



# Entrega 5

# INE-5424



# Mudanças em relação ao P4

## Time Sync Manager (Responsável pela sincronização temporal do veículo):

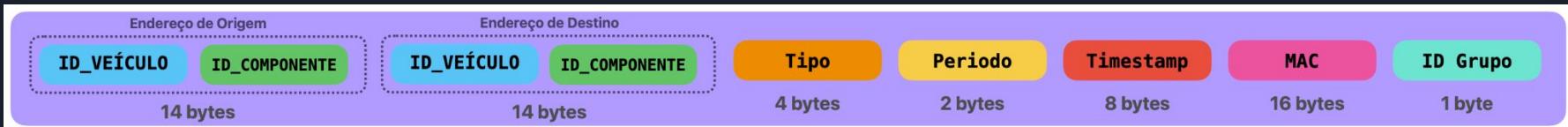
- Atualizado para funcionar apenas como Slave na sincronização de tempo.
  - Removida lógica e comportamento de líder.
  - Agora os Grandmaster Clock serão sempre as RSUs.
- Quando um veículo ingressa em um novo grupo, o RSU Handler repassa as informações do líder para o Time Sync Manager.
- TimeSyncManager passa a considerar apenas mensagens SYNC do líder.
  - Envia DELAY\_REQ apenas para a RSU líder.

# Entrega 5: Comunicação Segura em Grupos

Mudança no Header para conter o ID do grupo e MAC da mensagem

Formatos:

```
using MAC_key = std::array<uint8_t, 16>;      // (16 bytes)
using Group_ID = uint8_t;                        // (1 byte)
```



# Entrega 5: Comunicação Segura em Grupos

Adição de mensagens do tipo JOIN\_REQ e JOIN\_RESP e POSITION\_DATA  
Possibilitar comunicação entre RSU e Veículos.

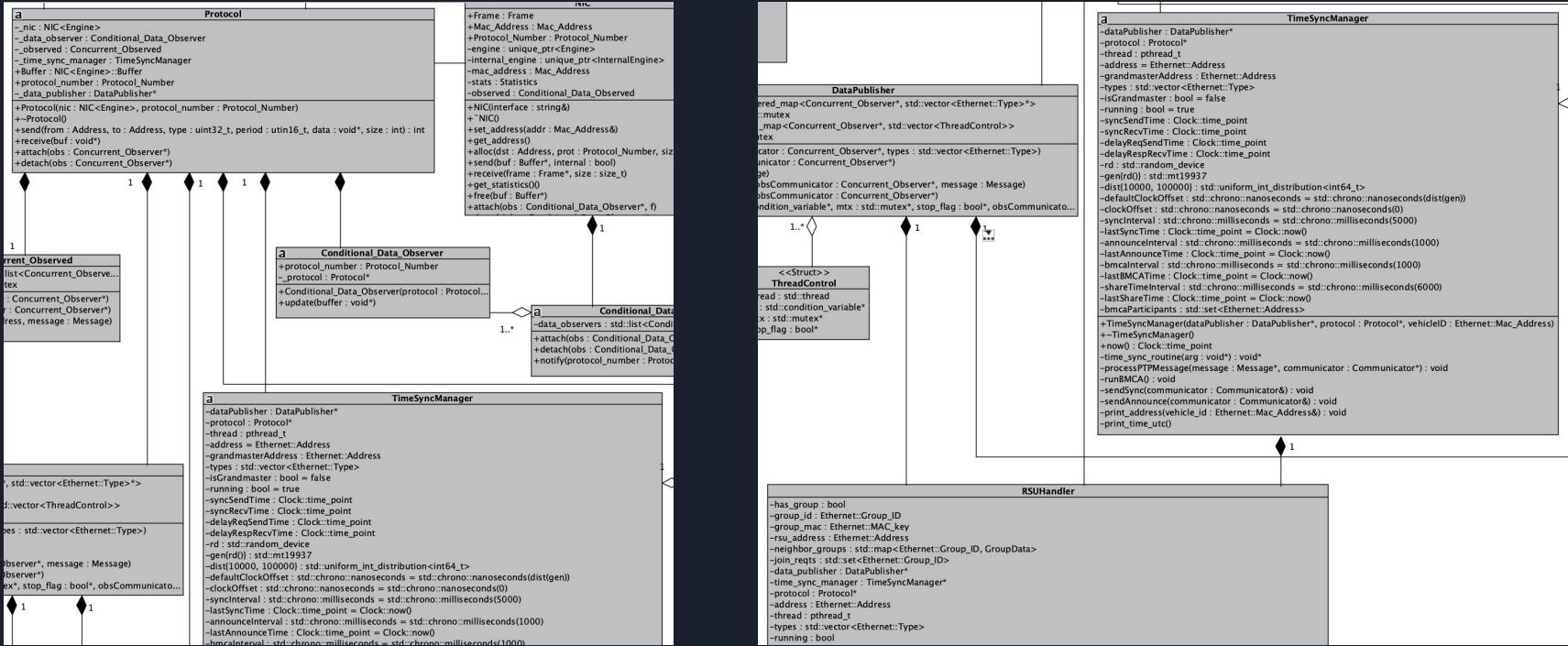
```
// Tipos de dados utilizados para comunicacao entre RSU e veiculos.  
Ethernet::Type constexpr static TYPE_RSU_JOIN_REQ = 0x0A; // (4 bytes)  
Ethernet::Type constexpr static TYPE_RSU_JOIN_RESP = 0x0B; // (4 bytes)  
  
// Tipo de dado enviado pelos componentes que fornecem a posicao dos veiculo.  
Ethernet::Type constexpr static TYPE_POSITION_DATA = 0x0D;
```

Adição de estruturas de posição do veículo e área de atuação das RSUs

```
// Estrutura de dados para envio da Localizacao dos veiculos.  
struct Position {  
    int x;  
    int y;  
};  
  
// Estrutura de dados para envio do Quadrante dos RSUs.  
struct Quadrant {  
    int x_min;  
    int x_max;  
    int y_min;  
    int y_max;  
};
```

