages.)

9.1 Next-in-Path Service

The problem of disccovering the next-in-path CASP node can be divided into an intra-domain and inter-domain component. The intra-domain problem can be split into two parts, namely discovering all CASP nodes within a local domain and determining which of these is visited next on the data path. Determining the next node in an adjacent domain (inter-domain) is more difficult. It would greatly simplify the problem if all border routers are CASP-aware. The scout protocol (see below) can locate the next node both within and beyond the local domain.

For discovering CASP-aware nodes within a domain, a number of methods can be envisioned:

Enhanced routing protocols: It may be possible to extend routing protocols to distribute information about CASP-capable routers to the local routing domain. For example, OSPF [20] could indicate this capability via an Options bit in the common LSA header or a new LSA. A new LSA is needed if capabilities are to be advertised. A CASP node then computes the route based on the CASP request destination address and determines the next CASP-aware node.

Routing protocol with probing: Since the identity of CASP-aware nodes is unlikely to change quickly, a CASP node can attempt to contact routers along the path of the request and cache both positive and negative results. Thus, each CASP node will build up a list of the CASP-capabilities of the local domain and can then determine the next CASP node as above.

Service discovery: Using standard service discovery mechanisms such as SLP [21], CASP nodes can find out about local CASP nodes and their capabilities.

First node: By adding an option to router advertisements [22], local nodes can discover the first CASP node in their path.

DHCP: If there is a single CASP node in a local network, DHCP [23] can advertise this node.

For inter-domain discovery, it may be possible to add information to BGP advertisements.

For next-in-path service, the node wishing to send a CASP message performs the following steps:

1. Determine address N of next CASP node for the destination IP address, using routing table inspection or a scout message.
2. If there already is a transport association with N, send message on that transport association. Done.
3. If not, send a scout message towards the network-layer destination. The first CASP node capable of handling the message responds and includes its IP address in the response. The origin checks whether there is an existing transport association and proceeds as before.

The mechanism above works well if the next router in the data path is CASP-capable, as the number of such routers is likely to be modest. If that is not the case, the origin has to send an exploration packet for each new signaling session, since a node cannot generally determine the next CASP node by inspecting the destination address.

9.2 Next AS Service

CASP messages can be routed so that they “touch down” once per autonomous system (AS), e.g., for a bandwidth-broker service. In this model, each AS has a logical server, possibly consisting of many physical servers, that provide service for a particular CASP client protocol.

One mechanism to find an instance of this server is to create a new, per AS, DNS namespace, such as `ASN.as.arpa`, where ASN is the AS number. DNS NAPTR [24] queries are then used to determine a suitable server for the AS, using the services "CASP+D2X" and "CASPS+D2X", where X is a letter that corresponds to the the transport protocol This specification defines D2U for UDP, D2T for TCP, and D2S for SCTP. Thus, for example, the NAPTR [24] record `17.as.arpa` would identify the CASP service in AS 17. Each NAPTR record in turn points to an SRV record, for coarse-grained redundancy and load balancing. This approach has the advantage that an AS can designate different server clusters for different CASP services. Also, it facilitates discovery.

This approach is only necessary if the BGP peers for a domain do not speak CASP. Otherwise, they can route and process CASP messages as needed.

10 Scout Protocol

The scout protocol is a specialized client protocol for CASP, having a relationship to the main protocol somewhat similar to control protocols like ICMP, IGMP and RTCP [25]. It is used to discover the next suitable CASP node and the required soft-state refresh interval. Scout messages are only required if the next CASP node is more than one network-layer hop away (Section 9) and if there is no other suitable means of discovering the next CASP node. (Other mechanisms are preferred if available since they incur lower overhead and delay.) Each CASP node that needs to discover the next node "triggers" a scout message that generates a response indicating the next node.

Scout messages use the cap_required capability negotiation mechanism to find a suitable node (Section 8).

Scout messages also return the session lifetime desired by the next node.

Scout messages are UDP packets containing a subset of the CASP message layer, a small client layer and the IP router alert option [13, 14]. There are scout requests and responses that follow the usual UDP request-response pattern of reversing source and destination address and ports. Scout requests have an IP destination address set to the target address of the triggering CASP request. Each CASP node that needs to determine the next node issues a scout request. An omnivorous CASP node always returns a scout response message. A selective CASP node checks if it supports the client protocol and other features named in the scout message. If so, it responds with a scout response message addressed to the packet's source address and port. The scout response contains an address record that describes how the CASP node can be reached, i.e., its IP address, the protocols supported and their ports. If not, the scout message is forwarded like a normal UDP/IP packet. The target node always turns around the scout message.

TBD: It may be possible to use ICMP instead of UDP, with a new ICMP message type.

Scout messages have their own reliability mechanism. They are retransmitted periodically, with exponentially increasing retransmission interval, starting at 500 ms.

Scout messages are strictly limited in size to one MTU. They are kept small by only including the common header and a nonce. Requests contain one 64-bit cryptographically random nonce; responses echo that nonce and include an additional random nonce. Responses contain the node's desired lifetime and capability vector, including the security capabilities. Since they do not establish sessions, the sequence number fields in scout messages is zero.

Scout messages can estimate the number of non-CASP IP nodes between two nodes by comparing the original IP TTL value to the one received by the next node. The original TTL value, if known, is included in the scout client message.

Scout messages transport identity and capability information which are security sensitive, as described in Section 16). Also, an attacker can mislead a node into contacting the wrong next CASP node. We define two nonces that protect against these attacks. The first nonce, InitiatorNonce, is in the scout client layer and is echoed by the target node. That prevents attackers that are not privy to the request from impersonating a CASP node. It does not prevent an attacker that can intercept the scout request from returning a bogus response. The ScoutCheck object in the scout response deals with this latter threat. The ScoutCheck allows the next CASP node to detect if it is receiving a request that was preceded by a scout request. The ScoutCheck object contains the InitiatorNonce and a quantity computed by the next CASP node. The method used for computing this quantity is implementation-defined; one possibility is a hash across a secret known to the next node only and the InitiatorNonce.

To reduce the threat of such denial-of-service attacks, CASP nodes SHOULD listen for all scout responses. (TBD: If a CASP node responds to all such answers, this would introduce an amplification attack, but this only occurs for attackers that can intercept messages, not random Internet hosts.)

11 Route Change and Mobility

CASP adapts to route changes and node mobility. It supports fast release of state on the old data paths that is no longer needed. There are two cases:

Observed: The next CASP node is also the next IP router. In that case, the CASP node can observe changes in the local routing table and detect that the next-hop router for a particular set of destinations has changed. If such a change occurs, the node triggers the discovery process (Section 9) to locate the new CASP node.

Refresh: CASP sessions are refreshed periodically end-to-end. Even if there was no change in the routing table, each CASP node has to perform the discovery process for each such message since it has no way of knowing whether the old next next CASP node is still correct.

If a CASP node sends a message to a CASP node that differs from the existing next node for a session, that node becomes a *branch point*.

