Protocol Pros A.Mishra,P.Acharya,A.Lazo,B.Edgerley
Request for Comments: 2211 S.Ryan,J.Villarreal,D.Schelanko
Texas A&M-Corpus Christi
Nov 2022

Message Sending Protocol

Status of this Memo

This RFC depicts an application-oriented protocol that describes the entire communication between different components in an Autonomous Intersection Management System. Components that are willing to integrate AIMS system in their own system may use this protocol. The current version of this protocol is based on the latest mechanism of the Autonomous Intersection Management System, as on November 11, 2022.

Discussion

The Message Sending Protocol for AIMS is used to handle the full-fledged communication within components of the AIMS, along with any other external agent that wishes to integrate and communicate with the AIMS. Based on the current scope of use-cases of AIMS, there would essentially be three communications, between three different systems.

These communications would take place on specific use-cases of the AIMS project. Different communications encompassing different use-cases would work on top of different transport layer protocols, either connection-oriented, or connection-less protocols.

Message Sending Protocol takes care of the communication between the AIMS Core Module and the other components within or without the AIMS system. Few such communication instances could be the passing of messages between the AIMS Message Sending Module and agents passing through an intersection. Likewise, the interaction between AIMS Core Processor and Smart City Management System, which is not an integral part of AIMS, is also described in the Message Sending Protocol.

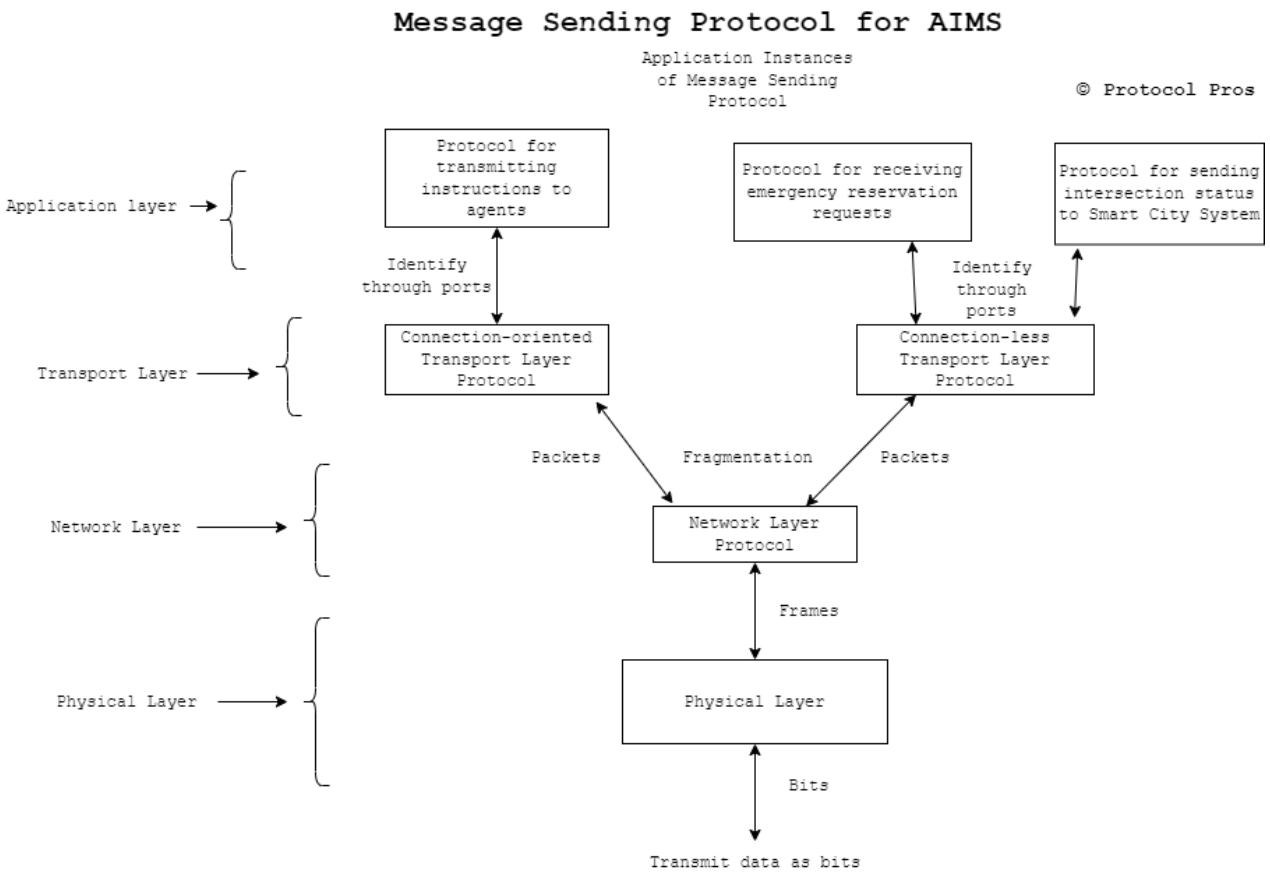


Fig: Instances of Message Sending Protocol in AIMS

TCP Based Message Sending Service

One of the communications in the existing implementation of AIMS works on top of Connection-oriented Transport Layer Protocol, namely TCP. This communication required a reliable and secured transmission of messages across the two partied, thus it functions on top of TCP.