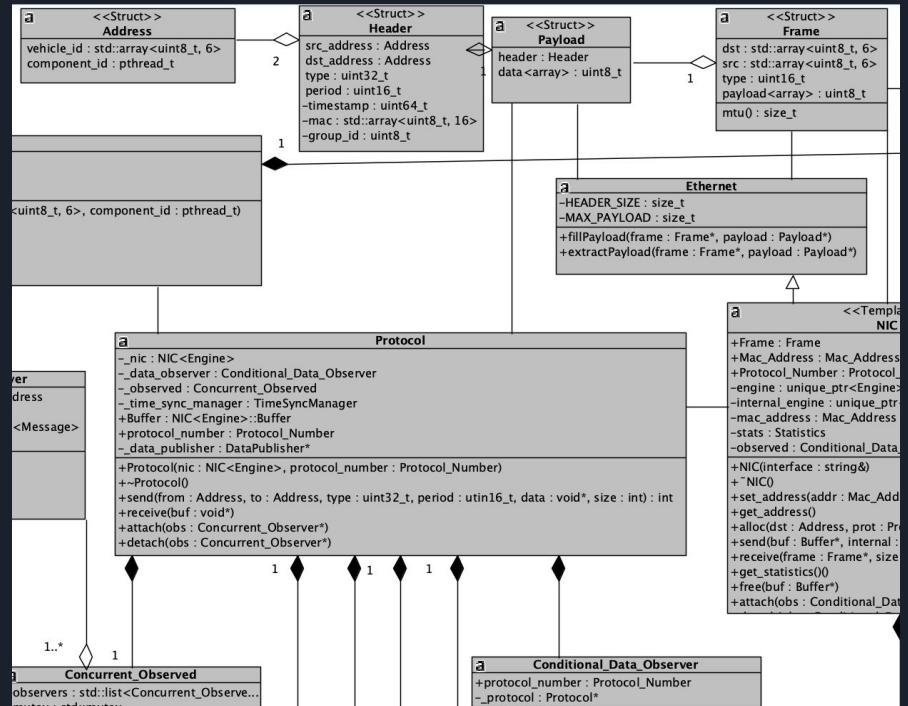
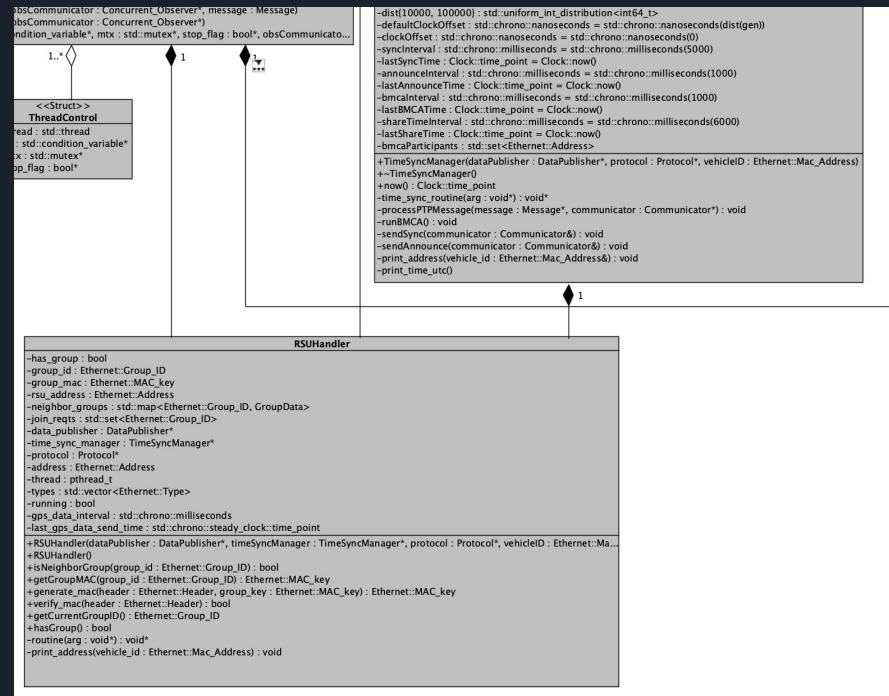
# Entrega 5: Comunicação Segura em Grupos

## Diagrama de Classes Atualizado



# Entrega 5: Comunicação Segura em Grupos

## Diagrama de Classes Atualizado





# Classe RSU

- Atua em um quadrante específico da área do sistema.
- Atua como líder do grupo de veículos dentro desse quadrante.
- Gera a chave MAC do grupo via `generate_group_key()`
- Envia periodicamente mensagens SYNC (thread dedicada):
  - Utiliza pelos membros do grupo para sincronização de relógio.
  - Utilizada como Beacon para veículos fora do grupo.
- Processa solicitações de ingresso(`JOIN_REQ`)
  - Distribui a chave MAC aos novos membros do grupo
- Responde mensagens `DELAY_REQ` dos seus membros.

# Rotina do RSU

1. Envia mensagens de SYNC periódicas com informações da sua área de atuação (quadrante)

```
// Funcao de rotina da thread periodica para envio da mensagem SYNC.
static void* send_routine(void* arg) {
    ThreadData* data = static_cast<ThreadData*>(arg);
    RSU* self = data->instance;

    auto next_send = std::chrono::system_clock::now();

    std::unique_lock<std::mutex> lock(self->mutex);
    while (self->running) {
        next_send += std::chrono::milliseconds(100);

        // Envia PTP_SYNC junto com ID e Quadrante da RSU.
        Message message;
        message.setDstAddress({{0,0,0,0,0,0}}, (pthread_t)0);
        message.setType(Ethernet::TYPE_PTP_SYNC);
        message.setGroupID(self->group_id);
        //std::cout << "RSU " << (int)message.getGroupID() << " enviando SYNC" << std::endl;
        message.setData(reinterpret_cast<Ethernet::Quadrant*>(&self->quadrant), sizeof(Ethernet::Quadrant));
        message.setPeriod(0);
        //std::cout << "RSU " << (int)self->group_id << " enviou SYNC" << std::endl;
        self->communicator->send(&message);

        self->cv.wait_until(lock, next_send, [&] { return !self->running; });
    }
    pthread_exit(nullptr);
}
```