When a node receives a CASP message for an existing session, but with a different IP source address, i.e., a different previous node, it deduces that it is the *merge point* after a route change. The merge point then performs *dead-branch removal* iff the dead-branch removal flag is set, removing CASP state on nodes that are no longer on the data path. The merge point, M, creates a DEL message and sends it to the old previous node. The CASP origin is set to M, the CASP destination of the message to the origin of the message that triggered the dead-branch removal.

> Having the merge point perform dead branch removal avoids that state is removed before new state is installed.

We define a branch identifier that is incremented at each node if a node sends a CASP message to a new next node for a given session. Branch identifiers are defined within a session only.

> TBD: Alternatively, the branch node could insert its address, but this works less well for mobile systems or for systems with addresses that are not globally unique.

Mobility is regarded as a special case of route change in CASP processing, i.e., the mobile node (source or destination) is the node which detects the route change.

Figure 2 shows an example of a route change. The old path traverses the CASP nodes N1, N2, N3, and N6, while the new path traverses N4 and N5. N1 is the branch point, N6 the merge point. The notation “B=” refers to the branch identifier. N6 detects that it is a branch point and sends back a DEL message on the old branch, but using the new branch identifier (B=2). N2 and N3 simply delete the old state; N1 recognizes that it generated the new branch and thus can terminate the DEL.

```
                              <-DEL(B=2)-
                         +----+    +----+
                     --- | N2 | -- | N3 | --
                    /    +----+    +----+   \
                   /B=1      old route       \
                  /                           \
+-----+ B=1    +----+                       +----+     +-----+
|  S  | -- > --| N1 |                       | N6 |-----|  D  |
+-----+        +----+                       +----+     +-----+
                  \                           /
                   \B=2                      /
                    \    +----+    +----+   /
                     --- | N4 | -- | N5 | --
                         +----+    +----+
                             new route
                              -ADD->
```

Figure 2: Route Change

12 Multicast Support

CASP supports source-specific multicast (SSM) service model [11], which allows one-to-many group communications.

This can be achieved by a special scout message for SSM multicast. If a node receives or initiates a CASP message intended for an SSM multicast group, it sends a scout message towards the SSM destination. The source and destination of this scout message should be set (e.g., via raw socket) as the SSM source S and the destination G, respectively. Also, the actual address of this CASP node should be included in the scout message so that a CASP node receiving the scout message can reply to the right address. There are two possibilities to do this: either use origin object in scout message, or introduce an optional scout-sender object. This scout message can detect new and old branch(es), thus CASP transport association along the source-specific multicast tree can be created, updated or removed.

> If an SSM multicast routing entry (S, G) is created in an SSM-enabled node, only messages with source address S and destination address G can be forwarded according to this entry.

In case the multicast routing next hop is CASP-aware, it is possible to speed up the process by inspecting the IP multicast routing next hop table as defined in the IP multicast MIB [26, 27].

1. If all the next hop addresses have an associated CASP state, done.

2. If a next hop has not yet CASP transport association (i.e., pruning a new branch), it sends an “SF/ADD” message to the next hop in this branch to create a new CASP transport association.

3. If a next hop was previously in the next hop table, but currently not any more, it sends an “SF/DEL” message toward this next hop to release the the unnecessary CASP transport association due to dynamic receiver membership.

Typically, multicast support replies on the special scout message in addition to similar mechanisms that handle route changes (Section 11).

13 CASP over Tunnels

The authors of [28] identified three types of tunnels. These tunnels are differentiated whether they support QoS reservation and to which degree (per-flow allocated resources within the tunnel or non-flow allocated resources). CASP tunnel operation as described in this document is independent of the C layer operation. The implication of tunnels is therefore that they modify routing and they might require modification to the traffic selector since information like ports and transport protocols are hidden. Unlike RSVP only the scout protocol uses a router alert option. A CASP aware ingress node can therefore decide whether to hide the router alert option. In case that the router alert option is hidden then the egress node is the next discovered CASP peer (assuming the egress node is CASP aware). Otherwise other CASP nodes along the tunneled region can be discovered as well.

CASP can operate over any types of tunnels (for example IPsec, IP-in-IP, IPv4/IPv6) if both ingress node and egress node of a tunnel support CASP. In case that CASP is not supported at these nodes then the CASP messages are hidden inside the tunnel region. The scout messages then do not discover CASP nodes inside the tunneled region because of the IP encapsulation of the router alert option.

Tunnels similar to those used in micro- and macro-mobility schemes where tunneling affects the traffic of a single host only (i.e. where the end-host usually participates in tunnel establishment and termination) are covered by CASP in the following way: A modification to routing (based on the establishment of a tunnel for example because of route optimization in Mobile IP) might require adaptation of the traffic selector along the path (or at parts). The same is true when (possibly nested) IPSec tunnels are used along the path to protect data traffic. Traffic selectors need to reflect the different traffic classification possibilities at various locations along the path. A careful selection of traffic selectors might therefore help not to require adjustments of state established along the path when changes in routing happen.

14 Protocol Heritage

CASP attempts to borrow concepts and ideas that have worked well in other application and network-layer signaling protocols. Examples include:

- The message format and the use of router alert options in scout messages is similar to RSVP [29]. However, CASP differs from RSVP in having a clearer layering and avoids the complexities of implementing components of a transport protocol [30].

  Multicast support is offered for source-specific multicast (SSM) [11], without the complexity of reservation styles [31], receiver diversity and maintaining multiple multicast sink trees in the signaling protocol.

- The notion of separating delivery and application was first explored by CSTP [2].

- The feature negotation approach borrows from RTSP [32] and SIP [33].

- The notion of messaging is also explored in BEEP [34].

15 IANA Considerations

A future version of the document will include IANA considerations for object types, client protocols and port numbers for the CASP protocol.

16 CASP Security

This section addresses various security issues of the CASP protocol with some background information and possible options. Additionally some threat-specific details are explained which are not yet covered or not described in detail in [35]. Many of the protection mechanisms described are based on what was learned when investigating security mechanisms in RSVP as described in [36].

The content of this section is organized as follows. The first paragraph investigates the scout protocol security. The next two paragraphs focus on securing the transport of signaling messages at various places in the network and the session (or reservation) ownership problem. Next a description of the CMS [37] usage for the client-layer is provided. Finally a miscellaneous issues section addresses security features which are somewhat independent of the previous sections and could also be entitled as framework security topics. Some of these security topics are not directly applicable for securing the CASP protocol itself but try to highlight the interaction with other protocols. The CASP protocol is designed not to rule out interactions with some other protocols or deployment within some non-standard architectures.

A summary of this fairly detailed section is given in security considerations section for the impatient reader.

16.1 Scout Messages

Problem-Description: The task of bootstrapping a node with configuration information is an important and security relevant task. CASP relies on a number of mechanisms for discovering the next CASP-aware peer. As described in Section 9 additionally to learning the identity of the next peer some capability information is provided. Every of the proposed protocols for distributing information is therefore vulnerable to similar attacks. This section however is mainly focused on the description of threats and the security of scout messages since the mechanisms is different to what is used for learning (or distributing) configuration information in general. A description of the vulnerabilities created by using other protocols like DHCP, Router Advertisements, etc. might be included in a future version of this document.