The use-case is the following: Instructions generated by the AIMS Core Processor for all the agents in an intersection is to be transmitted to every agent in a secured and reliable manner. Thus, the transmission of instructions from AIMS Message Sending Module to every agent passing

through an intersection occurs through TCP Based Message Sending Service.

The connection-establishment of these two entities is done whenever an agent approaches an intersection, followed by two possible actions. Either it raises an emergency reservation request explicitly, which is also done through Message Sending Protocol, but would be discussed later. Or, if it simply passes through the intersection, and the video camera pertaining to AIMS captures the license plate and streams it to AIMS Video Streaming Module, which then gets recognized by the AIMS OCR Module and transmits the agent behavior to AIMS Core Processor. Both these cases imply the need to make a reservation for that agent in certain intersection.

The typical 3-way handshake mechanism would be used for maintaining the connection establishment using the CON, ACK flags along with Sequence and Acknowledgement Number, along with subsidiary header fields such as Max Segment Size and Window Size. A Cookie can also be used throughout the connection in the Message Sending Protocol for ensuring reliable delivery of packets from AIMS Message Sending Module to every agent.

Similarly, connection termination also occurs using 3-way handshaking using the FIN and ACK Flags. It happens when the agent passes beyond the intersection by following its instructions received from AIMS.

Since Hashed Message Authentication Code mechanism is embedded in this communication, it would maintain the integrity of the message being delivered. Also, the header checksum would make sure that the delivery is being done in the correct destination. Using these underlying functionalities, it can be trusted that TCP-based Message Sending Module is reliable and would perfectly serve its purpose.

UDP Based Message Sending Service

There are two use-cases where fast communication is required, and low-latency is of higher priority than reliability. These use-cases use

that type of Message Sending Protocol which run on top of UDP-based transport layer protocol.

One of these use-cases is the communication that occurs between AIMS System as a separate entity and Smart City Management System as a standalone entity. In this communication, data being transmitted is relatively shorter and does not require high amount of protection.

Further, another such use-case is the communication between an agent, which raises an emergency reservation request while approaching an intersection, and the AIMS Message Sending Module which receives this emergency reservation request.

Message Syntax

The message will consist of different parts. The first part is a single octet depicting the protocol version, currently decimal 1. This is followed by the name of the user that the concerned message is being directed to, which shall be represented as an octet. This will precede the recipient terminal name, of a couple of octets, representing the application terminal of the message destination.

HMAC Code of one octet will follow, for the purpose of maintaining message integrity. Likewise, the username of sender of one octet will be embedded. The sixth part if the terminal name of the sender, essentially the message source.

The seventh part is a 1-bit number representing the transport layer protocol being used for the message, followed by another 3-bit number implying the urgency of message being transmitted. The ninth part is the cookie, which would ensure the packet delivery, especially in case of the emergency reservation request being raised.
The tenth part would be the actual message, or the message digest generated by running any dedicated encryption logic over the actual message, to be more specific.

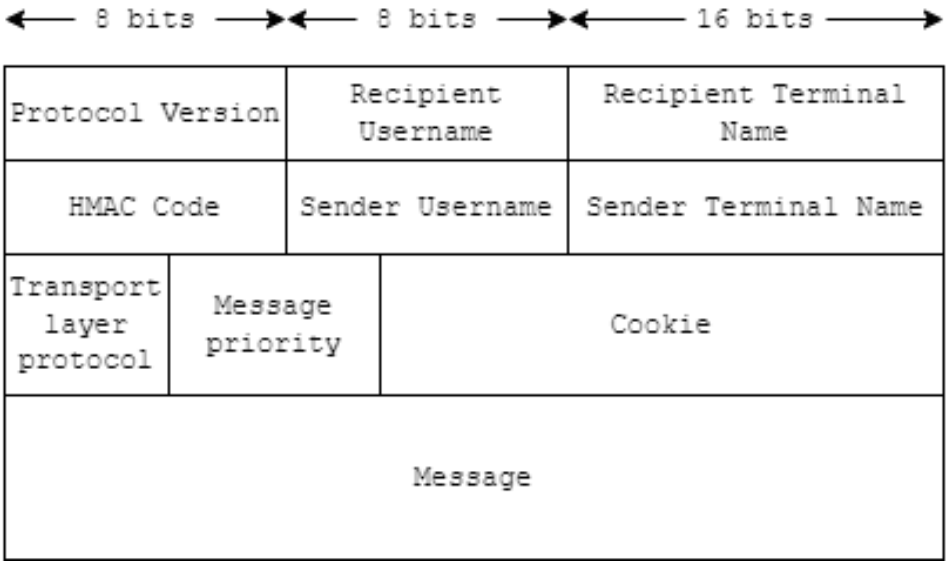
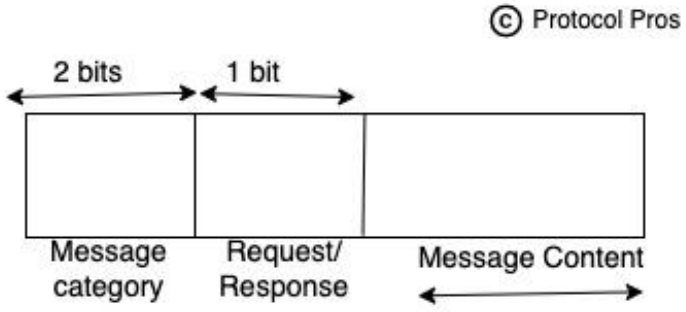