**\*\*Realizado por uma Thread periódica dedicada\*\***

# Rotina do RSU

## 2. Processa solicitações de ingresso (JOIN\_REQ) e Responde DELAY\_REQ

```
while (self->running) {
    //std::cout << "RSU aguardando mensagens..." << std::endl;
    if (self->communicator->hasMessage()) {
        Message message;
        self->communicator->receive(&message);
        switch (message.getType()) {
            case Ethernet::TYPE_RSU_JOIN_REQ:
                // Responde veiculo com ID, MAC e Quadrante do grupo (RSU).
                //std::cout << "RSU " << (int)self->group_id << " recebeu JOIN_REQ" << std::endl;
                message.setType(Ethernet::TYPE_RSU_JOIN_RESP);
                message.setDstAddress(message.getSrcAddress());
                message.setGroupID(self->group_id);
                message.setMAC(self->mac);
                message.setPeriod(0);
                message.setData(reinterpret_cast<Ethernet::Quadrant*>(&self->quadrant), sizeof(Ethernet::Quadrant));
                //std::cout << "RSU " << (int)self->group_id << " enviou JOIN_RESP" << std::endl;
                self->communicator->send(&message);
                break;
            case Ethernet::TYPE_PTP_DELAY_REQ:
                // Responde veiculo com DELAY RESP.
                //std::cout << "RSU " << (int)self->group_id << " recebeu DELAY_REQ" << std::endl;
                message.setType(Ethernet::TYPE_PTP_DELAY_RESP);
                message.setDstAddress(message.getSrcAddress());
                message.setPeriod(0);
                //std::cout << "RSU " << (int)self->group_id << " enviou DELAY_RESP" << std::endl;
                self->communicator->send(&message);
                break;
            default:
                break;
        }
    }
}
```

\*\*Realizado por uma Thread dedicada\*\*



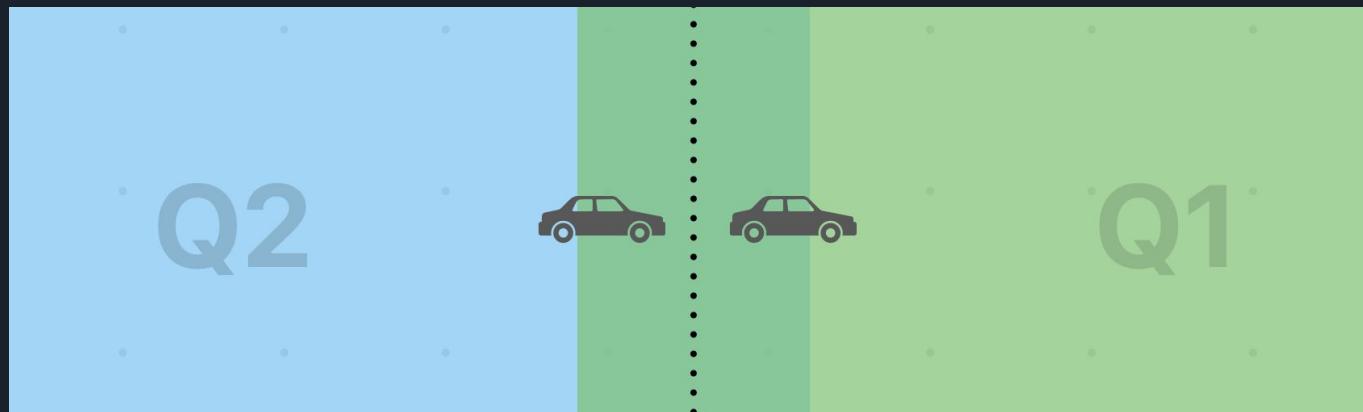
# Entrega 5: Comunicação Segura em Grupos

## Criação da classe **RSU Handler**:

- Interface do veículo para comunicação com RSU
- Solicita ingresso em grupos quando dentro ou próximo do quadrante
- Mantém registro de grupos vizinhos
- Verifica autenticidade das mensagens recebidas

# RSU Handler e Grupos Vizinhos

- Quando o RSU Handler verifica que o veículo está perto da borda/divisa com outro grupo, ele chama esse grupo de vizinho.
- Ingressar nos grupos vizinhos é importante para conseguir se comunicar com os veículos que pertencem a outro grupo e também estão na borda/divisa.



# Rotina do RSU Handle no recebimento de mensagem SYNC

1. Envia mensagem de interesse interno de GPS
2. Quando receber mensagem de GPS, pega o quadrante da mensagem SYNC

```
// Extrai o quadrante da mensagem SYNC.  
Ethernet::Quadrant quadrant = *reinterpret_cast<Ethernet::Quadrant*>(message.data());  
uint8_t group_id = message.getGroupID();
```

3. Se for do mesmo grupo, ignora.
4. Se for de outro grupo, verifica se é de grupo vizinho conhecido.
  - 4.1 Se não estiver perto, remove group\_id da lista de quadrantes vizinhos

```
if (self->neighbor_groups.count(group_id)) {  
    // Se o veículo se afastou do quadrante do grupo vizinho, remove dos vizinhos.  
    if (!near_quadrant) {  
        self->neighbor_groups.erase(group_id);  
    }  
}
```

# Rotina do RSU Handler no recebimento de mensagem SYNC

4.2 . Se o veículo estiver dentro do quadrante, remove os vizinhos e envia mensagem JOIN\_REQ

```
else if (in_quadrant) {
    self->neighbor_groups.erase(group_id);

    self->print_address(self->address.vehicle_id);
    std::cout << " Veiculo dentro ou proximo ao quadrante

    Message joinRequest;
    joinRequest.setDstAddress(message.getSrcAddress());
    joinRequest.setType(Ethernet::TYPE_RSU_JOIN_REQ);
    communicator.send(&joinRequest);
    self->join_reqts.insert(group_id);
}
```

# Rotina do RSU Handler no recebimento de mensagem SYNC

5 . Se SYNC veio de um grupo desconhecido

5.1 Verifica se já foi enviado JOIN\_REQ, se já foi, ignora

5.2 Se estiver dentro ou próximo do quadrante envia JOIN\_REQ

```
// Se estiver dentro ou próximo do quadrante, envia JOIN_REQ.  
if (near_quadrant) {  
    self->print_address(self->address.vehicle_id);  
    std::cout << " Veiculo dentro ou proximo ao quadrante do "  
  
    Message joinRequest;  
    joinRequest.setDstAddress(message.getSrcAddress());  
    joinRequest.setType(Ethernet::TYPE_RSU_JOIN_REQ);  
    communicator.send(&joinRequest);  
    self->join_reqts.insert(group_id);  
}
```