The main purpose of scout messages is to discover the next CASP-aware network element with a certain capability along the path. Since a scout message uses the router alert option mechanism it follows the data path by using the destination address of the IP packet as the endpoint destination address. As such it is not helpful to include a keyed message digest (or a similar cryptographic protection based on symmetric keys) already to the outgoing message since the identity of the receiving node is unknown. Using public key based cryptography (for example a digital signature) to protect the outgoing scout message could be used by an adversary to mount denial of service attacks against CASP-aware nodes. Nothing prevents an adversary from transmitting millions of bogus scout messages to a CASP node and to force heavy cryptographic processing for verification. Note that it would be possible to include a digital signature to the response to provide identity information of the responder in a cryptographic protected manner. Using such a mechanism could however easily be used as a denial of service attack.

Scout messages are continuously transmitted from the initiator towards the destination address to react on route changes (if no other configuration mechanism is used). Protecting scout messages which are sent towards the destination address knowing that they will hit a known CASP peer security protection is possible. However such a protection is not particularly useful since the scout reply messages of interest are those that indicate a path change. These scout messages hit a new CASP-aware router which returns among some other information its identity as described in Section 9 and in 4.

Threat 1: Downgrading Attack: Security relevant information included in the scout message for example is the identity of the next CASP-aware network device, supported security mechanisms (namely IPSec, TLS and EAP as described in this document) and other capabilities. In case of IPSec various key mangement protocols can be supported and also have to be indicated. Downgrading attacks are therefore easily possible when no protected negotiation takes place.

Threat 2: Man-in-the-Middle: An adversary might want to inject a bogus reply message forcing the scout message initiator to start a transport layer connection and the corresponding security association establishment. Figure 3 describes the attack in more detail.

This attack assumes that the adversary is able to eavesdrop the initial scout message sent by the scout message initiator, for example by a mobile node. A MITM-attack does not require that the first-hop router is CASP aware. There is no particular restriction to the placement of CASP-aware nodes within the network. Furthermore we assume that the scout reply message by the adversary returns to the scout message initiator faster than the real response. This represents some race condition characteristics if the next CASP aware node is very close (in IP-hop terms) to the initiator.

As shown in message step (2) in Fig. 3 the adversary returns a scout reply message with its own IP address as the next CASP aware node along the path. Without any additional information the scout message initiator has to trust this information. Then a transport layer connection is established with IPx (i.e. with the adversary) in step (3). The adversary then establishes a transport layer connection with the ”real” next CASP aware node (in Fig. 3 with the Access Router).

As a variant of this attack an adversary not able to eavesdrop transmitted scout requests could flood a node with bogus scout reply messages. In the scout message sender accidentally accepts one of those bogus messages then a MITM-attack as described in figure 3 is possible.

Threat 3: Bogus Reply: An other threat which would disturb the protocol behavior and would serve as a sort of denial of service attack is the following: An adversary might return a bogus scout reply message indicating a different identity which is still legimate CASP node (but not the first CASP node). The scout message initiator would then establish a transport layer connection with the wrong node. One of the possible consequences is that data traffic might not experience proper QoS treatment (in case that a QoS client layer is used) since signaling message establish state not along the data traffic. It would be very difficult for both nodes to discover this type of attack without any precautions.

An other variant of this attack is the following. An adversary returns a bogus Scout-reply message to convince a CASP node to establish a new connection with a different CASP node although the previous connection is still valid.

Proposed Security Protection: Providing iron-clad security protection for scout messages is difficult. Since they provide information to the initiator of the scout message to which node to start the establishment of a transport layer and security association establishment some threats are possible.

- As part of the security association setup process at the M layer authentication and authorization has to be provided. Authentication and authorization prevent identity spoofing and MITM attacks.
- To prevent downgrading attacks the information exchanged at the scout protocol is repeated later at the messaging layer to verify the exchanged information. Since signaling messages have to be protected this mechanism allows to detect a downgrading attack. It assumes that the downgraded security mechanisms (because modified by an adversary) provides the necessary security for detecting the modification. An adversary would therefore have to break the chosen security mechanism in realtime in order not to be discovered. This approach is somewhat similar to the protocol exchange provided in [38].

  Downgrading to no security protection must not be possible since various attacks could be mounted against the CASP signaling protocol even if separate protection at the client layer via CMS is provided.
- To prevent bogus replies and MITM attacks in addition to authentication and authorization two cookie values are used (Cookie(i) and Cookie(r)).
  - A scout request message contains a Cookie value (Cookie(i)) which is a 64-bit random number to match replies with requests. Including Cookie(i) also in the reply message prevents adversaries from accidentally accepting a bogus scout reply message from an adversary.
  - A scout response message contains the second Cookie value (Cookie(r)) with the same length which servers functionality similar as used in Mobile IPv6 [39] (for securing the binding update between the mobile node and the corresponding node). A Cookie field with variable length would be required when the Cookie contains encrypted fields (such as Cookie(i)) instead of a keyed message digest algorithm. In any case no per-session state should be stored at the scout message responder when receiving the initial scout message. In order to prevent MITM and bogus replay type of attacks the exchanged information (or some parts of it) might be included in the computation of a keyed hash or encrypted with a locally known key.
  - Both cookie values are included later in the protected signaling message exchange. When receiving the Cookie value a verification at the responder is possible.
    If an adversary injects bogus reply messages then a verification step would immediately fail since the new CASP responder would detect the attack when verifying Cookie(r).

  Since the Cookie(r) value has to be verified only locally its structure is mostly implementation specific. One suggestion for creating this cryptographic cookie would be to use $Cookie(r) = Encrypt(local\_key, Cookie(i))$ in case of encryption or $Cookie(r) = HMAC-MD5(local\_key, ScoutRequ$ in case of a keyed integrity algorithm. Note that the values used inside the cookie have to be repeated later to support the verification. The time interval for changing the local host key ($local\_key$) is policy dependent.
- To prevent the variant of the bogus reply attack whereby an adversary wants the scout message sender to create a new transport layer connection to tear down the old but still valid connection the following counter-measure is necessary. The establishment of a new connection (in replacement to an existing one) must be successful before a tear down of the previous connection takes place.

16.2 Securing the Transport and Messaging Layers

The security protection of signaling messages at the messaging layer can be classified into authentication, integrity and replay protection. Providing proper data origin authentication, integrity and replay protection is required as motivated in [4] and in [35]. As a difference to the security provided in RSVP [40] the support for authentication and key establishment protocols should be integrated at the early beginning of the protocol.

