UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
DEPARTAMENTO DE INFORMÁTICA APLICADA
SISTEMAS OPERACIONAIS II N - INF01151
Prof. Alberto Egon Schaeffer Filho

Daniel Maia - 243672
Denyson Grellert - 243676
Felipe D'Amico - 243677
Filipe Joner - 208840
Rodrigo Dal Ri - 244936

Relatório Trabalho Prático 1

1. Descrição do Ambiente de Teste:
Versão sistema operacional e distribuição: Ubuntu 16.04 LTS
Configurações da Máquina: 4gb RAM, Core i5 3,3ghz.
Compilador: Gcc

2. a) Explique o problema de sincronização de relógios endereçado pelo algoritmo de Cristian e suas limitações.

Algoritmo de Cristian (servidor passivo – centralizado):

Utiliza um servidor para sincronizar computadores externamente, fornecendo sua própria hora local quando solicitado.

- Nova hora do cliente é $T + (T_1 – T_0) / 2$;
- Possui 1 único ponto de falhas;
- Assume que o servidor gasta sempre o mesmo tempo para informar a hora;

b) Como a replicação passiva foi implementada na sua aplicação e quais foram os desafios encontrados.

Não foi implementado a replicação passiva.

c) Explique os aspectos de segurança fornecidos pela nova versão da aplicação, em relação àquela desenvolvida na Parte 1.

Nesta versão da aplicação foi implementada conexão com SSL.

O protocolo SSL provê a privacidade e a integridade de dados entre duas aplicações que comuniquem pela internet. Isso ocorre por intermédio da autenticação das partes envolvidas e da cifragem dos dados transmitidos entre as partes. Ainda, esse protocolo ajuda a prevenir que intermediários entre as duas extremidades das comunicações obtenham acesso indevido ou falsifiquem os dados que estão sendo transmitidos.

d) Estruturas e funções adicionais que você implementou.

Client

Estrutura adicionada para enviar contexto do ssl e socket à thread do cliente e sync.

```
typedef struct arg_threads{
  int socket;
```

```
  SSL *ssl;
} arg_threads;
```

Funções adicionadas:

```
void applySslToSocket(int socket, int isSyncSocket);
void get_server_time();
```

Server

Apenas as primitivas e chamadas de funções da lib openssl foram adicionadas ao servidor, além da initializeSSL().

```
void send_server_time();
```

Util

Funções adicionadas:

```
void initializeSSL();
```

3. Problemas Durante a Implementação

Não foram encontrados grandes problemas durante a implementação. A codificação do SSL apenas se deu pela manutenção das primitivas da biblioteca, removendo reads e wirtes e adicionando SSL_read e SSL_writes, além da aplicação do SSL aos sockets dos clientes.

Já para implementação do relógio, utilizando a biblioteca *time* o algoritmo de Cristian foi realizado de maneira fácil.