Fig: Message Syntax (Header)

The Actual Message Syntax

The tenth part of the above-mentioned classification of message syntax has to be described separately, as this syntax is more oriented towards the application, rather than the functionality. The first two bits would depict the Message Category, followed by another 1 bit which would act as a flag for differentiating response messages and request messages. The following part / third part is the content of the message, which will be based on the purpose of sending message.



Message Category

00 → Request for emergency reservation
01 → Reservation response (Instruction transmission)
10 → Request for intersection status
11 → Intersection status response

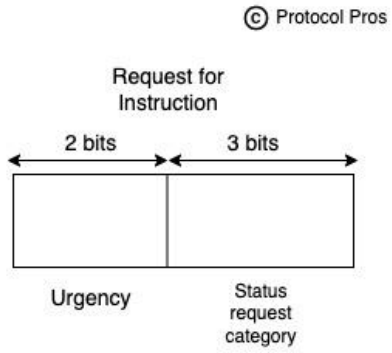
Request / Response

0 → Request to AIMS
1 → Response from AIMS

Message Content

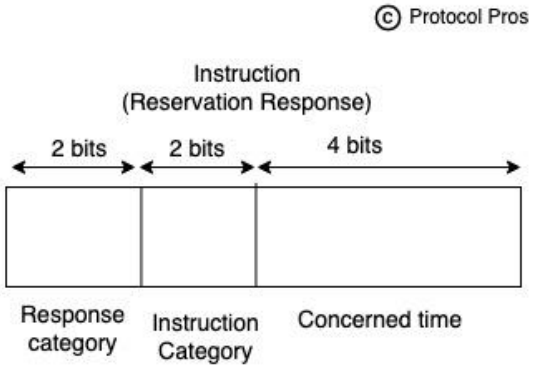
Request for emergency reservation

The first two bits of the message content is the urgency index of the message, followed by 3 bits depicting the agent category. The third and final 4 bits of the message content would be the concerned time stipulation, which indicates the amount of time within which it has to pass beyond the intersection.



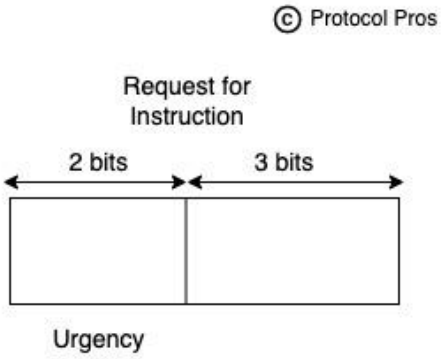
Reservation Response (Instruction Transmission)

The first couple of bits of the message content will represent the response category, as there is chance of conflict between response even within the message content. ’00’ is for instruction transmission. The following 2 bits will be the instruction category, or the command to be followed by the agent. Then there will be 4 bits dedicated for sending time, which will indicate the time within which the instructions are expected to be followed.



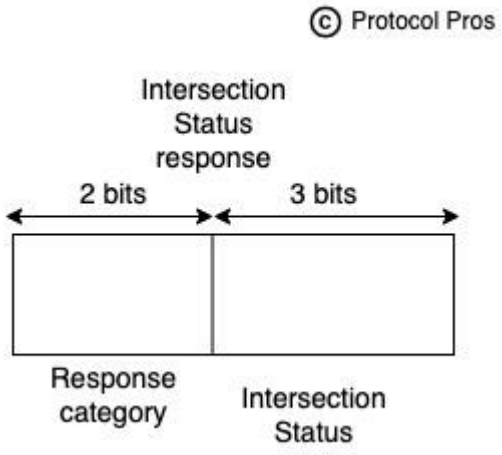
Request for intersection status

The first two bits would be the urgency or priority of requirement of the status of an intersection. This request would usually be raised by external agents like Smart City Management Systems. The following 3 bits would be the status response category.



Intersection Status Response

The first two bits would be the response category. '01' for intersection status response. Then the following 3 bits would depict the intersection status, among 8 possible status categories.



Advisories

It is advisable for external entities like Smart City Systems to raise urgency index only when required, as it would lead to clash between emergency reservation and intersection status request.

Checksums are always used in both TCP and UDP versions of this service to ensure whether the header is correct or not.

Security Considerations

The instruction, or the reservation response is expected to be transmitted not as a message itself, but as a message digest generated as a response to any hash function dedicated within a functionality, or the cipher text of any encryption logic integrated within a functionality.