

Projeto #2

**Sistema seguro para acesso e gestão remota de ficheiros,
com segurança de comunicação e garantias de privacidade de operação**

Ver também o enunciado do trabalho

Objetivo

- Conceber e implementar **um sistema seguro para acesso e gestão remota de ficheiros, com segurança de comunicação e garantias de privacidade de operação.**
- O sistema envolve desenhar e implementar os diversos módulos da solução, nomeadamente:
 - **um módulo de controlo de despacho de operações** disponibilizadas ao cliente (através de uma API que pode concretizar um *endpoint* de serviço seguro REST ou um protocolo seguro REQUEST/REPLY suportados em Sockets ou WebSockets TLS)
 - **um módulo de autenticação de utilizadores**
 - **um módulo de controlo de acessos** (ou de gestão de permissões)
 - um módulo de gestão segura de ficheiros com preservação de privacidade do armazenamento.

Arquitetura (Fase 1 e Fase 2)

TLS-PROT-ENF <tls protocol enforced>

TLS-AUTH: <auth-type>

CIPHERSUITES: <csuite options>

Em que <tls protocolo enforced> poderá ser: TLS-1.2 ou TLS-1.1

<auth-type> poderá ser: SERV ou MUTUAL

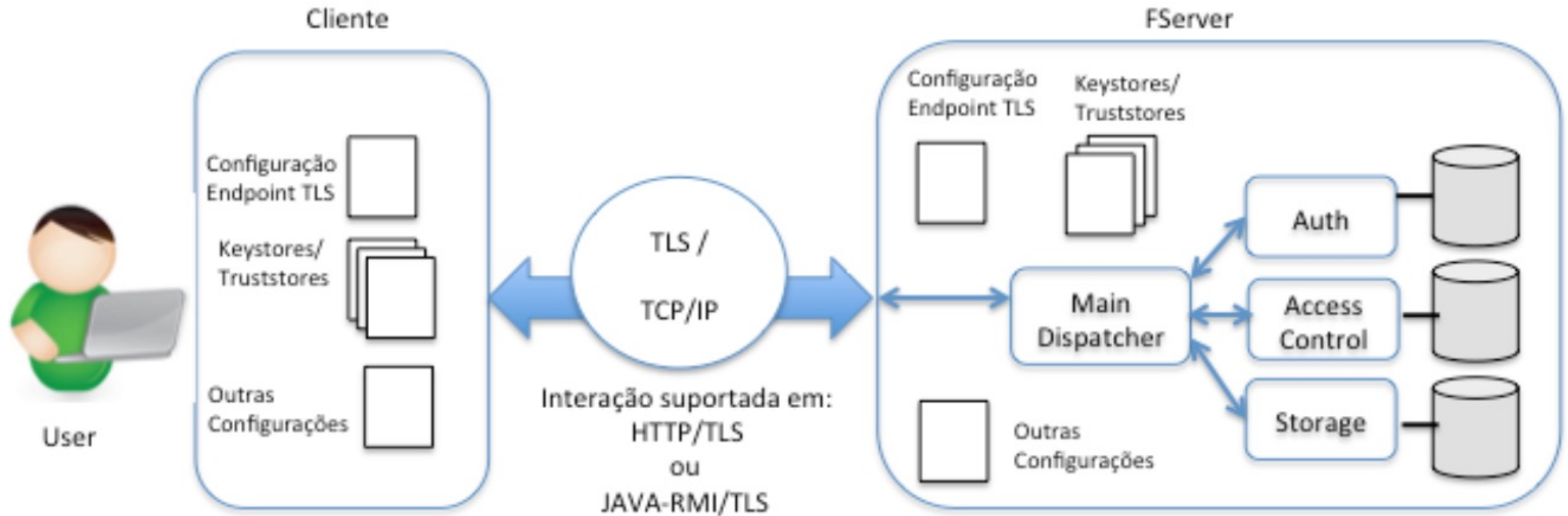
<csuite options> é uma lista de configurações expressadas na forma:

TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384

TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

TLS: TLS 1.2 ou 1.3
↔

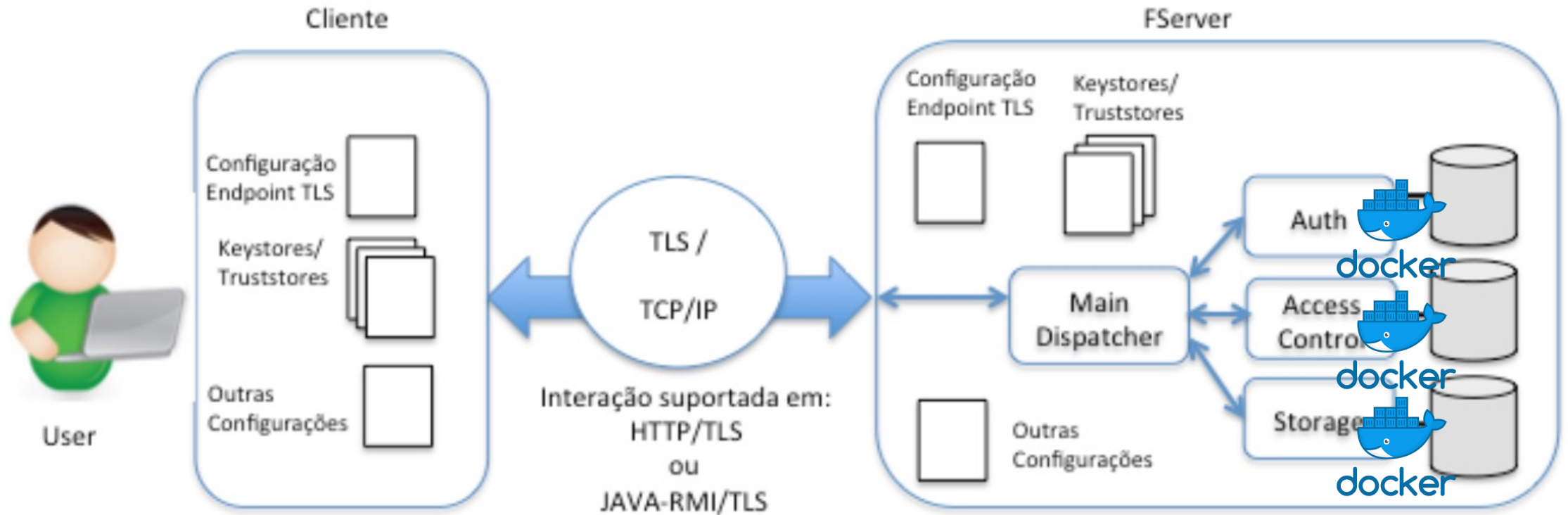


Arquitetura (Fase 1 e Fase 2) c/ Módulos Virtualizados

TLS: TLS 1.2 ou 1.3

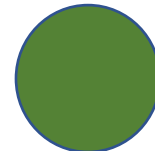
TLS-PROT-ENF <tls protocol enforced>
TLS-AUTH: <auth-type>
CIPHERSUITES: <csuite options>

Em que <tls protocolo enforced> poderá ser: TLS-1.2 ou TLS-1.1
<auth-type> poderá ser: SERV ou MUTUAL
<csuite options> é uma lista de configurações expressadas na forma:
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384



