CASP – Cross-Application Signaling Protocol

Henning Schulzrinne
August 27, 2002
Overview

- Protocol properties
- Message delivery
- Transport protocol usage
- Message forwarding
- Message format
- Next-hop discovery
  - Scout protocol
- Mobility and route changes
- Protocol heritage
What is CASP?

- **Generic signaling service**
  - establishes state along path of data
  - one sender, typically one receiver
    - can be multiple receivers → multicast
  - *can* be used for QoS per-flow or per-class reservation
  - but not restricted to that

- avoid restricting users of protocol (and religious arguments):
  - sender vs. receiver orientation
  - more or less closely tied to data path
    - router-by-router
    - network (AS) path
CASP network model – on-path

- CASP nodes form CASP chain
- not every node processes all client protocols:
  - non-CASP node: regular router
  - omnivorous: processes all CASP messages
  - selective: bypassed by CASP messages with unknown client protocols
CASP network model – out-of-path

- Also route network-by-network
- Can combine router-by-router with out-of-path messaging
CASP protocol structure

- **client layer** does the real work:
  - reserve resources
  - open firewall ports
  - ...

- **messaging layer**:
  - establishes and tears down state
  - negotiates features and capabilities

- **transport layer**:
  - reliable transport

- **scout protocol**:
  - UDP
  - IP router alert
CASP messages

- Regular CASP messages
  - establish or tear down state
  - carry client protocol

- Scout messages
  - discover next hop

- Hop-by-hop reliability
- Generated by any node along the chain
Most signaling messages are small and infrequent

but:
- not all applications → e.g., mobile code for active networks
- digital signatures
- re-"dialing" when resources are busy

Need:
- reliability → to avoid long setup delays
- flow control → avoid overloading signaling server
- congestion control → avoid overloading network
- fragmentation of long signaling messages
- in-sequence delivery → avoid race conditions
- transport-layer security → integrity, privacy

This defines standard reliable transport protocols:
- TCP
- SCTP

Avoid re-inventing wheel → see SIP experience
CASP transport protocol usage

- One transport connection → many M- & C-layer sessions
- may use multiple TCP/SCTP ports
- can use TLS for transport-layer security
  - compared to IPsec, well-exercised key establishment
  - not quite clear what the principal is
- re-use of transport →
  - no overhead of TCP and SCTP session establishment
  - avoid TLS session setup
  - better timer estimates
  - SCTP avoids HOL blocking
Message forwarding

- Route stateless or state-full:
  - stateless: record route and retrace
  - state-full: based on next-hop information in CASP node

- Destination:
  - address → look at destination address
  - address + record → record route
  - route → based on recorded route
  - state forward → based on next-hop state
  - state backward → based on previous-hop state

- State:
  - no-op → leave state as is
  - ADD → add message (and maybe client) state
  - DEL → delete message state
Message format

- No M-layer distinction between requests and responses
  - just routed in different directions
  - client protocol may define requests and responses

- Common header defines:
  - destination flag
  - state flag
  - session identifier
  - traffic selector: identify traffic "covered" by this session
  - message sequence number
  - response sequence number
  - message cookie → avoid IP address impersonation
  - origin address → may not be data source or sink
  - destination address or scope
Message format, cont'd

- Limit session lifetime
- Avoid loops → hop counter
- Mobility:
  - dead branch removal flag
  - branch identifier
- Record route: gathers up addresses of CASP nodes visited
- Route: addresses that CASP message should visit
Capability negotiation

- CASP has named capabilities
  - including client protocols

- Three mechanisms:
  - discovery: count capabilities along a path
    - "10 out of 15 can do QoS"
  - record: record capabilities for each node
  - require: for scout message, only stop once node supports all capabilities (or-of-and)

- avoid protocol versioning
Next-hop discovery

- Next-in-path service
  - enhanced routing protocols → distribute information about node capabilities in OSPF
  - routing protocol with probing
  - service discovery, e.g., SLP
  - first hop, e.g., router advertisements
  - DHCP
  - scout protocol

- Next AS service
  - touch down once per autonomous system (AS)
  - new DNS name space: ASN.as.arpa, e.g., 17.as.arpa
  - use new DNS NAPTR and SRV for lookup
    - similar to SIP approach
Next-hop discovery

- scout messages are special CASP messages
- limited < MTU size
- addressed to session destination
- UDP with router alert option → get looked at by each router
- reflected when matching CASP node found
Mobility and route changes

- avoids session identification by end point addresses
- avoid use of traffic selector as session identifier
- remove dead branch
The weight of CASP

- CASP state = transport state + CASP M-state + client state
- M-state = two sockets
- transport state = $O(100)$ bytes $\rightarrow$ 10,000 users consume 1 MB
Conclusion

- CASP = unified infrastructure for data-affiliated sessions
- avoid making assumptions except that sessions want to "visit" data nodes or networks
- not just mobility, but also mobility
- protocol framework in place
  - but need to work out packet formats
### CASP properties

- **Network friendly**
  - congestion-controlled
  - re-use of state across applications

- **transport neutral**
  - any reliable protocol
  - initially, TCP and SCTP

- **policy neutral**
  - no particular AAA policy or protocol
  - interaction with COPS, DIAMETER needs work

- **soft state**
  - per-node time-out
  - explicit removal

- **extensible**
  - data format
  - negotiation
CASP properties, cont'd.

- Topology hiding
  - not recommended, but possible

- Light weight
  - implementation complexity
  - security associations (re-use)
  - may not need kernel implementation