# Rotina do RSU Handler no recebimento de mensagem JOIN\_RESP

1. Remove o ID do grupo da lista de JOIN\_REQs já enviados e pega o quadrante da mensagem

```
// Remove o ID do grupo da lista de JOIN_REQs ja enviados.  
self->join_reqts.erase(message.getGroupID());  
  
// Pega o quadrante da mensagem.  
Ethernet::Quadrant quadrant = *reinterpret_cast<Ethernet::Quadrant*>(message.data());
```

2. Se estiver dentro do quadrante atualiza o novo grupo do veículo e notifica o TimeSyncManager sobre o novo líder

```
if (in_quadrant) {  
    if (!self->has_group) {  
        self->has_group = true;  
    } else {  
        // Remove threads periodicas do DataPublisher destinadas ao grupo antigo.  
        self->data_publisher->delete_group_threads(self->group_id);  
    }  
  
    // Atualiza novo grupo do veiculo.  
    self->group_id = message.getGroupID();  
    self->group_mac = message.getMAC();  
    self->rsu_address = message.getSrcAddress();  
  
    // Notifica TimeSyncManager sobre o novo lider.  
    self->time_sync_manager->setGrandmaster(message.getGroupID(), message.getSrcAddress());  
}  
  
**DataPublisher: Finaliza todas as threads periódicas de interesse do grupo antigo.**
```

# Rotina do RSU Handler no recebimento de mensagem JOIN\_RESP

3. Se estiver perto do quadrante, adiciona o grupo ao grupo de vizinhos

```
// Verifica se o veiculo esta proximo do quadrante da RSU.
} else if (near_quadrant) {
    // Adiciona grupo a estrutura de grupos vizinhos ao veiculo.
    self->neighbor_groups[message.getGroupID()] = {message.getSrcAddress(), message.getMAC()};

    self->print_address(self->address.vehicle_id);
    std::cout << " vizinho ao grupo da RSU " << (int)message.getGroupID() << std::endl;
    break;
```



# Rotina do RSU Handler no recebimento de mensagem POSITION\_DATA

1. Atualiza sua informação de posição atual do veículo com o novo dado recebido.

```
case Ethernet::TYPE_POSITION_DATA:  
    if (!first_position_received) { first_position_received = true; }  
    position = *reinterpret_cast<Ethernet::Position*>(message.data());  
    //std::cout << "RSU Handler recebeu nova posicao do veiculo: " << "  
    break;
```

# Estrutura do MAC

```
// Gera MAC usando o cabeçalho da mensagem (exeto campo mac) e a chave do grupo.  
Ethernet::MAC_key generate_mac(const Ethernet::Header& header, const Ethernet::MAC_key group_key) {  
    // Faz XOR dos bytes do cabeçalho (exeto mac) com a chave do grupo.  
    Ethernet::MAC_key mac;  
  
    // Copia a chave do grupo como base do MAC  
    std::memcpy(mac.data(), group_key.data(), 16);  
  
    // Obtém ponteiro para o início do header  
    const uint8_t* data = reinterpret_cast<const uint8_t*>(&header);  
  
    // Calcula o tamanho do cabeçalho sem o campo MAC (últimos 17 bytes: 16 de MAC + 1 de group_id)  
    constexpr size_t mac_offset = offsetof(Ethernet::Header, mac);  
    constexpr size_t mac_field_size = sizeof(Ethernet::MAC_key) + sizeof(Ethernet::Group_ID);  
    const size_t size_to_hash = sizeof(Ethernet::Header) - mac_field_size;  
  
    // Faz XOR do conteúdo do header com os 16 bytes do MAC base  
    for (size_t i = 0; i < size_to_hash; ++i) {  
        mac[i % 16] ^= data[i];  
    }  
  
    return mac;  
}
```

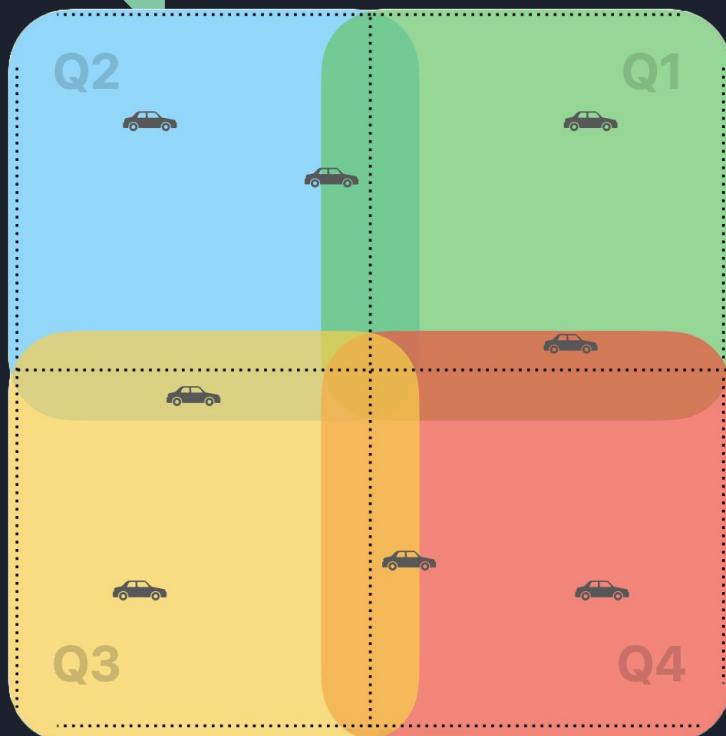
- Implementação simples usando XOR entre cabeçalho e chave do grupo
- Verificação compara o MAC recebido com o MAC calculado localmente