TCP and SCTP are the main protocols for exchanging signaling message content. It is therefore useful to reuse existing security protocols to protect the integrity of signaling messages. In case of TCP and SCTP a good choice is TLS providing both session key establishment based on unilateral and optionally mutual public key based authentication as described in [6]. Additionally support for Kerberos as described in [41] is available although rarely used. This is primarily because of the dominance of public key based server-to-client authentication in the web environment. As an alternative IPSec [42, 43] can be used to secure CASP signaling messages (not including the scout messages) at the network layer. IPSec allows a separation between the key exchange protocol and the actual protection of data packets. IPSec AH and IPSec ESP provide protection of IP packets whereas various key exchange protocols may be used to establish the required IPSec SAs. IKE [44] is the default key management but also KINK [45] and in the near future SON-of-IKE ([17], [46]) can be used. These authentication and key exchange protocols allow some room for adaptation to particular environments with different trust relationships.

Establishing security associations for protecting signaling messages at different parts of the network is however difficult. Hence the following subsections describe the security issues at each part of the network.

First-Peer Communication: First-peer communication refers to the communication between the originator of the signaling message and the edge router in the attached network. In most mobility scenarios the signaling messages are initiated by (or terminate at) the mobile node. The signaling messages when entering the visiting network (i.e. access network) are intercepted most likely at the edge.

The protection of first-peer communication is probably the most difficult for a number of reasons. First there is lack of trust between the initiator and the first network. This requires special care for authentication and more difficulties in session key establishment. Next there is a need to support a larger number of authentication and session key establishment protocols.

Different usage scenarios for CASP require support for a particular authentication mechanism and a smooth integration into the existing infrastructure. Furthermore there are both performance limitations that exclude one or the other authentication and session key establishment protocol.

In the following text the advantages and disadvantages of using TLS and IPSec for protecting signaling messages are described.

When considering possible difficulties listed in the above paragraph then the problem of securing the signaling messages can be separated into two parts. The first part is the actual protection of the messages similar to what is known from the TLS Record Layer or from IPSec ESP/AH. The protocols either add a keyed message digest or encrypt the message and add replay protection (based on sequence numbers) to it to protect the messages from eavesdropping (only in case of confidentiality protection) and from modification.

The TLS Handshake Layer provides session key establishment, unilateral and optionally mutual authentication. Although TLS offers client to server authentication as an option we suggest running EAP [47] on top of TLS whereby the EAP exchange is protected by TLS to support client to visited network authentication. Note that the approach taken to secure CASP is different to the approaches suggested to protect the EAP message exchange with TLS (see [48] and [49]). The main motivation for choosing this approach is to make use of an existing protection mechanisms as provided by the TLS Record Layer. Issues regarding session resuming as described in Section 4.2 of [49] are however applicable also in this context. Additionally the usage of [50] for including OCSP [51] protocol information within the TLS protocol for the purpose of certificate validation might also be useful in this context as described in Section 4.3 of [49]. The decision to protect EAP with TLS was therefore not done for protection of EAP message payloads or for providing client identity protection in mobile environments. Nearly all EAP mechanisms do not require confidentiality protection of the EAP message exchange itself. Even though some EAP mechanisms allow the distribution of a session key there is currently no support for a protection mechanism at the transport layer similar to the modular IPSec approach where the key management is separated from the actual data protection mechanisms. Using the fresh session key distributed with AAA (based on some EAP methods) could for example be used to trigger IKE with pre-shared secret authentication mode. Using the EAP distributed keys immediately (without running IKE) for creating IPSec SAs is not possible since information of IKE Phase II (IPSec SA negotiation) is missing.

When using TLS we believe that a non-public key based client-to-visited network authentication mechanism is required since not every client is supposed to support client certificates. Using mutual public key based authentication requires a widely deployed PKI. We think that a requirement for a global PKI would put a large burden on the deployment of a protocol.

An additional motivation for supporting EAP to support authentication from the client-to-visited network is that a previously established session key can be reused and local authentication to the local AAA server is executed. In future versions of this document message flows and suggestions for EAP authentication methods will be given.

Support for IPSec-based protection with IKE or similar to establish an IPSec SA to protect signaling messages between the CASP peers requires interaction between the CASP implementation and the key management daemon. First there needs to be an interface to trigger the dynamic creation and modification of IPSec security associations (such as provided by PF_KEY [52]). Since protection is necessary for the CASP signaling messages only the IPSec security policy database should be able to install traffic selectors with a granularity at protocol type (TCP, SCTP) and specific port numbers. Additionally there is a need to fetch the credentials used at the key management protocol for the purpose of policy based admission control and accounting. Using TLS these tasks are easy since these APIs are available and widely used for example comparing the identity used in a certificate and the URL at a TLS-supporting web browser. We believe that the protection of the transport layer and messaging layer and a separate protection at the client layer for the purpose of ”identity representation” [53] as done in RSVP might not be required in most scenarios. This is especially true if both mechanisms terminate at the same endpoint.

Intra-Domain Communication: Protecting signaling messages within a single administrative domain is simpler because of the strong trust assumptions, simplified key management and rare network topology changes. Both TLS and IPSec would satisfy the requirements for protecting signaling messages in on a peer-to-peer basis. For key management both pre-shared secret (IKE, main or aggressive mode with pre-shared secret authentication), Kerberos (KINK [45], Kerberos support for TLS [41]) and public key based key management (IKE, SON-of-IKE, TLS) are available. The variety of key exchange protocols gives administrators a large degree of freedom.

Inter-Domain Communication: For inter-domain communication again TLS and IPSec can be used. The only difference is the more difficult key management which might demand a public key infrastructure used between the network operators.

End-to-End Security: At the messaging layer no end-to-end security (encryption or integrity) of the signaling messages is provided. This is primarily due to the fact that intermediate CASP-aware nodes have to read or modify certain parts of the message. Message objects that need not to be read or modified by nodes along the path direct end-to-end communication is likely to be more efficient and appropriate. If a specific protocol usage requires that some objects have to be protected in an end-to-end fashion then the following two approaches are possible:

- The required objects could be exchanged between the desired parties (i.e. end-to-end) without using CASP at all.
- If the objects in question have relevance for a particular client-layer (and therefore for some other objects along the path) then CMS protection and encapsulation of objects might be the best choice.

16.3 Session Ownership

By allowing a node to refer to a previously established session state an identifier is required. This session identifier is independent of a traffic selector. Apart from different options for generating such an identifier the question arises how a node proves whether it is owner of a session state or not. In the following section first a problem description is given, then different solutions approaches are discussed and finally a solution is proposed for the CASP protocol.

Problem Description: The CASP protocol is used to establish distributed state along a path in the network. The number of nodes storing state might be be larger (for example 30) and could also vary from time to time (for example in case of route changes). The session identifier is used to point to a particular stored state at each router. If an adversary is able to obtain the session identifier (for example by eavesdropping) then he might attach himself to any node along the path to modify existing state information. Only the participating entities end-hosts and routers should be able to trigger session state modifications. Routers must be able to react on route changes to execute a local repair mechanism. Hence a protection mechanism should only protect against nodes which are never intended to participate in the signaling message exchange.

Some of the described problems are less problematic in non-mobile environments since the first CASP-aware router (for example the edge or access router) could easily associate authentication state with the reservation identifier. Hence ownership verification is possible. If we assume a mobility scenario then the movement of a node makes this verification step much more difficult since each CASP-aware node along the path could possibly forced to do this verification.

