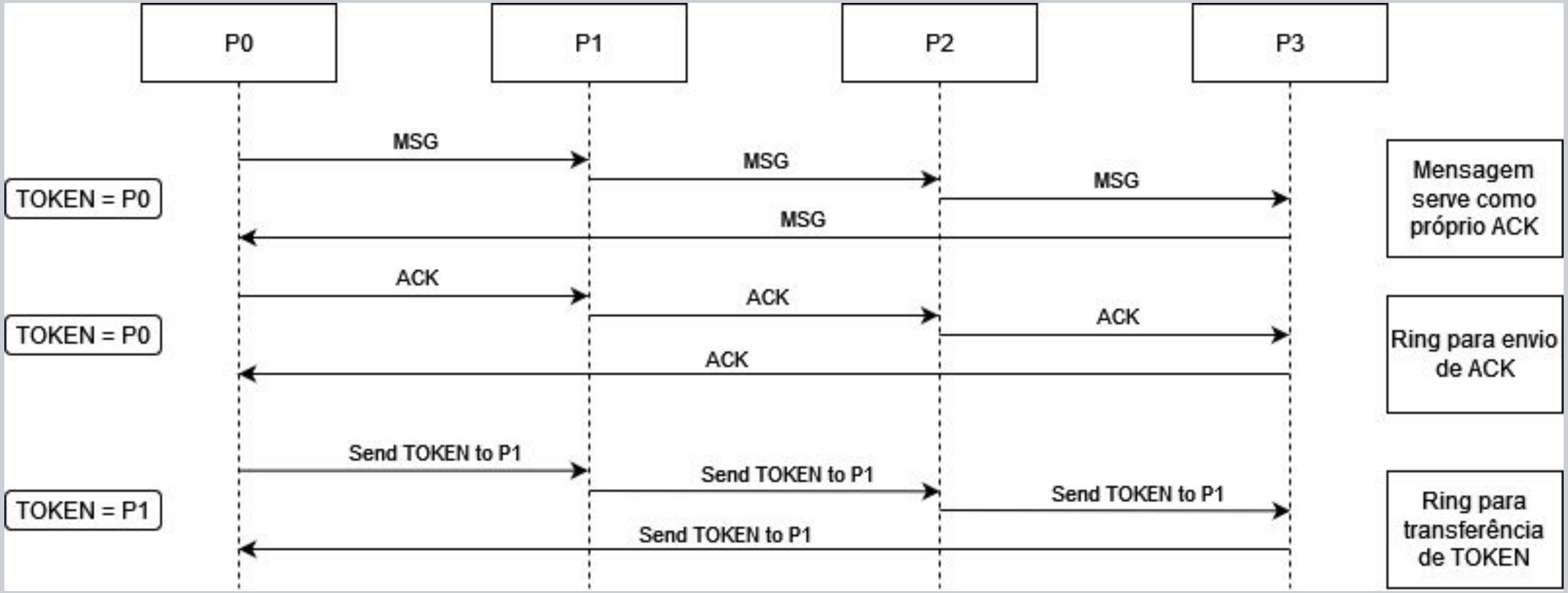
P3 + P4

INE5424-06208B (20242) – Sistemas Operacionais II | Curso de Ciências da Computação

Alexis Mendes Sequeira (16100717)
Luis Henrique Goulart Stemmer (18105165)
Luiz Maurício do Valle Pereira (21104157)
Rafael Neves de Mello Oliveira (17102816)