# Protocol: Filtro de mensagens

- Descarte das mensagens inválidas realizado dentro do receive do Protocol.
- Protocol descarta as mensagens de interesse/reposta que não viram do grupo que o veículo pertence ou vizinhos a ele.
- Realiza a checagem inicial pelo ID do grupo.
- Só verifica MAC caso mensagem seja do grupo
  - Menos processamento.

```
// Descarta mensagens de interesse/respostas externas que: nao pertencem
// ao grupo do veiculo, ou a nenhum grupo vizinho, ou que foram adulteradas.
if (_rsu_handler != nullptr) {
    // Verifica se mensagem eh externa.
    if (payload.header.src_address.vehicle_id != _nic->get_address()) {
        // Desconsidera mensagens enviadas pela RSU.
        if (payload.header.type != Ethernet::TYPE_PTP_SYNC &&
            payload.header.type != Ethernet::TYPE_PTP_DELAY_RESP &&
            payload.header.type != Ethernet::TYPE_RSU_JOIN_RESP) {
            // Descarta mensagens de grupos que o veiculo nao pertence e nao eh vizinho.
            if (payload.header.group_id != _rsu_handler->getCurrentGroupID() &&
                !_rsu_handler->isNeighborGroup(payload.header.group_id)) {
                return;
            } else {
                //std::cout << (int)_rsu_handler->getCurrentGroupID() << " RECEBEU INTERESE";
                // Verifica MAC da mensagem.
                if (!_rsu_handler->verify_mac(payload.header)) {
                    return; // Descarta mensagem se MAC invalido.
                }
                //std::cout << "MAC VALIDADO: PROCESSANDO MENSAGEM." << std::endl;
            }
        }
    }
}
```

# Teste de comunicação em grupos



Ambiente dividido em 4 quadrantes, cada um com uma RSU líder. RSUs nomeadas com o número do seu quatrante.

Em cada quadrante:

- 1 veículo estático no centro
- 1 veículo estático na borda

Veículos estáticos usam GPS Estático para posição fixa.

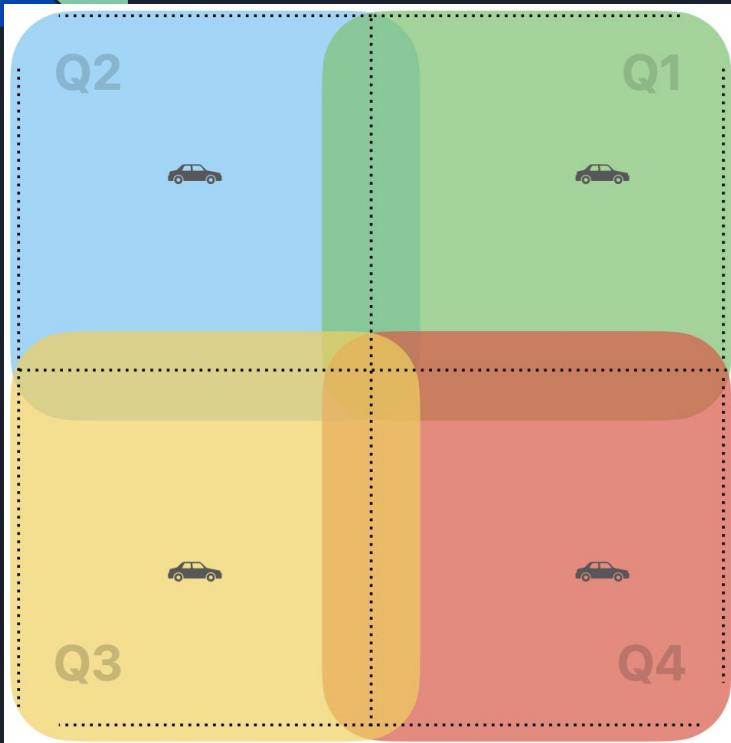
1 veículo dinâmico percorre os quadrantes (sentido anti-horário).

- Usa GPS Dinâmico para posição atualizada
- Possui Detector de Veículos para identificar veículos próximos.

Comunicação entre veículos do mesmo grupo ou vizinhos.

Detecção contínua de veículos próximos durante o trajeto.

# Teste de comunicação em grupos



## TESTE: Comunicação e Reconhecimento de Veículos em Diferentes Grupos.

Cada grupo (quadrante) possui dois veículos estáticos: centro e borda.  
Um Veículo Dinâmico circula entre todos os quadrantes no sentido anti-horário,  
Detectando veículos do seu grupo ou vizinho ao seu grupo (área da borda), quando for o caso.

### Parâmetros do teste:

Interface de rede: enp0s1  
Intervalo criação dos veículos (ms): 500  
Intervalo troca de posição do veículo dinâmico (ms): 2000  
Número de voltas do veículo dinâmico : 1  
Período de detecção de veículos (ms): 500

### Criando RSU para cada quadrante:

RSU criada com ID do grupo: 1 e MAC: D9:68:AB:DC:5F:6D:00:2A:7D:43:DD:21:7D:E9:B8:51  
RSU criada com ID do grupo: 2 e MAC: 4A:49:86:D7:B4:73:D9:97:BF:AC:59:F8:23:2E:4E:F0  
RSU criada com ID do grupo: 3 e MAC: F5:A4:7C:0A:79:A3:DF:43:BF:11:FC:7A:F3:66:20:0B  
RSU criada com ID do grupo: 4 e MAC: 0D:24:03:1B:F1:2E:FD:A6:8F:BA:BE:0F:0F:01:69:81

### Criando Veículos Estáticos no Centro de cada quadrante:

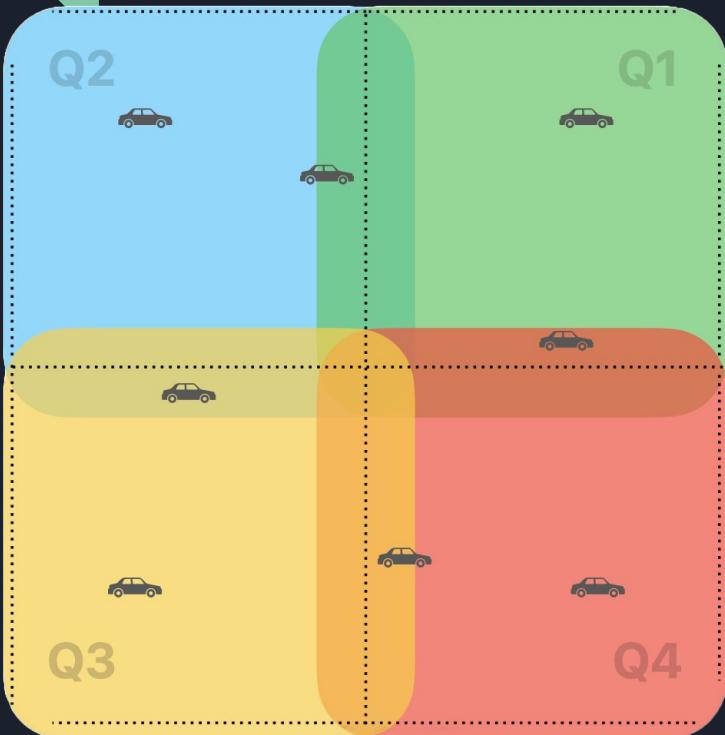
Centro do Quadrante 1  
Posição Veículo: x: 50, y:50  
Vehicle ID: 5:e:dc:00:b3:27:23 Veículo dentro ou próximo ao quadrante do RSU 1, enviando JOIN\_REQ.  
Vehicle ID: 5:e:dc:00:b3:27:23 RSU Handler recebeu JOIN\_RESP da RSU 1  
Vehicle ID: 5:e:dc:00:b3:27:23 ingressou no grupo da RSU 1  
Vehicle ID: 5:e:dc:00:b3:27:23 Líder Sincronização Temporal atualizado para RSU 1