Figure 4 depicts the session ownership problem for a mobility case:

Figure 4 shows that the mobile node establishes state at CASP nodes along the path. As a result the Access Router1 (AR1) (and other nodes along the path - Router1 (R1), Router3, Router4, etc.) store a session state including the session identifier (SIDx). As assumed only the mobile node and the AR1 (prior to the movement of the mobile node) share a security association. Signaling message communication between the participating entities is secured based on the peer-to-peer principle. In our example this means that AR1 has a security association with router R1 and secures signaling message. The session identity is included in the signaling messages and forwarded along the path as it serves as a reference to the established state. Without end-host mobility an existing session state is simply modified by transmitting a properly protected signaling message from the mobile node to the AR1 by additionally including the session identifier. Verification can be done at AR1 locally.

An adversary who was able to obtain the session identifier of the mobile node includes the same identifier (SIDx) in a new signaling message. The message causes state to be installed at Access Router2 (AR2) and subsequently at Router2. Finally the signaling message hits Router3 where an existing state is noticed. Router3 has to assume a route change took place. The adversary included a the ”dead branch removal” flag in the signaling message the path between the mobile node and Route3 is removed.

Router3 is unable to decide whether the new signaling message was sent from an adversary or from the mobile node (i.e. the real owner) since there is no additional information that aids in verification.

Some approaches to address the problem: Some possible means for verification and proofing ownership of a session are given below. These examples should give the reader background information what issues have been considered during the protocol design phase. Verification can be done by

1. using a cryptographic reservation identifier (for example by using a reverse hash-chain or by transmitting a digitally signed reservation whereby the reservation identifier is the hash of the public key similar to [54]. This scheme would only allow the initiator of the signaling message to trigger a modification. Special handling would be required in cases where route changes happen within the network (such as a local repair). Only the entity possessing the key pair is able to initiate authorized signaling messages. From the described protocol behavior this is not desired.
2. being able to authenticate a signaling message at each router along the path. It is then possible to associate a particular identity with a reservation identifier. The reservation identifier may even serve as a pointer to a security association. Without knowing the correct key belonging to the reservation identifier no updates are allowed.
3. using an authorization token based approach. With this approach the network would return an opaque token (opaque for the mobile node) to the MN with the initial signaling message. The token would then be included in a later signaling protocol message after a handover.
4. relying on Context Transfer which allows reservation state to be forwarded from one access router to an other. Using such a scheme the verification of the reservation identifier can be done at the new access router immediately. Unfortunately Context Transfer is applicable only within a single administrative domain. Reservation merging is however possible at any location along the path.
5. verifying a reservation request by help of a centralized entity within an administrative domain (for example with the help of the PDP). However this approach is again only applicable at a single administrative domain.
6. distributing a session key along the signaling path. A new signaling message then contains a cryptographic object which allows the verification at an arbitrary CASP-aware node along the path where signaling state was established.

Approach taken by CASP: Providing confidentiality protection to protect the reservation identifier makes it more very difficult for an adversary to eavesdrop the session identifier and to reuse it for a subsequent attack. The required authentication of signaling message originator Since authentication is required an adversary can furthermore not hide its identity when starting an attack.

TLS without integrity-only suites and IPSec ESP without NULL encryption algorithm [55] provides the desired confidentiality protection.

To summarize we can state that in order to provide proper security for the reservation ownership problem a solution has to face many challenges including performance, state maintenance and replay protection. The above-described problem of authorization is not restricted to communication at the edge as described above. The problem basically occurs anywhere in the network whenever an old path becomes invalid and a reservation along a new path has to be established. The merge point (or cross-over router from the above mobility scenario) has to make sure that only the legitimate owner of the reservation issued this request. The difference between network internal and edge reservation ownership issues are only trust issues.

16.4 Protection of the Client-Layer

In some scenarios the transport-layer hop-by-hop protection (based on TLS or IPSec) might not be sufficient. For those cases we suggest the selective protection of objects at the client-layer by using CMS where such a protection is meaningful. CMS allows objects to be protected with a digital signature and in case that the public key of the recipient is known also encryption. The fact that CMS is known and already used to protect SIP message parts and also applied to selectively protect Diameter AVPs convinced us to include support for such a mechanism. As a difference to Diameter [56] proposed protection where digitally signed AVPs have a P-bit set but are not included in the CMS-protected object the CASP protocol always includes protected objects (both digitally signed and encrypted) inside a CMS object. This avoids conflicts in complex object signing/encrypting scenarios.

Additionally to the capability to protect objects CMS also allows a session key to be established based on a key agreement or a key transport technique. An established session key would allow to speed-up the protection of objects between the same peers in a later point in time.

Since CMS objects are known to be large (because of certificates, CRLs, digital signatures and the large number of optional support objects) messages containing these CMS objects often exceed the maximum MTU size leading to fragmentation. By using a transport layer protocol like TCP and SCTP such a problem is avoided.

It is worth noting that objects of the Client-Layer do not necessarily have to be added by the signaling originating host. Instead for example a PDP might also be able to add a CMS protected authorization token to allow secure delivery of information within a specific administrative region.

As mentioned above the identity of the receiving node is required in case of encryption. If this information is not available beforehand then a discovery phase similar to the one proposed for Diameter [56] has to be executed. Since this discovery/negotiation procedure seems to be useful a more detailed description will be included in a future version of the document.

16.5 Miscellaneous Issues

The following section discusses security issues which are not immediately applicable for the operation of CASP. However they are worth discussing in the context of a framework. Hence this section is meant to provide background information for a CASP deployment.

Authorization: Authorization in a quality of service environment is required as part of the policy-based admission control procedure. What information to include in such a step is however still under discussion. In a corporate network environment information such as group membership and special access rights can be used whereas a very generic mobile environment scenarios only requires the assurance that a particular user is able to pay for the requested resources.

Independent of the question which information to distribute in a particular scenario for a given application (e.g. QoS signaling protocol, NAT/Firewall-traversal protocol, etc.) two possible approaches can be used to deliver the required information to various entities.

The first approach could be labeled ”offline”. The handling is similar to what is offered with Kerberos and Attribute-Certificates. A processing entity is thereby able to retrieve the authorization information directly from the credentials received from the user. Authentication and authorization information are therefore cryptographically bundled together. An example how to distribute authorization information within a Kerberos ticket is described in [57].

The second approach is called ”online”. Using this approach the processing entity would use the identity based on a successful authentication and queries a repository which represents the online characteristic. If the desired information is not cached at the verifying entity then an online request is required. One important requirement for this approach to work is a successful mapping between the identity used for authentication and the entity with which the information at the repository can be obtained. If the information at the repository itself is exchanged based on for example an AAA exchange between the user’s home network and the visiting network then a mismatch between the identities might be common. Hence care must be taken to define an appropriate mapping between different identity formats used by various authentication protocols (for example: NAI, DN and Kerberos principal name and realm).