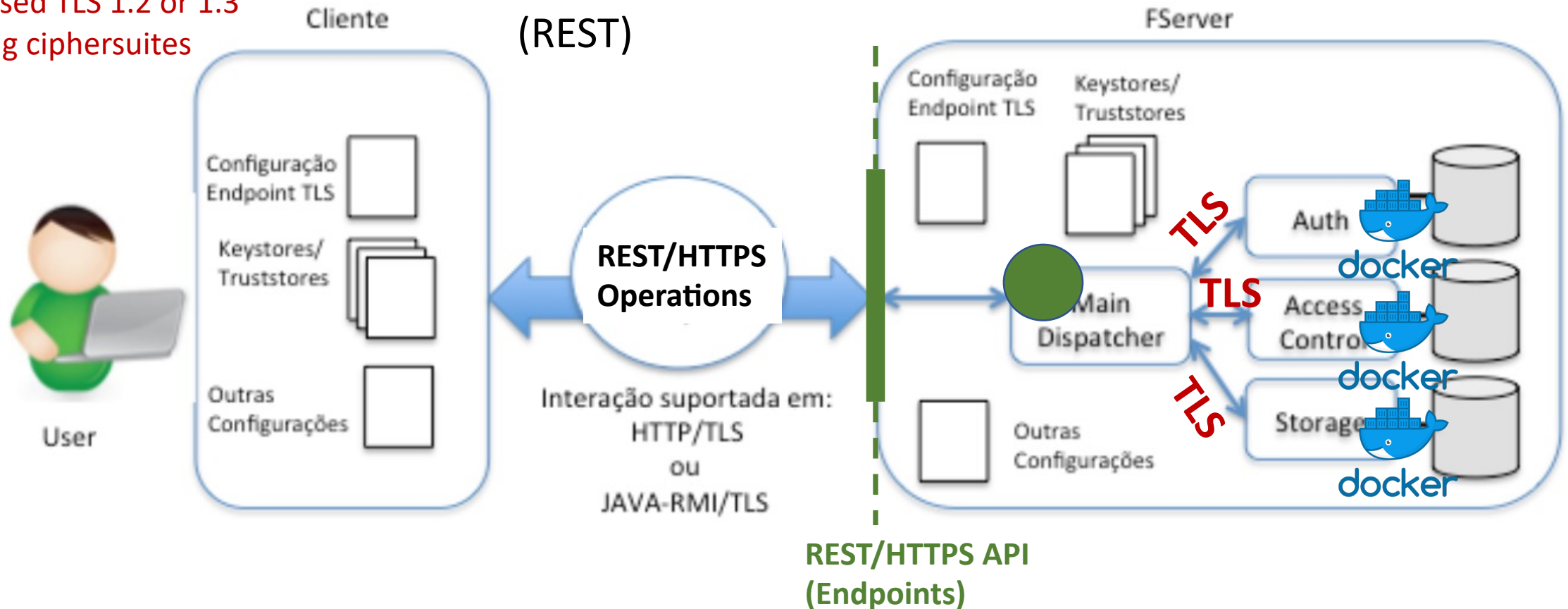
Arquitetura (Fase 1 e Fase 2) c/ Módulos Virtualizados

TLS TLS: TLS 1.2 ou 1.3
Server side vs.
Mutual Authentication
Imposed TLS 1.2 or 1.3
Strong ciphersuites

 **Main Dispatcher as
a TLS enabled
WebService
(REST)**

TLS-PROT-ENF <tls protocol enforced>
TLS-AUTH: <auth-type>
CIPHERSUITES: <csuite options>

Em que <tls protocolo enforced> poderá ser: TLS-1.2 ou TLS-1.1
<auth-type> poderá ser: SERV ou MUTUAL
<csuite options> é uma lista de configurações expressadas na forma:
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384



Key points in the reference architecture

- Separation of concerns in a Distributed Architecture with Decentralized Trust Parameterizations:
 - Single Point of Access (via the Main Dispatcher Service Module)
 - Authentication Service (via the Authentication Service Module)
 - Issues Authentication Proofs for users after successful authentication
 - Access Control Service (via the Access Control Service Module)
 - Issues Access-Contrl Proofs or Capabilities
 - Storage servisse (via the Storage Service Model)
 - Supports data (file) management operatios, acting as a remote file service

Advantages/Limitations of this architecture ?)