Centro do Quadrante 2  
Posição Veículo: x: -50, y:50  
Vehicle ID: e6:bd:25:c4:2e:6c Veículo dentro ou próximo ao quadrante do RSU 2, enviando JOIN\_REQ.  
Vehicle ID: e6:bd:25:c4:2e:6c RSU Handler recebeu JOIN\_RESP da RSU 2  
Vehicle ID: e6:bd:25:c4:2e:6c ingressou no grupo da RSU 2  
Vehicle ID: e6:bd:25:c4:2e:6c Líder Sincronização Temporal atualizado para RSU 2

Centro do Quadrante 3  
Posição Veículo: x: -50, y:-50  
Vehicle ID: dd:1:e:0:a:e8:7b:42 Veículo dentro ou próximo ao quadrante do RSU 3, enviando JOIN\_REQ.  
Vehicle ID: dd:1:e:0:a:e8:7b:42 RSU Handler recebeu JOIN\_RESP da RSU 3  
Vehicle ID: dd:1:e:0:a:e8:7b:42 ingressou no grupo da RSU 3  
Vehicle ID: dd:1:e:0:a:e8:7b:42 Líder Sincronização Temporal atualizado para RSU 3

Centro do Quadrante 4  
Posição Veículo: x: 50, y:-50  
Vehicle ID: f5:b1:50:59:95:47 Veículo dentro ou próximo ao quadrante do RSU 4, enviando JOIN\_REQ.  
Vehicle ID: f5:b1:50:59:95:47 RSU Handler recebeu JOIN\_RESP da RSU 4  
Vehicle ID: f5:b1:50:59:95:47 ingressou no grupo da RSU 4  
Vehicle ID: f5:b1:50:59:95:47 Líder Sincronização Temporal atualizado para RSU 4

# Teste de comunicação em grupos



Criando Veículos Estáticos na Borda de cada quadrante:

Borda do Quadrante 1

Posição Veículo: x: 50, y:5

Vehicle ID: 1e:7f:20:da:06:a6 Veiculo dentro ou proximo ao quadrante do RSU 1, enviando JOIN\_REQ.  
Vehicle ID: 1e:7f:20:da:06:a6 Veiculo dentro ou proximo ao quadrante do RSU 4, enviando JOIN\_REQ.  
Vehicle ID: 1e:7f:20:da:06:a6 RSU Handler recebeu JOIN\_RESP da RSU 1  
Vehicle ID: 1e:7f:20:da:06:a6 ingressou no grupo da RSU 1  
Vehicle ID: 1e:7f:20:da:06:a6 Lider Sincronizacao Temporal atualizado para RSU 1  
Vehicle ID: 1e:7f:20:da:06:a6 RSU Handler recebeu JOIN\_RESP da RSU 4  
Vehicle ID: 1e:7f:20:da:06:a6 vizinho ao grupo da RSU 4

Borda do Quadrante 2

Posição Veículo: x: -5, y:50

Vehicle ID: 78:5e:be:b3:e1:bb Veiculo dentro ou proximo ao quadrante do RSU 1, enviando JOIN\_REQ.  
Vehicle ID: 78:5e:be:b3:e1:bb Veiculo dentro ou proximo ao quadrante do RSU 2, enviando JOIN\_REQ.  
Vehicle ID: 78:5e:be:b3:e1:bb RSU Handler recebeu JOIN\_RESP da RSU 1  
Vehicle ID: 78:5e:be:b3:e1:bb vizinho ao grupo da RSU 1  
Vehicle ID: 78:5e:be:b3:e1:bb RSU Handler recebeu JOIN\_RESP da RSU 2  
Vehicle ID: 78:5e:be:b3:e1:bb ingressou no grupo da RSU 2  
Vehicle ID: 78:5e:be:b3:e1:bb Lider Sincronizacao Temporal atualizado para RSU 2

Borda do Quadrante 3

Posição Veículo: x: -50, y:-5

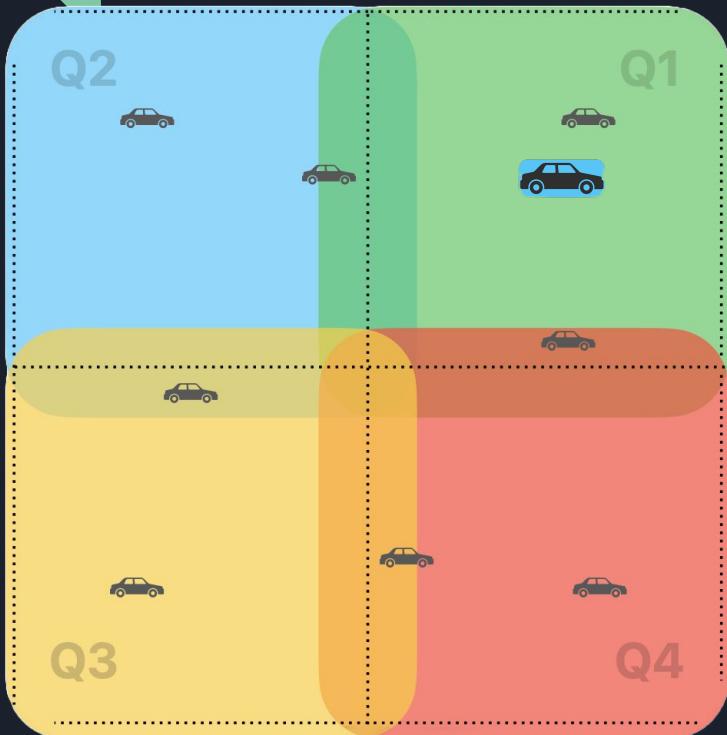
Vehicle ID: 32:27:50:50:f4:cd Veiculo dentro ou proximo ao quadrante do RSU 3, enviando JOIN\_REQ.  
Vehicle ID: 32:27:50:50:f4:cd Veiculo dentro ou proximo ao quadrante do RSU 2, enviando JOIN\_REQ.  
Vehicle ID: 32:27:50:50:f4:cd RSU Handler recebeu JOIN\_RESP da RSU 3  
Vehicle ID: 32:27:50:50:f4:cd ingressou no grupo da RSU 3  
Vehicle ID: 32:27:50:50:f4:cd Lider Sincronizacao Temporal atualizado para RSU 3  
Vehicle ID: 32:27:50:50:f4:cd Veiculo ja enviou JOIN\_REQ para o grupo 2, ignorando SYNC.  
Vehicle ID: 32:27:50:50:f4:cd RSU Handler recebeu JOIN\_RESP da RSU 2  
Vehicle ID: 32:27:50:50:f4:cd vizinho ao grupo da RSU 2

