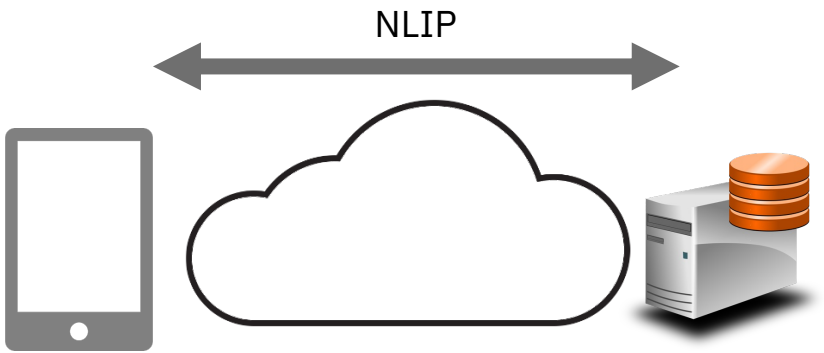
# Scenario

A Chat Client is interacting with a NLIP Server and interacting via conversational mode

- The NLIP Server may be responding directly
- The NLIP Server may be responding via backend systems or other mechanisms

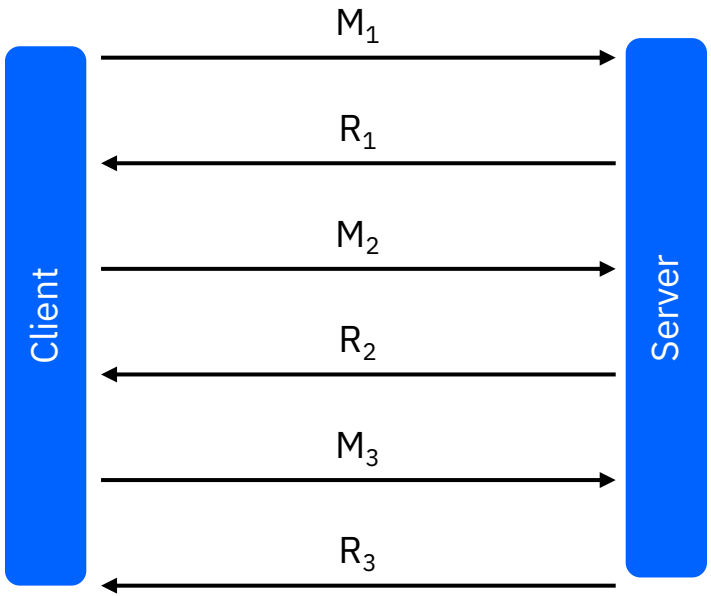
How can one maintain a stateful conversation with the NLIP Server

- What are the options and how does one handle the challenges in



# Conversation

Each Interactive Session between the client and the server consists of multiple Exchanges




The context of a message $M_i$ sent by client is the previous set of messages and responses $\{M_1,R_1...M_{i-1},R_{i-1}\}$

Question:

– Who stores this context

- Client stores the context
- Server stores the context
- Both store the context

# Option 1: Server stores context

Current Implementation in github

Supported by present protocol spec.

Pros:

- Less bandwidth used in communication

Cons

- Malicious client can cause DoS attack
- Server side storage needs grow (may be okay)

Choices:

- When does the new context begin and end?

Context Management

- Context starts and stops with HTTP Connection
  - Ties NLIP down to HTTP
- Context is maintained using NLIP token and correlator subformat
  - Context maintained at NLIP level
  - Token created and managed by the server
  - Maintained as long as tokens are exchanged
    - Removed if token is not used for some time
    - Can be removed if context size exceeds limit

# Option 2: Client stores context

Protocol would need augmentation

Pros:

- Less load on Server
- More secure – clients can not delay

Cons

- Additional bandwidth consumed

Choices:

- When does the new context begin and end?

Context Management

- Context starts and stops with HTTP Connection
  - Ties NLIP down to HTTP
- Context is maintained using NLIP token and correlator subformat
  - Context maintained at NLIP level
  - Token maintained and created by the client
  - Context transferred as structured JSON
    - Need to standardize how to mark who sent the request and response
      - » Fields: client & server; or
      - » Fields: ID of originator

# Option 3: Hybrid

Protocol would need augmentation

3a:

- Client and Sever negotiates who will manage the context
- An initial control message
  - Exchange in natural language to determine who will manage for each client

3b

- Server manages context for a subset of clients (e.g. authenticated clients)
  - Other clients (e.g. unauthenticated clients) need to manage their own context
- An initial control message
  - Exchange for server to declare its policies for context management

# Discussion

Which option to select?

If selecting option 2 or 3

– Suitable protocol extensions