Distributed Systems

Secure RPC File System - Design Document

Name: **Prerak Mann** Roll No: **2k17/CO/241**

Name: **Saurabh Mittal** Roll No: **2k17/CO/309**

Introduction

A distributed system is a system whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another.

This is the design for a distributed secure RPC file system. Multiple distributed nodes are available which have access to multiple file servers. Users can communicate with the distributed nodes to manage the file systems securely. Remote Procedure Calls (RPCs) are used for communication between servers and distributed nodes and Symmetric key cryptography is used to secure all the communication.

Problem Statement

The objective is to design a secure file system that allows distributed system(DS) nodes to access the remote files stored on the remote file servers(FS) in a secure manner using Remote Procedure Calls (RPCs).

Requirements -

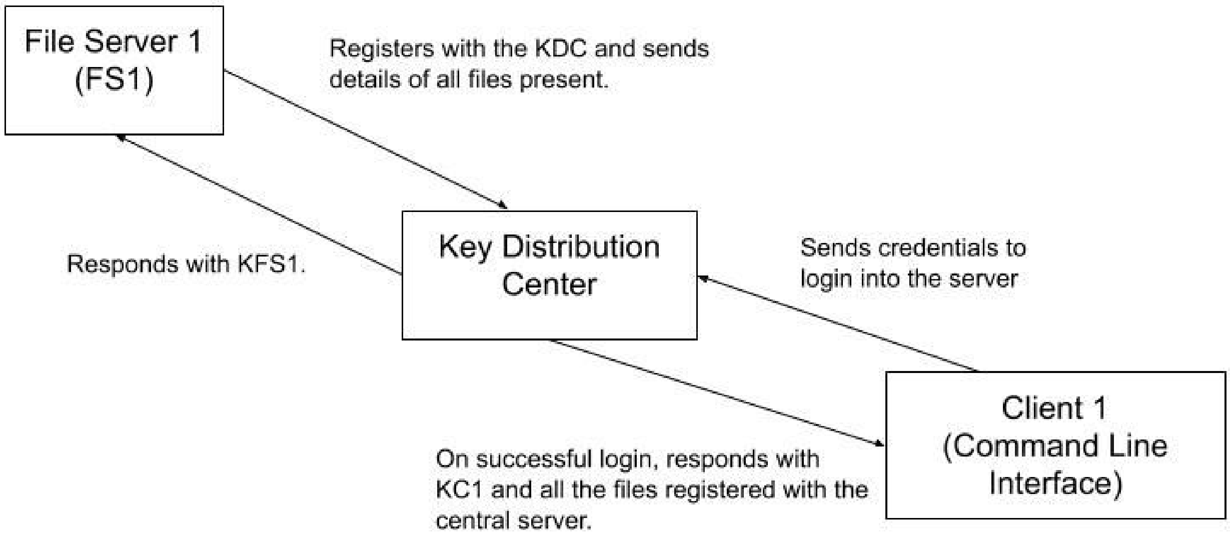
- Symmetric encryption and decryption are to be used for all communication.
- A Key Distribution Center/Server (KDC) handles the generation of session keys and assigning unique ids (UIDs) to servers and distributed nodes. The RPC calls must be encrypted using these session keys.
- File servers register their file content with KDC upon connection.
- Distributed Nodes mount the files to known file system servers initially when connected.
- Any updates and operations performed must be communicated to other Distributed Nodes.
- The users should be able to perform the following operations on files
    - pwd - print working directory
    - ls - list files in a directory
    - cp - copy file in the same folder
    - cat - print contents of the file

Details

1. Components -

- File system Server (1+)
- Distributed System Node (1+)
- Key Distribution Center (1)

2. Initialisation -


File Server 1 (FS1)
Registers with the KDC and sends details of all files present.
Responds with KFS1.
Key Distribution Center
Sends credentials to login into the server
Client 1 (Command Line Interface)
On successful login, responds with KC1 and all the files registered with the central server.