Note that the usage of authorization information inside a Kerberos ticket or with attribute certificates is currently rarely used in protocols like KINK, TLS or IKE. Hence some further investigation is required.

Trust Establishment: Relying on an authentication and key exchange protocol to mutually authenticate the both endpoints (and to secure subsequent signaling messages) in a wireless environment is often not sufficient. It no additional information is supplied then it often does not provide particular useful information to the signaling message originator that the other endpoint is a router with given IP address (or a FQDN). This is particularly true in a mobile environment where a mobile node attaches to the network for the first time. More important in this case is that either a successful AAA exchange or the service provider’s digitally signed certificate gives assurance that a particular business relationship is present and known. In case of public key based authentication a mobile client would be pre-configured with some CA certificates that allow verification of the host-spot provider’s certificate. In most cases it is unlikely that the end hosts is able to verify the certificate of the access network provider directly.

Non-Repudiation: Non-repudiation of signaling messages is not provided as part of this protocol. However CMS allows Client-Layer objects to be counter-signed where such a procedure is necessary and useful. The main rationality for excluding support at the lower layer in the CASP protocol is based on the question of usefulness and an avoided performance impact. We think that public key cryptography should be used only in cases where security associations have to be established. Much faster cryptographic protection of signaling messages should be based on symmetric techniques to lower computational requirements for nodes participating in the signaling message exchange.

Denial of Service: With the design of the protocol we tried to avoid possible denial of service vulnerabilities to some extend. Some denial of service attacks are described in the context of scout messages. Mechanisms to prevent these attacks is explained in the same paragraph.

When choosing a transport layer protocol for CASP it should be noted that TCP is more vulnerable to denial of service attacks (for example TCP SYN flooding) than SCTP as described in [8] and in [58]. Using IPSec to protect all upper layer protocols prevents this attack.

Especially key exchange protocols tend to be vulnerable against DoS attacks. Hence when using CASP and a specific key exchange protocol it is necessary to consider such a vulernability. Since CASP is to some extend a build block which requires fitting it to a particular architecture this issue needs to be considered. By using TLS the key exchange protocol is built-in. Using IPSec a number of options need to be taken into consideration for each of the protocols like IKE, SON-of-IKE, KINK etc.

Network Topology Hiding: This Section discusses network topology hiding issues for CASP when using a record route message forwarding approach. Similar issues have already been discussed at the SIP working group. The following picture should elaborate the idea in more detail.

As described in figure 5 router R3 serves as a trust boundary which allows the internal routers R1 and R2 of the administrative domain A to be hidden. For router R4, which belongs to a different administrative domain, only the edge router R3 is visible. When a message with a record route object arrives at router R1 then the router adds its identity. The same procedure happens at R2. A policy at router R3 indicates to hide the internal network entities to the outside world. Hence when the signaling message arrives at R3 the identities of router R1 and R2 at the record route object are replaced by an encrypted version. The key used for encryption is locally known only and the rekeying time interval is also policy dependent. The entire encryption process (including algorithms) is an implementation issue and requires not standardization effort. This approach allows R3 still to be stateless. Finally the result of this encryption and the identity of R3 is forwarded to R4. When R4 routes a signaling message back (by visiting the routers as indicated in the record route in the reverse order) then router R3 decrypts the encrypted routing information and forwards the message back to R2 and R1. It should be noted that the encrypted identity information should be labeled as encrypted to avoid misinterpretations.

Using a stateful approach where router R3 simply stores state and indicates only its identity to router R4 the same result is achieved. In this case topology hiding is provided as part of state storing at the individual routers.

For diagnostic and query messages a policy might indicate whether the processing is allowed or not. However in such a case some problems with network hiding arise. In general there is a question whether identity hiding in such a case is worth the reduced functionality of the protocol. Most network administrators consider a diagnosis and discovery capability as a very useful tool in their daily work. The same is true for developers.

For the above-described issues network topology hiding can be supported as part of a local configuration or policy decision. A network administrator should decide whether to support the above-described mechanism by encrypting the identities.

17 Security Considerations

CASP protocol communication is divided into a discovery part (for example using the scout protocol) and a regular message exchange for which a transport layer connection is established. Scout messages allow the discovery of nodes participating in the CASP protocol (if no other mechanisms is used). These messages experience custom security protection. The subsequently established transport layer connection used to transport M (and C) layer messages is either protected by the TLS Record Layer or by IPSec ESP. These two protocols support protection of the CASP protocol messages. In case of TLS the key exchange protocol is built-in whereas several choices are offered for IPSec (for example KINK, IKE and in the near future SON-of-IKE). Signaling messages travel between different parts of the network where different trust assumptions are valid. To reflect this circumstance we suggest that intra-domain and inter-domain signaling message communication should be either protected by TLS or by IPSec depending on the administrator's choice. For mobile environments protection of the communication between the mobile node and the first-peer in the access network might be based on TLS for access network to client authentication and EAP-based authentication for client to server authentication. This decision is motivated by the different requirements for user-to-network authentication and the non-existence of a global PKI.

The above-described security mechanisms operate in a CASP-peer to CASP-peer manner. Some Client-Layer objects however might require additional protection. Digitally signed or encrypted Client-Layer objects can be provided by CMS as successfully used by other protocols like S/MIME, SIP or Diameter. Authorization tokens or information which has to be verified within a domain only can make use of such a selective protection.

Security protection for the session ownership problem is difficult and requires more investigation. This document describes the problem and lists a number of possible approaches to tackle the problem. A first version of the protocol may rely on confidentiality protection (as provided by most cipher-suites for TLS and IPSec ESP without NULL encryption) to avoid eavesdroppers to learn the 128-bit random session identifier. Without knowledge of this session identifier the described attack is difficult. Note that CASP-aware network elements along the CASP chain must know the session identifier in order for the protocol to operate correctly.

18 Open Issues and Future Work

This Section contains some identified future work items:

- The process of discovering CASP-aware nodes is a security sensitive process. Some references and considerations regarding security vulnerabilities of proposed discovery mechanisms for CASP have to be described (in addition to the scout description).

- To avoid users the requirement for mutual public key based authentication in TLS for both first-peer as well as last-peer communication some additional text has to be provided. Although EAP is proposed for first-peer communication to supported non-public key based authentication from a user to the network the roles of client/server are reversed in case last-peer communication.

- Selective client layer object protection is provided by CMS. The process of discover/negotiation needs some further considerations. Additionally the establishment of symmetric keys for efficiently protecting multiple message exchanges have to be considered.

- The usage of EAP is proposed in relationship with client-authentication. Further work is necessary to describe protocol interactions and message flows based on some selected EAP mechanisms.
- Interactions with accounting and charging and their implications for the security framework will be described in more detail in a separate document. Together with this document some basic explanations of the trust relationships should be provided.
- The security section is more detailed than other parts of the document. It might be therefore suitable to distribute a more detailed description of the security issues in a separate security draft.