Interesting open design options and protocols in the Reference Architecture

- Isolation (confinement of modules)
 - Can be managed in a distributed environment
 - Virtualized modules (what level of virtualization) ?
- Advantages of manageable / configurable TLS endpoints

Interesting open design options and protocols in the Reference Architecture

- Need to start from the reference and design your protocols for implementation, refining your specification and design model for your implementation protocols
 - Authentication Protocol (Client / Dispatcher / Authentication Service)
 - Access Control Protocol (Client / Dispatcher (Access Control Service)
 - Client uses Authentication Credentials (Authentication Tokens, Tickets ...) previously obtained to obtain permission-authorization Tokens, Tickets ... for the required operations, issued by the Access Control Service
 - Storage (Remote File Access) Protocol (Client . Dispatcher / Storage Service)

Funcionalidade (Fase 1)

login username password (obrigatório na fase 1)

ls username (obrigatório na fase 1)

// mostra ficheiros ou directórios do utilizador *username* na sua home-root

ls username path (opcional na fase 1)

// mostra ficheiros ou directórios do utilizador *username* na path indicada, sendo esta
// especificada na forma /a/b/c, a partir do directório home do utilizador.

mkdir username path (opcional na fase 1)

// cria um directório na path indicada

put username path/file (obrigatório fase 1 com um único directório home
do utilizador)

// coloca um ficheiro file na path indicada

get username path/file (obrigatório fase 1 com um único directório home do utilizador)

// obtém ficheiro file na path indicada

Funcionalidade

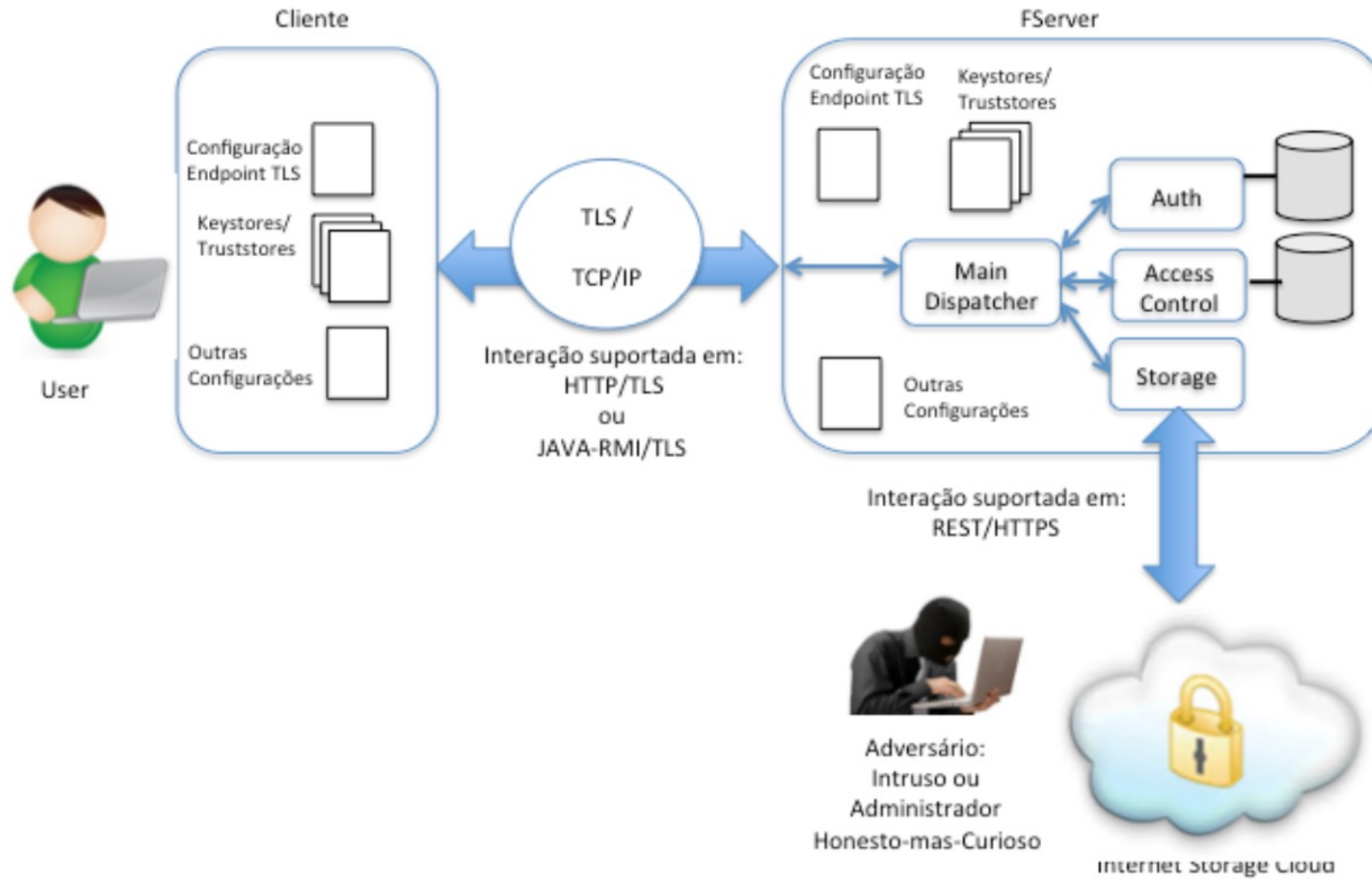
(Fase 1)