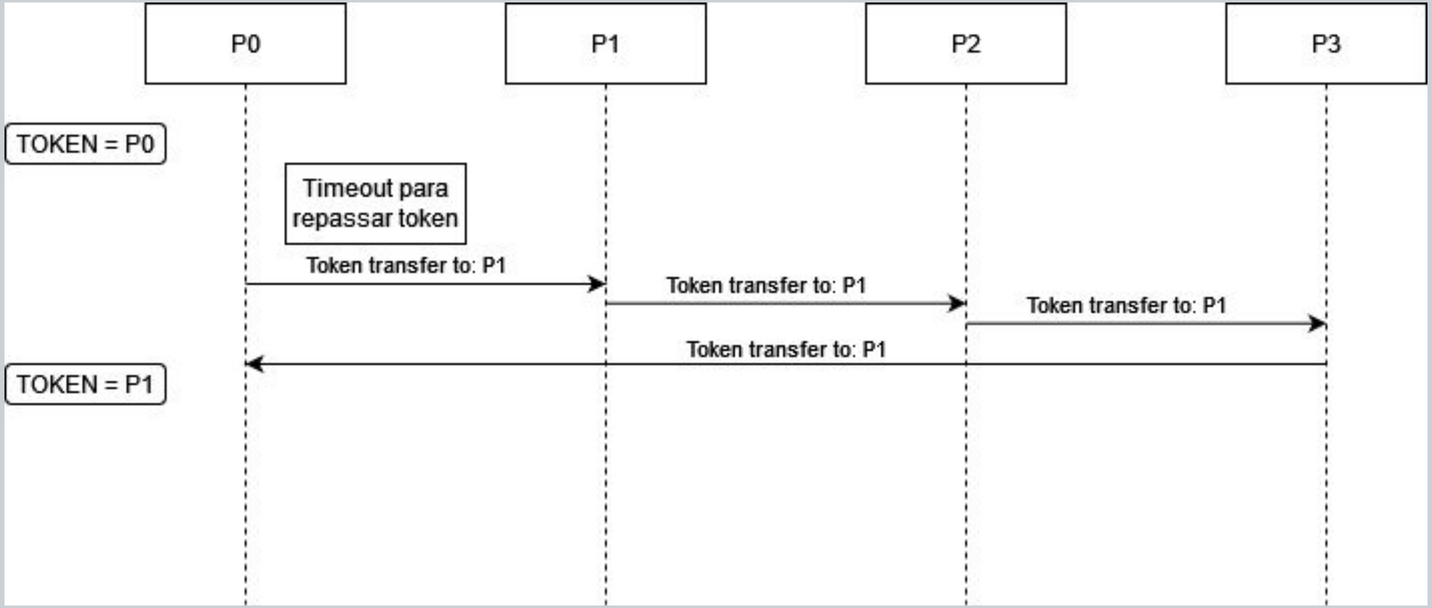
UNIVERSIDADE FEDERAL
DE SANTA CATARINA

P3 - Atomic via Token Ring

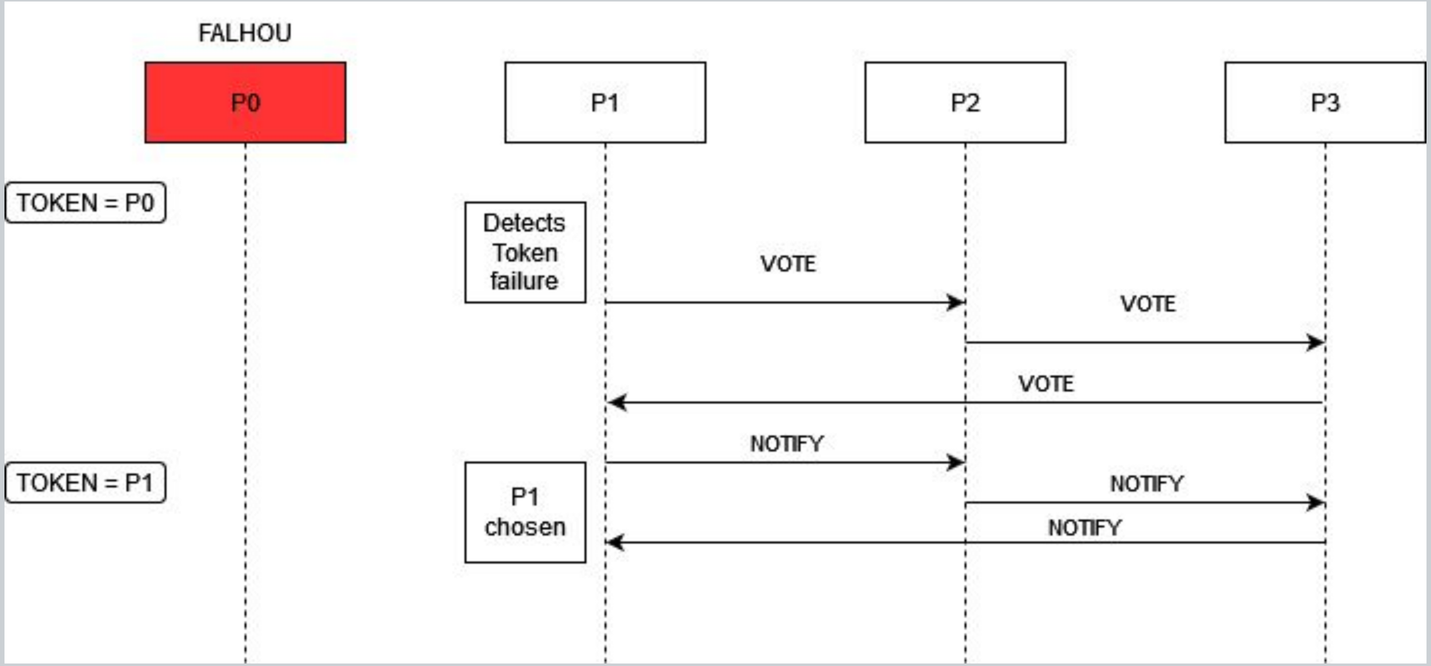


- Todos são notificados quando o token é passado

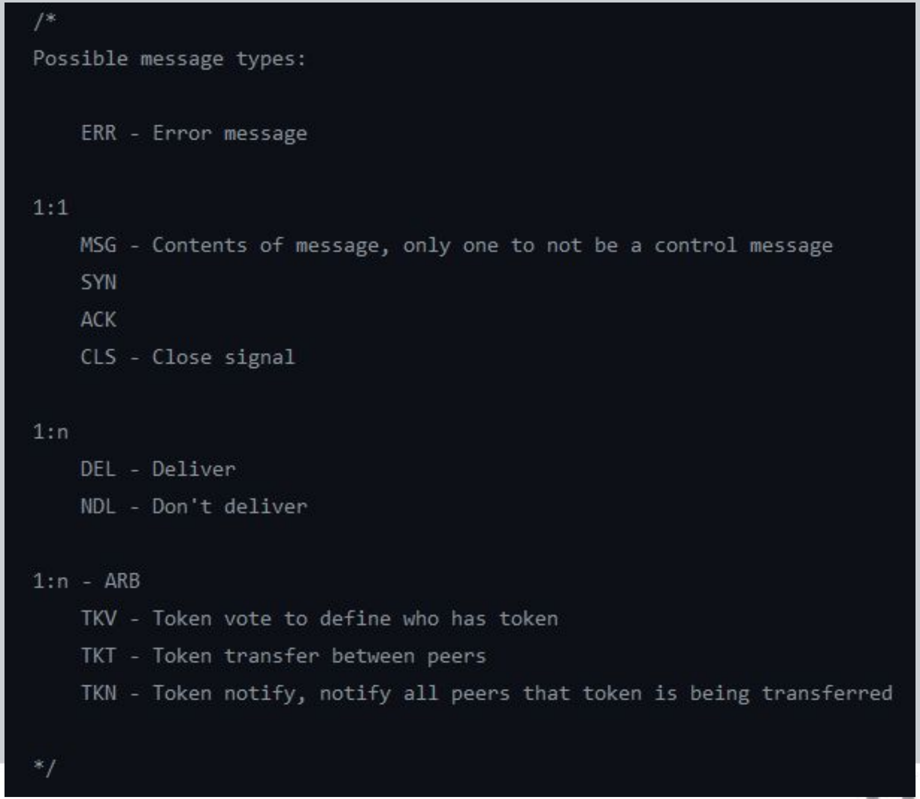