19 Acknowledgements

We would like to thank (in alphabetical order) Wolfgang Buecker, Jorge Cuellar, Rainer Falk and Dirk Kroeselberg for their comments to this draft. Jorge and Dirk provided us with a large number of comments to improve the security section of this document. Cornel Pampu contributed to the mobility discussions.

A Message Format Details

For concreteness, we describe a strawman packet format below.

All CASP messages are composed of one or more TLV (type-length-value) objects. Within each object, elements are aligned on multiples of their size, to speed processing. All objects have lengths of a multiple of 32 bits. The length field in the object indicates the number of 32-bit words.

We describe messages and objects as pseudo-C structs. Elements are enumerated in transmission order. We use the data types uint8, uint16, uint32, uint64, uint128 to identify unsigned integers with 8, 16, 32, 64 or 128 bits, respectively. We define the following data types:

```
typedef struct {
   uint16 msg_type;
   uint16 msg_length;
} tl;

typedef struct {
  uint16 length;          /* actual string length, in bytes */
  uint8 chars[length];  /* UTF-8 */
} string;
```

Message types not defined in this document are registered with IANA (Section 15).

Text strings are coded in the UTF-8 character set encoding [59]. They are padded to a multiple of 32 bits with zero bytes.

```
typedef struct {
  tl tl = {message type, total length of packet},
  ... objects ...
} casp_message;
```

```
typedef struct {
  tl tl = {payload},
  ... payload bytes ..
} casp_payload;

typedef struct {
  tl tl = {msgid},
  uint64 random;
} msgid;
```

A.1 Network Addresses

```
typedef struct {
   int8 network;
   int8 transport;
   int16 port;
   uint128 address;
} ip6_address;

typedef struct {
  int8 network;
  int8 transport;
  int16 port;
  uint32 address;
} ip4_address;

typedef struct {
  uint8 nonce[20]
} address_cookie;

typedef struct {
  tl tl = {,};
  union {
    ip4_address;
    ip6_address;
  } address[];
} req_route;
```

The address_cookie object contains an encrypted version of an IPv4 or IPv6 address. It can be used for topology hiding. A node that wants to obscure the network addresses in a region encrypts all IPv4 or IPv6 address objects, adding random salt, and then replaces them with the address_cookie. Upon return, the node decrypts the address information and routes the request normally. Thus, the egress node of a protected region is responsible for encrypting and decrypting address objects.

A.2 Capability Discovery

```
typedef struct {
   tl tl;
   uint16 cap[];
} cap_discovery;

typedef struct {
  tl tl;
  uint16 cap[];
} cap_required;
```

B Authors' Addresses

Henning Schulzrinne
Dept. of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
USA
EMail: schulzrinne@cs.columbia.edu

Hannes Tschofenig
Siemens AG
Otto-Hahn-Ring 6
Munich 81739
Germany
EMail: Hannes.Tschofenig@mchp.siemens.de

Xiaoming Fu
Technical University Berlin
Sekr. FT 5-2, Einsteinufer 25
Berlin 10587
Germany
EMail: fu@ee.tu-berlin.de

Jochen Eisl
Siemens AG
Otto-Hahn-Ring 6
81739 Munich
Germany
EMail: Jochen.Eisl@icn.siemens.de

Robert Hancock
Roke Manor Research
Old Salisbury Lane

Romsey, Hampshire
UK
EMail: robert.hancock@roke.co.uk

References

[1] B. Carpenter and S. Brim, “Middleboxes: Taxonomy and issues,” RFC 3234, Internet Engineering Task Force, Feb. 2002.

[2] B. Braden and B. Lindell, “A two-level architecture for internet signaling,” Internet Draft, Internet Engineering Task Force, Nov. 2001. Work in progress.

[3] H. Schulzrinne, H. Tschofenig, X. Fu, and J. Eisl, “A quality-of-service resource allocation client for CASP,” Internet Draft, Internet Engineering Task Force, Sept. 2002. Work in progress.

[4] M. Brunner, “Requirements for QoS signaling protocols,” Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[5] B. Hancock *et al.*, “Towards a framework for QoS signaling in the internet,” Internet Draft, Internet Engineering Task Force, Feb. 2002. Work in progress.

[6] T. Dierks and C. Allen, “The TLS protocol version 1.0,” RFC 2246, Internet Engineering Task Force, Jan. 1999.

[7] L. Ong and J. Yoakum, “An introduction to the stream control transmission protocol (SCTP),” RFC 3286, Internet Engineering Task Force, May 2002.

[8] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, “Stream control transmission protocol,” RFC 2960, Internet Engineering Task Force, Oct. 2000.

[9] D. Senie, “Network address translator (nat)-friendly application design guidelines,” RFC 3235, Internet Engineering Task Force, Jan. 2002.

[10] P. Calhoun *et al.*, “Diameter base protocol,” Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[11] H. Holbrook and B. Cain, “Source-specific multicast for IP,” Internet Draft, Internet Engineering Task Force, Feb. 2002. Work in progress.

[12] S. Bradner, “Key words for use in RFCs to indicate requirement levels,” RFC 2119, Internet Engineering Task Force, Mar. 1997.

[13] D. Katz, “IP router alert option,” RFC 2113, Internet Engineering Task Force, Feb. 1997.

[14] C. Partridge and A. Jackson, “IPv6 router alert option,” RFC 2711, Internet Engineering Task Force, Oct. 1999.

[15] J. Rajahalme, A. Conta, B. Carpenter, and S. Deering, “IPv6 flow label specification,” Internet Draft, Internet Engineering Task Force, June 2002. Work in progress.

[16] L. Berger and T. O’Malley, “RSVP extensions for IPSEC data flows,” RFC 2207, Internet Engineering Task Force, Sept. 1997.

[17] D. Harkins, C. Kaufman, *et al.*, “Propsal for the IKEv2 protocol,” Internet Draft, Internet Engineering Task Force, Apr. 2002. Work in progress.

[18] J. Manner *et al.*, “Localized RSVP,” Internet Draft, Internet Engineering Task Force, May 2002. Work in progress.

[19] D. Meyer, “Administratively scoped IP multicast,” RFC 2365, Internet Engineering Task Force, July 1998.

[20] J. Moy, “OSPF version 2,” RFC 2328, Internet Engineering Task Force, Apr. 1998.

[21] E. Guttman, C. Perkins, J. Veizades, and M. Day, “Service location protocol, version 2,” RFC 2608, Internet Engineering Task Force, June 1999.

[22] T. Narten, E. Nordmark, and W. Simpson, “Neighbor discovery for IP version 6 (ipv6),” RFC 2461, Internet Engineering Task Force, Dec. 1998.

[23] R. Droms, “Dynamic host configuration protocol,” RFC 2131, Internet Engineering Task Force, Mar. 1997.

[24] M. Mealling and R. Daniel, “The naming authority pointer (NAPTR) DNS resource record,” RFC 2915, Internet Engineering Task Force, Sept. 2000.

[25] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: a transport protocol for real-time applications,” RFC 1889, Internet Engineering Task Force, Jan. 1996.

