Tactical MANET Project Requirements

Mahmoud Adas Yosry Mohammad Ahmed Mahmoud Abdulrahman Khalid

July 15, 2021

1 Abstract

This document lists details of our graduation project requirements and specifications.

2 Project Description

A `mobile ad-hoc network` communication system for military, for operations in areas with no internet infrastructure. The system connects the command center(s) with deployed units in two-way communications.

3 System Architecture

The system is composed of devices (nodes) running linux-based operating systems and have certain programs running in them.

3.1 Nodes

All nodes are provided with wireless communication modules that follow `IEEE 802.11` standards.

There are 2 types of nodes: units and command centers.

3.1.1 Units

Devices with deployed units in the operation field, connected with:

- LCD screen with resolution of `48x84` pixels.
- Helmet video camera.
- Audio input.
- Keybad.
- GPS (or any other position detection system.)
- Heartbeat sensor.

Features:

- Low power consumption.
- Running on battery.
- Low wireless range.
- High mobility.
- Operated by one person.

3.1.2 Gateways

Devices deployed on semi-stationary vehicles, connected to units and centers. Acts as a gateway between the 2 groups.

- Low-gain high-frequency antennas, connected to units.
- High-gain low-frequency high-power antennas, connected to command centers.
- High power consumption.
- Medium mobility.
- High wireless ranges.
- Operated by one person.
- Acts as a unit.

3.1.3 Command Centers

High-end computers at the command and control centers, accessed by units leaders.

Features:

- Capabale of high power consumption.
- Powerfull CPUs.
- Big storage and RAM.
- Operated by multiple peeople with multiple wide screens.

- Wide wireless range.
- Installed nearby the operation field, and has a connection to devices in the field.
- Low (or zero) mobility.

3.2 Programs

Programs running in devices are running as daemons, started at the startup of the system and are always running and restarted on failure.

3.2.1 Units

Each unit has a public and private key, and a map of command centers `IPs` and their corresponding public keys.

> Extension: units announce their `IPs` to command centers and share their keys dynamically.

A unit device has 2 programs: - `Router`: implements routing protocol. - `Unit Client Daemon`: Connected to device hardware and network interface and provide all unit features.

3.2.2 Command Centers

Each command center has a public and private key, and a map of units `IPs` and their corresponding public keys.

> Extension: command centers announce their `IPs` to units and share their keys dynamically.

A command center computer has 3 programs: - `Router`: implements routing protocol, same router as in unit devices. - `Command Client Daemon`: Exposes an interface to `UI` program, connects to units clients and handles all communication with units. - `Command Client UI`: Connects to `Command Client Daemon`, shows all data in the daemon and controls it.

4 Functional Requirements

4.1 Units

- Stream video from combat cameras to command center(s) only if the latter requested them. Video streaming terminates if the unit received an end-stream request, or the start request wasn't refreshed after 1 minute.
- Stream the heartbeat & location of the device owner and their position every 10 seconds.
- Store all the recorded video and sensors (location & heartbeat) data locally.
- If the device user requested:
  - Send audio messages from the microphone.
  - Send code messages (every code has its predefined meaning.)
- Receive audio messages from command centers into a queue.
- Play received audio messages from the queue instantly.
- Receive and show code messages.

4.2 Command Centers

- Send audio command & command codes to a single unit (TCP).
- Send audio commands to a group (multicast) or everyone (unlimited-radius broadcast) (UDP).
- Store all sent and received data.
- Show old data (audio, messages, videos, sensor data)
- Show notifications when an audio message is received and an option to play it.
- Show video streams as they are received.
- Show map with group color.
- If sensor data isn't received in 2 minutes, mark it as inactive.
- If a unit's heartbeat is below a threshold, mark it as in danger.

4.2.1 Internal Interface

- API `(UI <-> Daemon)`:
  - POST /audio-msg , /msg (dst_IP / group_id is a parameter, body is payload)

    - GET /audio-msgs , /msgs , /videos , /sensors-data (unit IP is a parameter)
- Websocket `(Daemon -> UI)`: audio, msg, video (frame), sensor data

4.3 Transfered Data

4.3.1 Command Center to Unit

- Unicast (TCP): Audio message (recording), Code message (predefined integers).
- Multicast / Broadcast (UDP): Audio command (addressed to all nodes or anyone in group), StartVideoStreaming request and EndVideoStreaming request.

4.3.2 Unit to Command Center

- Unicast (TCP): Audio message (recording), Message code (predefined integers).
- Unicast (UDP): Video stream, sensor data (heartbeat & location).

4.3.3 Required Models

- Audio message.
- Code message.
- Sensor data (heartbeat & location).
- Video fragment.

5 Non-functional Requirements

5.1 Reliability

The following must be delivered reliably (with gurantee of delivery):

- Code messages.
- Audio messages.

The following can be delivered unreliably (*no* gurantee of delivery):

- Video streams.
- Position and heartbeat messages (minimum 80% delivery success rate).

5.2 Speed

The system allows nodes to communicate with low latency and high throughput. Video streams must be viewable at minimum of 20 fps.

5.3 Routing

The system uses a complex routing protocol that utilizes redundancy in the topology to increase communication reliability.

The routing protocol is multipath-multicasting with Multiple Description Coding (MDC) which is optimized for video streaming in ad-hoc networks.

5.4 Security

- All transmitted data are encrypted.
- Authentiction is required for accessing command center by its UI.
- All stored data in command centers and units are encrypted.
- Units don't persist any data, messages self destruct after a 3 minutes of receiving them.

6 Testbeds

The system will be tested in 2 different environments: virtual and actual hardware.

6.1 Virtual

Using virtualization/emulation, each node (unit/command center) will be deployed in a virtual machine. Each node will have a static ip equivalent to that stored in nodes databases.

Ther should be `UI` for units' clients that: - connects with them over forwarded ports, - receives their screens and audio, - and sends them button actions and fake audio/video/position/heartbeat inputs,

Mininet-wifi will be used to simulate the wiereless connections and create topologies.

The following mobility models should be tested:

- Random Walk
- Truncated Levy Walk
- Random Direction
- Random Way Point
- Gauss Markov
- Reference Point
- Time Variant Community

Different toplogies with up-to 25 nodes should be tested.

6.2 Hardware

1. Install clients on our laptops.
2. Create a minimum topology with at least one command center.
3. Use the virtual `UI` for unit client.
4. Test streaming video/audio in a small ad-hoc network.
5. Remove one unit and test that the rest of the units noticed that and changed their routes.

   Extension: create actual hardware for the unit, modify the client to read actual inputs instead of taking them virtually.

7 Deliverables

- Source code of all programs listed in Subsection 3.2.
- Instructions on how to:
  - Configure devices, connect inputs.
  - Install all dependencies.
  - Configure all software.
  - Install and run all software.
- A paper that describes the modification(s) to the routing protocol, if any.
- Experiments' results about latency and throughput using different mobility models.