P3 - Timeout para repasse do Token



P3 - Falha do processo Token



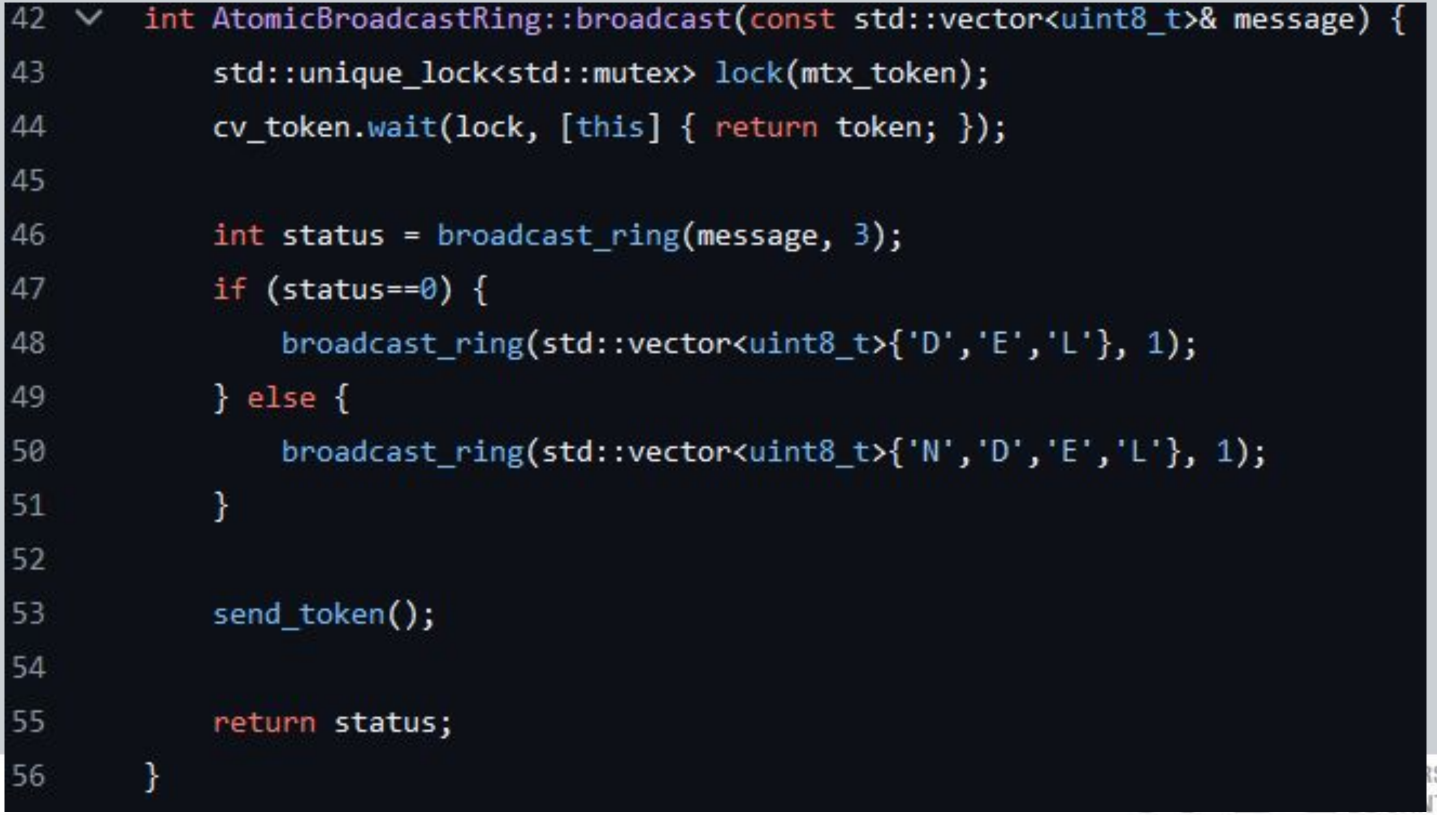
P3 - Novos sinais de Message



```
/*
Possible message types:

    ERR - Error message

1:1
    MSG - Contents of message, only one to not be a control message
    SYN
    ACK
    CLS - Close signal

1:n
    DEL - Deliver
    NDL - Don't deliver

1:n - ARB
    TKV - Token vote to define who has token
    TKT - Token transfer between peers
    TKN - Token notify, notify all peers that token is being transferred

*/
```