Borda do Quadrante 4

Posição Veículo: x: 5, y: -50

Vehicle ID: 4a:56:6b:4b:33:dc Veiculo dentro ou proximo ao quadrante do RSU 3, enviando JOIN\_REQ.  
Vehicle ID: 4a:56:6b:4b:33:dc Veiculo dentro ou proximo ao quadrante do RSU 4, enviando JOIN\_REQ.  
Vehicle ID: 4a:56:6b:4b:33:dc Veiculo ja enviou JOIN\_REQ para o grupo 3, ignorando SYNC.  
Vehicle ID: 4a:56:6b:4b:33:dc Veiculo ja enviou JOIN\_REQ para o grupo 4, ignorando SYNC.  
Vehicle ID: 4a:56:6b:4b:33:dc RSU Handler recebeu JOIN\_RESP da RSU 3  
Vehicle ID: 4a:56:6b:4b:33:dc vizinho ao grupo da RSU 3  
Vehicle ID: 4a:56:6b:4b:33:dc RSU Handler recebeu JOIN\_RESP da RSU 4  
Vehicle ID: 4a:56:6b:4b:33:dc ingressou no grupo da RSU 4  
Vehicle ID: 4a:56:6b:4b:33:dc Lider Sincronizacao Temporal atualizado para RSU 4

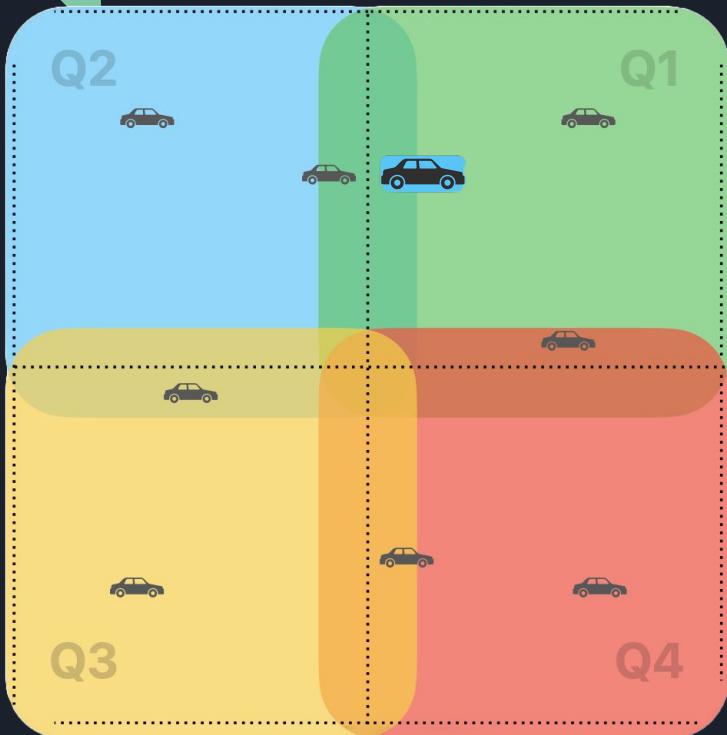
# Teste de comunicação em grupos



Criando Veiculo Dinamico.  
Iniciou volta número: 1

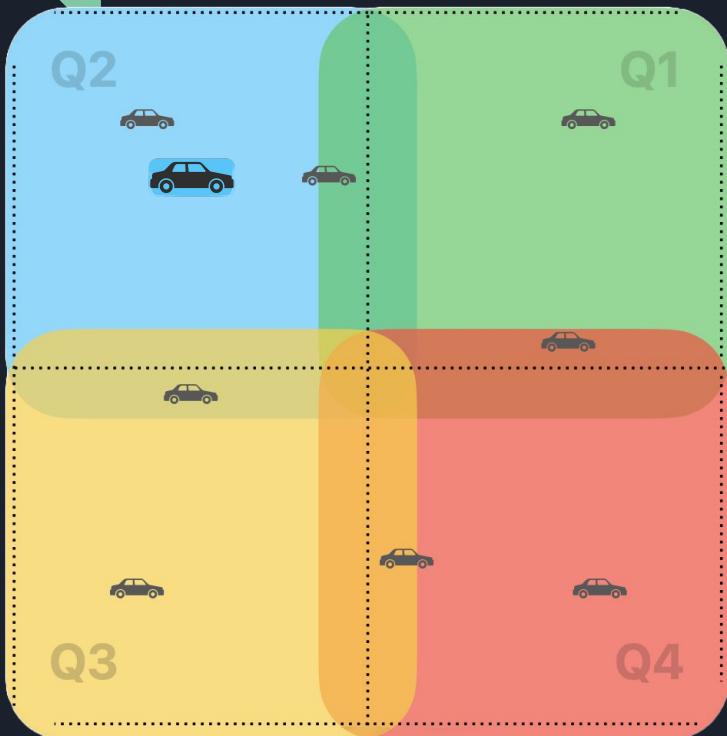
Nova Posição Veiculo Dinâmico: x: 50, y:50  
Vehicle ID: b4:8b:d0:e4:41:30 Veiculo dentro ou proximo ao quadrante do RSU 1, enviando JOIN\_REQ.  
Vehicle ID: b4:8b:d0:e4:41:30 RSU Handler recebeu JOIN\_RESP da RSU 1  
Vehicle ID: b4:8b:d0:e4:41:30 ingressou no grupo da RSU 1  
Vehicle ID: b4:8b:d0:e4:41:30 Lider Sincronizacao Temporal atualizado para RSU 1  
DetectorVeiculos: enviou interesse.  
DetectorVeiculos: detectou veiculo na posicao: (50, 50)  
DetectorVeiculos: detectou veiculo na posicao: (50, 5)  
DetectorVeiculos: enviou interesse.  
DetectorVeiculos: detectou veiculo na posicao: (50, 5)  
DetectorVeiculos: detectou veiculo na posicao: (50, 50)  
DetectorVeiculos: enviou interesse.