Initial Setup

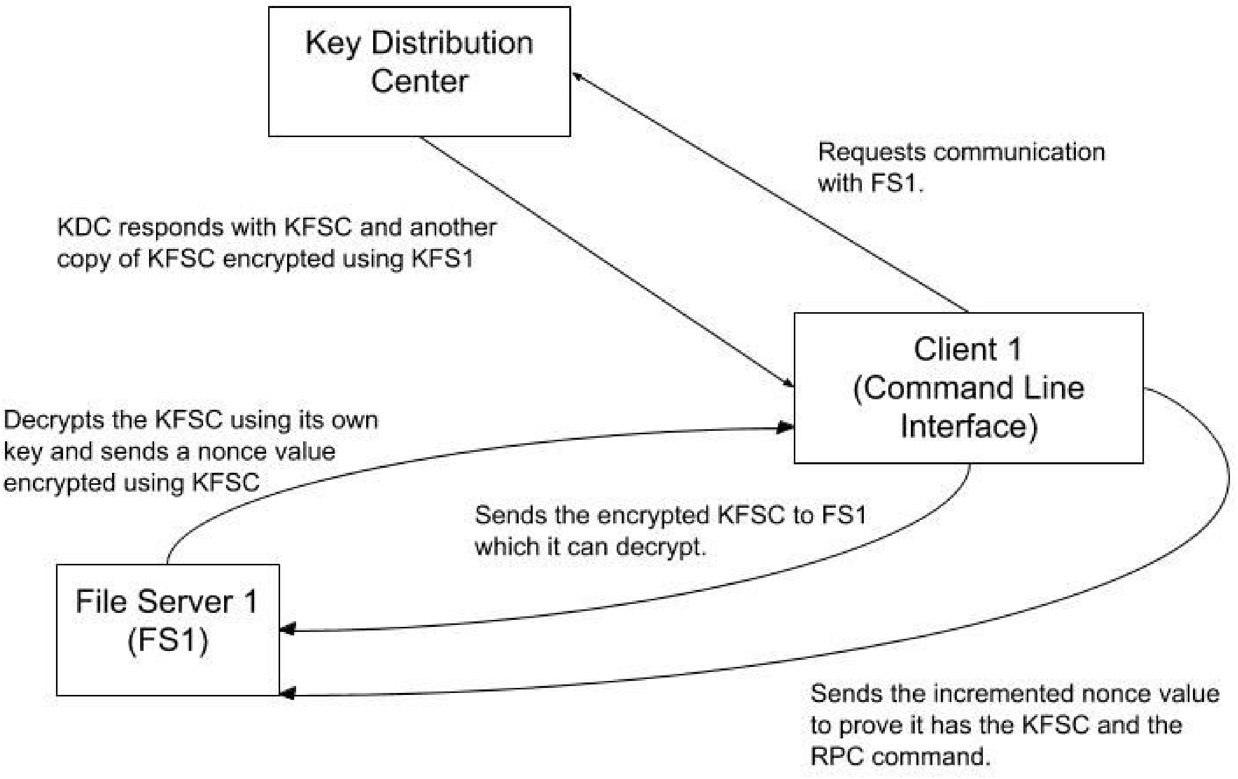
File server -

- Obtain Session Key from KDC and send the information of the file system.

Distributed Node -

- Obtain Session Key from KDC and the entries of the file system (to be displayed on the client’s shell).

3. Communication -


Key Distribution Center
Requests communication with FS1.
KDC responds with KFSC and another copy of KFSC encrypted using KFS1
Client 1 (Command Line Interface)
Decrypts the KFSC using its own key and sends a nonce value encrypted using KFSC
Sends the encrypted KFSC to FS1 which it can decrypt.
File Server 1 (FS1)
Sends the incremented nonce value to prove it has the KFSC and the RPC command.

Communication

Distributed Node -

- Request communication with a FS node to the KDC which responds with the session key and a copy of it encrypted using the key of FS node.
- Sends the encrypted session key to FS and gets a nonce value in return which it decrypts.
- Sends incremented nonce, to prove it has the session key, and the RPC command information.

File server -

- Decrypts the received encrypted session key using its key and sends a nonce value to client encrypted using the session key.

Notations-
KFS1: Symmetric Key of File System 1
KC1: Symmetric Key of Client 1
KFSC: Symmetric Key for both File System and Client generated by KDC or Session Key

Protocols

- Needham–Schroeder Protocol - for the generation of session keys
- Remote Procedure Call (RPC)
- HTTP or TCP/IP for communication between nodes

Platform Used

Python 3 will be used for the implementation of a command-line application for the user to communicate with the distributed system nodes. Also, python would be used to create HTTP/TCP servers for the Key Distribution Center and the file system nodes.