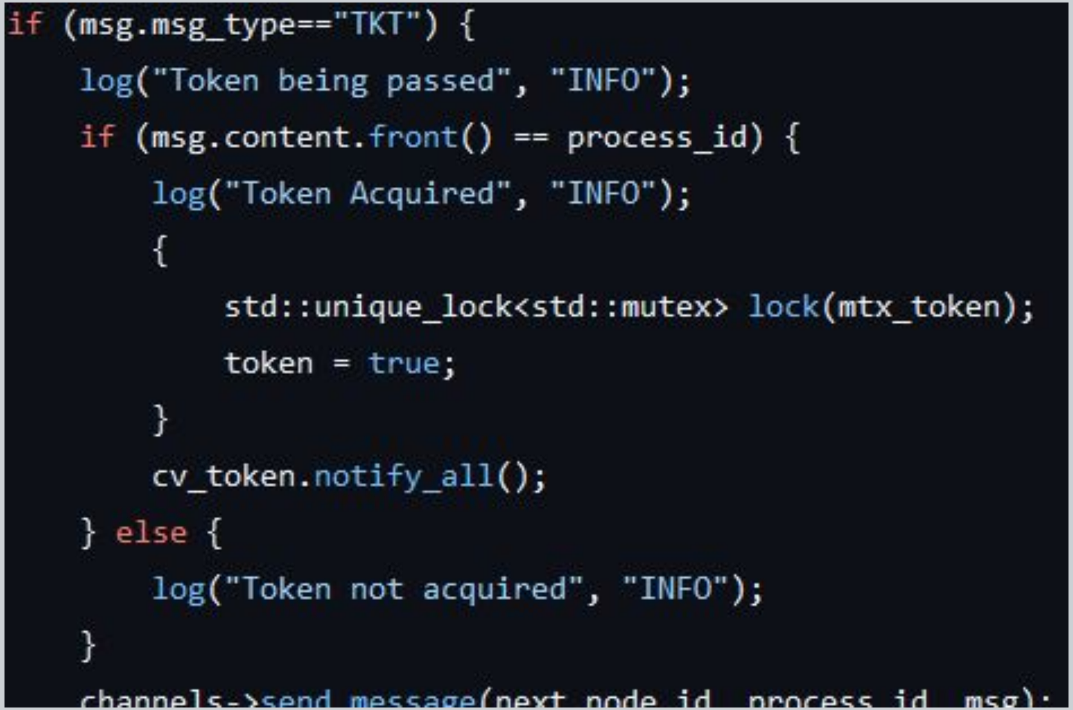
UFSC UNIVERSIDADE FEDERAL DE SANTA CATARINA

P3 - Broadcast



```
int AtomicBroadcastRing::broadcast(const std::vector<uint8_t>& message) {
    std::unique_lock<std::mutex> lock(mtx_token);
    cv_token.wait(lock, [this] { return token; });

    int status = broadcast_ring(message, 3);
    if (status==0) {
        broadcast_ring(std::vector<uint8_t>{'D','E','L'}, 1);
    } else {
        broadcast_ring(std::vector<uint8_t>{'N','D','E','L'}, 1);
    }

    send_token();

    return status;
}
```

P3 - Passando Token



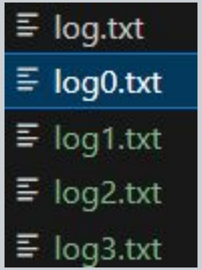
```
if (msg.msg_type=="TKT") {
    log("Token being passed", "INFO");
    if (msg.content.front() == process_id) {
        log("Token Acquired", "INFO");
        {
            std::unique_lock<std::mutex> lock(mtx_token);
            token = true;
        }
        cv_token.notify_all();
    } else {
        log("Token not acquired", "INFO");
    }
    channels->send_message(next_node_id, process_id, msg);
```

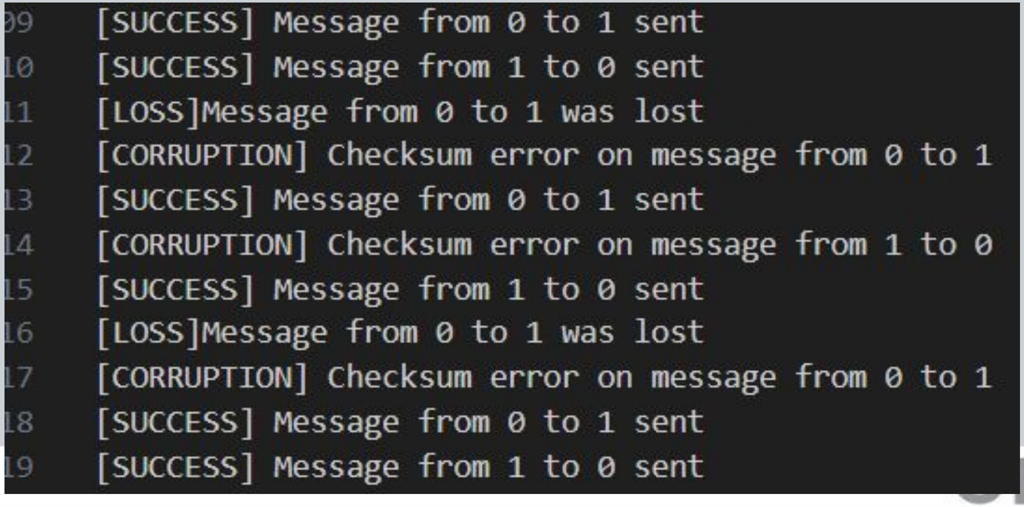
UFSC UNIVERSIDADE FEDERAL DE SANTA CATARINA