# Teste de comunicação em grupos



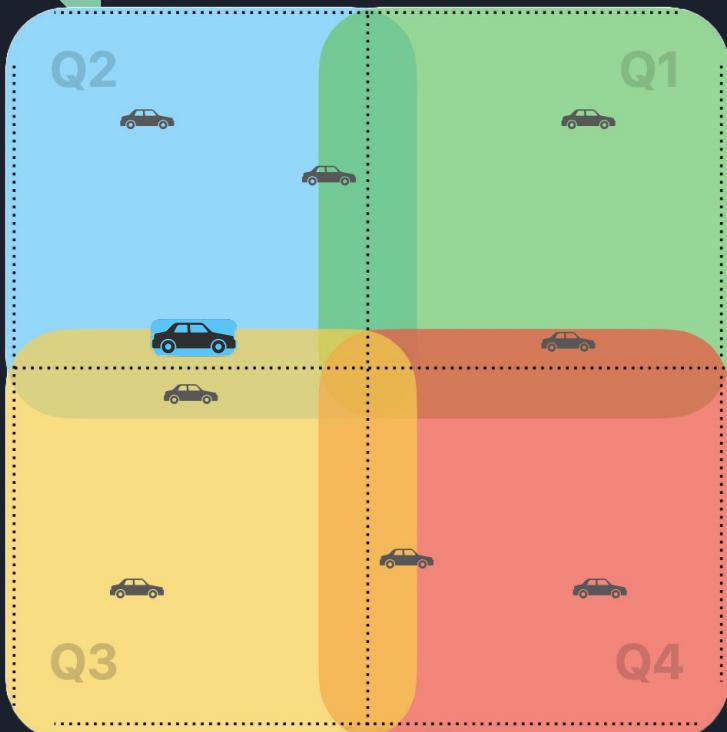
```
Nova Posição Veículo Dinâmico: x: 5, y:50
Vehicle ID: b4:8b:d0:e4:41:30 Veiculo dentro ou proximo ao quadrante do RSU 2, enviando JOIN_REQ.
Vehicle ID: b4:8b:d0:e4:41:30 Veiculo ja enviou JOIN_REQ para o grupo 2, ignorando SYNC.
Vehicle ID: b4:8b:d0:e4:41:30 RSU Handler recebeu JOIN_RESP da RSU 2
Vehicle ID: b4:8b:d0:e4:41:30 vizinho ao grupo da RSU 2
    DetectorVeiculos: detectou veiculo na posicao: (50, 5)
    DetectorVeiculos: detectou veiculo na posicao: (50, 50)
    DetectorVeiculos: enviou interesse.
    DetectorVeiculos: detectou veiculo na posicao: (50, 50)
    DetectorVeiculos: detectou veiculo na posicao: (50, 5)
    DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    DetectorVeiculos: enviou interesse.
    DetectorVeiculos: detectou veiculo na posicao: (50, 50)
    DetectorVeiculos: detectou veiculo na posicao: (50, 5)
    DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    DetectorVeiculos: enviou interesse.
    DetectorVeiculos: detectou veiculo na posicao: (50, 5)
    DetectorVeiculos: detectou veiculo na posicao: (50, 50)
    DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    DetectorVeiculos: enviou interesse.
```

# Teste de comunicação em grupos



Nova Posição Veículo Dinâmico: x: -50, y:50  
Vehicle ID: b4:8b:d0:e4:41:30 Veiculo dentro ou proximo ao quadrante do RSU 2, enviando JOIN\_REQ.  
Vehicle ID: b4:8b:d0:e4:41:30 RSU Handler recebeu JOIN\_RESP da RSU 2  
Vehicle ID: b4:8b:d0:e4:41:30 ingressou no grupo da RSU 2  
Vehicle ID: b4:8b:d0:e4:41:30 Lider Sincronizacao Temporal atualizado para RSU 2  
DetectorVeiculos: detectou veiculo na posicao: (50, 5)  
DetectorVeiculos: detectou veiculo na posicao: (-5, 50)  
DetectorVeiculos: detectou veiculo na posicao: (50, 50)  
DetectorVeiculos: enviou interesse.  
DetectorVeiculos: detectou veiculo na posicao: (-5, 50)  
DetectorVeiculos: detectou veiculo na posicao: (-50, 50)  
DetectorVeiculos: enviou interesse.  
DetectorVeiculos: detectou veiculo na posicao: (-5, 50)  
DetectorVeiculos: detectou veiculo na posicao: (-50, 50)  
DetectorVeiculos: enviou interesse.  
DetectorVeiculos: detectou veiculo na posicao: (-5, 50)  
DetectorVeiculos: detectou veiculo na posicao: (-50, 50)  
DetectorVeiculos: enviou interesse.

# Teste de comunicação em grupos



```
Nova Posição Veículo Dinâmico: x: -50, y:5
Vehicle ID: b4:8b:d0:e4:41:30 Veiculo dentro ou proximo ao quadrante do RSU 3, enviando JOIN_REQ.
Vehicle ID: b4:8b:d0:e4:41:30 RSU Handler recebeu JOIN_RESP da RSU 3
Vehicle ID: b4:8b:d0:e4:41:30 vizinho ao grupo da RSU 3
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, 50)
    🚗 DetectorVeiculos: enviou interesse.
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, -5)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, 50)
    🚗 DetectorVeiculos: enviou interesse.
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, -5)
    🚗 DetectorVeiculos: enviou interesse.
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-5, 50)
    🚗 DetectorVeiculos: detectou veiculo na posicao: (-50, -5)
    🚗 DetectorVeiculos: enviou interesse.
```

\*\*Continua até completar o número de voltas definido.\*\*