[26] K. McCloghrie, D. Farinacci, and D. Thaler, “IPv4 multicast routing MIB,” RFC 2932, Internet Engineering Task Force, Oct. 2000.

[27] J. Nicholas, “Protocol independent multicast MIB,” Internet Draft, Internet Engineering Task Force, June 2002. Work in progress.

[28] A. Terzis, J. Krawczyk, J. Wroclawski, and L. Zhang, “RSVP operation over IP tunnels,” RFC 2746, Internet Engineering Task Force, Jan. 2000.

[29] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, “Resource ReSerVation protocol (RSVP) – version 1 functional specification,” RFC 2205, Internet Engineering Task Force, Sept. 1997.

[30] L. Berger, D. Gan, G. Swallow, P. Pan, F. Tommasi, and S. Molendini, “RSVP refresh overhead reduction extensions,” RFC 2961, Internet Engineering Task Force, Apr. 2001.

[31] X. Fu, C. Kappler, and H. Tschofenig, “Analysis on RSVP regarding multicast,” Internet Draft, Internet Engineering Task Force, June 2002. Work in progress.

[32] H. Schulzrinne, A. Rao, and R. Lanphier, “Real time streaming protocol (RTSP),” RFC 2326, Internet Engineering Task Force, Apr. 1998.

[33] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “SIP: session initiation protocol,” RFC 3261, Internet Engineering Task Force, June 2002.

[34] M. Rose, “The blocks extensible exchange protocol core,” RFC 3080, Internet Engineering Task Force, Mar. 2001.

[35] H. Tschofenig, “NSIS threats,” Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[36] H. Tschofenig, “RSVP security properties,” Internet Draft, Internet Engineering Task Force, June 2002. Work in progress.

[37] R. Housley, “Cryptographic message syntax,” RFC 2630, Internet Engineering Task Force, June 1999.

[38] J. Arkko *et al.*, “Security mechanism agreement for SIP sessions,” Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[39] D. Johnson, C. Perkins, and J. Arkko, “Mobility support in IPv6,” Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[40] F. Baker, B. Lindell, and M. Talwar, “RSVP cryptographic authentication,” RFC 2747, Internet Engineering Task Force, Jan. 2000.

[41] A. Medvinsky and M. Hur, “Addition of kerberos cipher suites to transport layer security (TLS),” RFC 2712, Internet Engineering Task Force, Oct. 1999.

[42] S. Kent and R. Atkinson, “IP authentication header,” RFC 2402, Internet Engineering Task Force, Nov. 1998.

[43] S. Kent and R. Atkinson, "IP encapsulating security payload (ESP)," RFC 2406, Internet Engineering Task Force, Nov. 1998.

[44] D. Harkins and D. Carrel, "The internet key exchange (IKE)," RFC 2409, Internet Engineering Task Force, Nov. 1998.

[45] M. Thomas *et al.*, "Kerberized internet negotiation of keys (KINK)," Internet Draft, Internet Engineering Task Force, Nov. 2001. Work in progress.

[46] W. Aiello, S. Bellovin, *et al.*, "Just fast keying (JFK)," Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[47] L. Blunk and J. Vollbrecht, "PPP extensible authentication protocol (EAP)," RFC 2284, Internet Engineering Task Force, Mar. 1998.

[48] P. Funk and S. Blake-Wilson, "EAP tunneled TLS authentication protocol (EAP-TTLS)," Internet Draft, Internet Engineering Task Force, Mar. 2002. Work in progress.

[49] H. Andersson, S. Josefsson, G. Zorn, *et al.*, "Protected extensible authentication protocol (PEAP)," Internet Draft, Internet Engineering Task Force, Feb. 2002. Work in progress.

[50] S. Blake-Wilson *et al.*, "Transport layer security (TLS) extensions," Internet Draft, Internet Engineering Task Force, May 2002. Work in progress.

[51] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 internet public key infrastructure online certificate status protocol - OCSP," RFC 2560, Internet Engineering Task Force, June 1999.

[52] D. McDonald, C. Metz, and B. Phan, "PF_KEY key management API, version 2," RFC 2367, Internet Engineering Task Force, July 1998.

[53] S. Yadav, R. Yavatkar, R. Pabbati, P. Ford, T. Moore, S. Herzog, and R. Hess, "Identity representation for RSVP," RFC 3182, Internet Engineering Task Force, Oct. 2001.

[54] S. Bradner, A. Mankin, and J. Schiller, "A framework for purpose built keys (PBK)," Internet Draft, Internet Engineering Task Force, July 2002. Work in progress.

[55] R. Glenn and S. Kent, "The NULL encryption algorithm and its use with IPsec," RFC 2410, Internet Engineering Task Force, Nov. 1998.

[56] P. Calhoun, S. Farrell, and W. Bulley, "Diameter CMS security application," Internet Draft, Internet Engineering Task Force, Mar. 2002. Work in progress.

[57] Microsoft, "Microsoft authorization data specification v. 1.0 for Microsoft Windows 2000 Operating Systems," tech. rep., Microsoft, Redmond, Washington, Apr. 2000.

[58] L. Coene, "Stream control transmission protocol applicability statement," RFC 3257, Internet Engineering Task Force, Apr. 2002.

[59] F. Yergeau, "UTF-8, a transformation format of unicode and ISO 10646," RFC 2044, Internet Engineering Task Force, Oct. 1996.

```
                +-----------+
                | adversary |   transport layer connection
           +--->+           +<-----------------+  (4)
transport  |    +----+------+                  |
 layer     |     IPx |                         |
connection |         |scout reply              v
   (3)     |         | (IPx)              +---+-------+
           v         |  (2)               |   Access  |
    +------+-----+   |       /----------->+   Router  +--------
    | scout msg  +<--+      / scout       +-----------+
    | initiator  +---------/  request         IPr
    +------------+             (1)
        IPi
```

Figure 3: Man-in-the-Middle Attack

```
                                 +--------+
               +-----------------+ Router +---->
               |     SIDx        |   4    |
           +---+----+            +--------+
           | Router |                     SIDx
    +------+   3    +<-----+
    |      +---+----+      |
    |                      |
    |     SIDx             |
+---+----+             +---+----+
| Router |             | Router |
|   1    |             |   2    |
+---+----+             +---+----+
    |                      ^
    |     SIDx             |     SIDx
+---+----+             +---+----+
| Access |             | Access |
| Router |             | Router |
|   1    |             |   2    |
+---+----+             +---+----+
    |                      ^
    |     SIDx             |      SIDx
+---+----+             +---+-----+
| Mobile |             |Adversary|
| Node   |             |         |
+--------+             +---------+
```

Figure 4: Session Ownership Problem

```
----------------------------------------+   +----------------
                                        |   |
                                        |   |
    +----+        +----+        +----+  |   |    +--+-+
----+ R1 +--------+ R2 +--------+ R3 +--+---+----| R4 +------
    +----+        +----+        +----+  |   |    +--+-+
                                        |   |
        Administrative Domain A         |   | Admin.Domain B
----------------------------------------+   +----------------
```

Figure 5: Network Topology Hiding