P4 - Injeção de Falhas

- Falhas selecionadas:
  - Loss de pacote - Não entrega ou não envio de mensagem
  - Corrupção de dados - Mudança no checksum

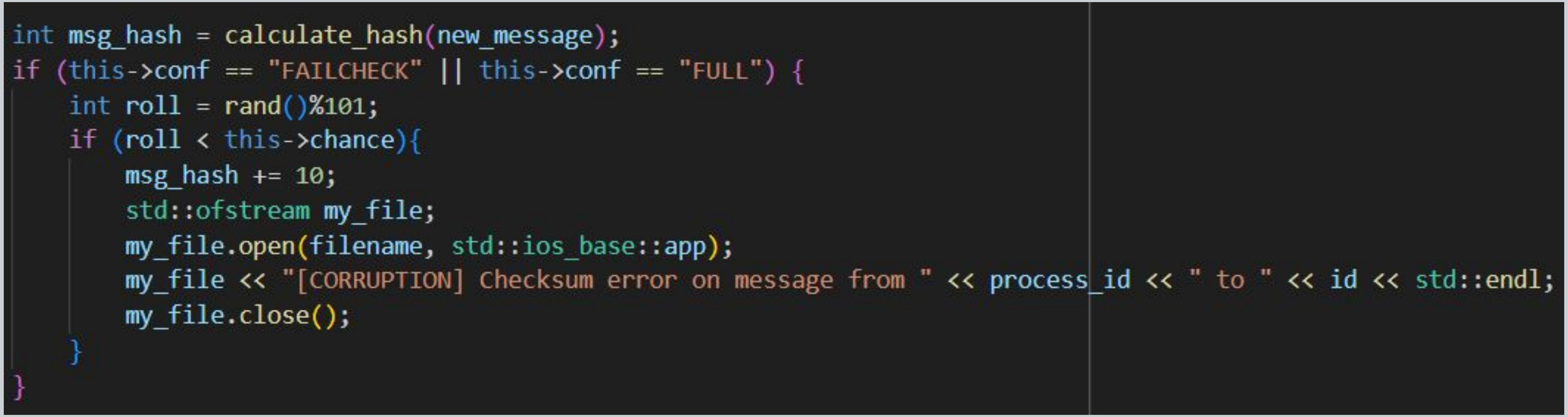
- Geração de arquivos de log via processo para demonstrar as falhas





```
[SUCCESS] Message from 0 to 1 sent
[SUCCESS] Message from 1 to 0 sent
[LOSS]Message from 0 to 1 was lost
[CORRUPTION] Checksum error on message from 0 to 1
[SUCCESS] Message from 0 to 1 sent
[CORRUPTION] Checksum error on message from 1 to 0
[SUCCESS] Message from 1 to 0 sent
[LOSS]Message from 0 to 1 was lost
[CORRUPTION] Checksum error on message from 0 to 1
[SUCCESS] Message from 0 to 1 sent
[SUCCESS] Message from 1 to 0 sent
```

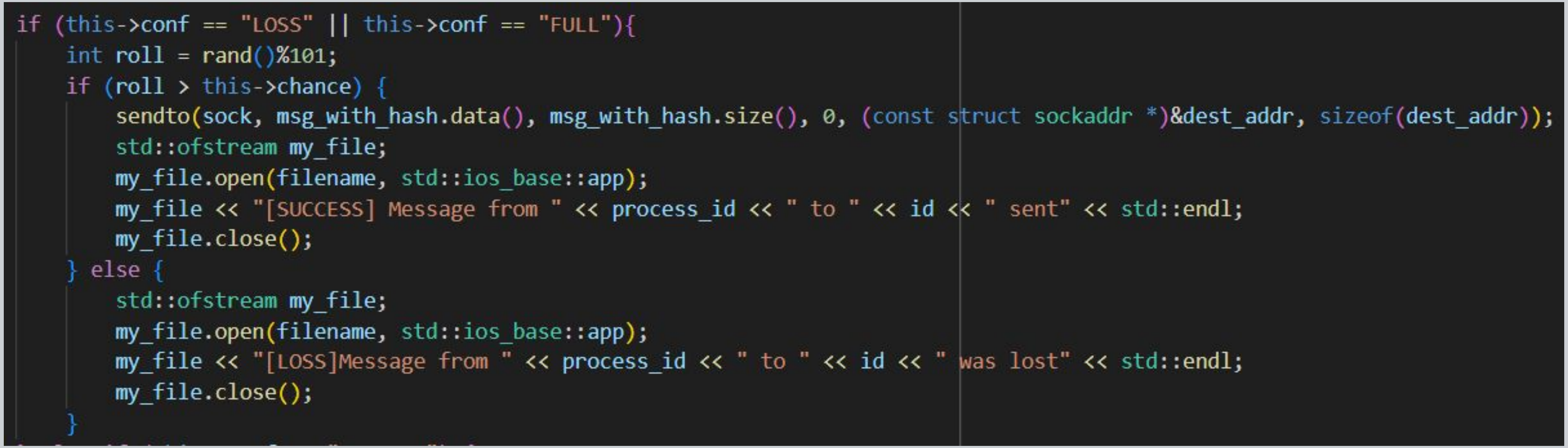
P4 - Código



```cpp
int msg_hash = calculate_hash(new_message);
if (this->conf == "FAILCHECK" || this->conf == "FULL") {
    int roll = rand()%101;
    if (roll < this->chance){
        msg_hash += 10;
        std::ofstream my_file;
        my_file.open(filename, std::ios_base::app);
        my_file << "[CORRUPTION] Checksum error on message from " << process_id << " to " << id << std::endl;
        my_file.close();
    }
}
```

UFSC UNIVERSIDADE FEDERAL DE SANTA CATARINA

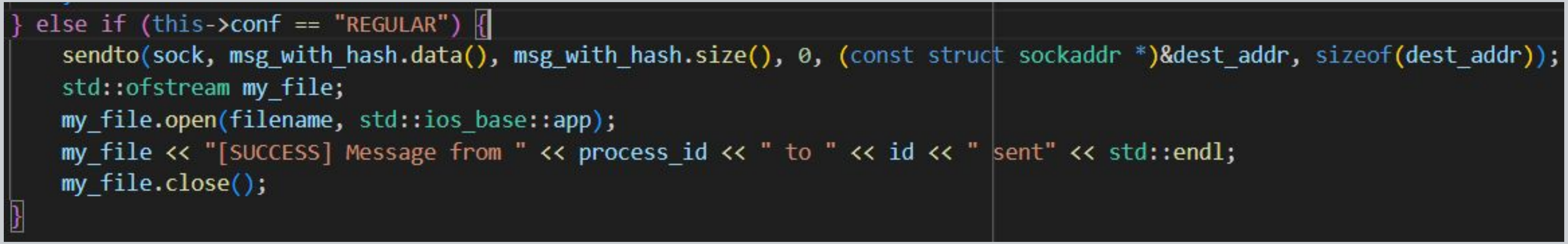
P4 - Código



```
if (this->conf == "LOSS" || this->conf == "FULL"){
    int roll = rand()%101;
    if (roll > this->chance) {
        sendto(sock, msg_with_hash.data(), msg_with_hash.size(), 0, (const struct sockaddr *)&dest_addr, sizeof(dest_addr));
        std::ofstream my_file;
        my_file.open(filename, std::ios_base::app);
        my_file << "[SUCCESS] Message from " << process_id << " to " << id << " sent" << std::endl;
        my_file.close();
    } else {
        std::ofstream my_file;
        my_file.open(filename, std::ios_base::app);
        my_file << "[LOSS]Message from " << process_id << " to " << id << " was lost" << std::endl;
        my_file.close();
    }
```

UFSC UNIVERSIDADE FEDERAL DE SANTA CATARINA

P4 - Código



```
} else if (this->conf == "REGULAR") {
    sendto(sock, msg_with_hash.data(), msg_with_hash.size(), 0, (const struct sockaddr *)&dest_addr, sizeof(dest_addr));
    std::ofstream my_file;
    my_file.open(filename, std::ios_base::app);
    my_file << "[SUCCESS] Message from " << process_id << " to " << id << " sent" << std::endl;
    my_file.close();
}
```

Equipe

Alexis Mendes Sequeira (16100717)

Rafael Neves de Mello Oliveira (17102816)

Luiz Maurício do Valle Pereira (21104157)

Luis Henrique Goulart Stemmer (18105165)

UNIVERSIDADE FEDERAL DE SANTA CATARINA