	Operação	Obrigatório FASE 1?	Opcional na Fase 1?
Login username password	Login de um utilizador no sistema.	SIM	
ls username	Mostra ficheiros ou directórios do utilizador <i>username</i> na sua <i>home-root</i> no repositório de ficheiros remotos	SIM	
ls username path	Mostra ficheiros ou directórios do utilizador <i>username</i> na <i>path</i> indicada, sendo esta especificada na forma /a/b/c, a partir do directório <i>home</i> do utilizador.	NÃO	SIM
mkdir username path	Cria um directório no caminho <i>path</i> indicado	NÃO	SIM
put username path/file	Coloca um ficheiro <i>file</i> no caminho <i>path</i> indicada	SIM	
get username path/file	Obtém o ficheiro <i>file</i> no caminho <i>path</i>	SIM	
cp username path1/file1 path2/file2 (Copia ficheiro <i>file 1</i> no caminho <i>path 1</i> para <i>file 2</i> no caminho <i>path 2</i>	SIM	
rm username path/file	Remove/Apaga o ficheiro <i>file</i> no caminho <i>path</i>	SIM	
file	Mostra atributos do ficheiro <i>file</i> na <i>path</i> indicada, devolvendo o nome, se é um directório ou se é um ficheiro, o tipo de ficheiro, data da criação, data da última modificação (seguir a semântica do comando <i>file</i> em ambiente Unix)	NÃO	SIM

FASE 2

- Suporte para robustecimento da solução e melhoria das condições de segurança com preservação de privacidade
- Suporte acrescido para preservação de privacidade face adversários do tipo “honestos-mas-curiosos”
 - *HbB (Honest-But-Curious) Adversaries*

Architecture (Fase



Exemplo:
Sysadmins
Operadores com elevados privilégios

Fase 2

Na fase 2, para além da extensão da funcionalidade dos anteriores módulos na passagem da fase 1 para a fase 2, a arquitetura irá incorporar alguns módulos adicionais: um módulo de indexação no cliente que permite que todos os ficheiros sejam armazenados no módulo de Storage de forma a que os ficheiros serão armazenados pelo módulo de storage com as seguintes garantias a fornecer aos clientes:

- Confidencialidade: os ficheiros geridos e armazenados, são mantidos e operados sempre cifrados;
- Integridade: os ficheiros estão armazenados de forma a ficar preservada e detectável a sua integridade;
- Autenticidade: os ficheiros serão armazenados com provas de assinatura digital do utilizador que colocou esses ficheiros no repositório remoto

Other interesting issues

- Can add “search” functionality on encrypted files ?
 - Ex: file result = search pattern path/file
- How to manage encryption keys protecting the stored encrypted files ?
- How to index stored files ?
- How to improve the solution with “secure sharing” in a multiuser environment ?

Sugestões de aspetos de concepção, implementação e teste para a Fase 2

- a) Implementação da funcionalidade (operações) consideradas opcionais para a Fase 1
- b) Todos os ficheiros que ficam no sistema de armazenamento ficam, guardados sempre cifrados (garantindo-se confidencialidade), possuem prova de integridade e estão assinados pelo utilizador que lá os guardou. Isso garante que um adversário do tipo “HbC” não poderá alterar nem quebrar a autenticação do utilizador que lá os colocou nem consegue quebrar a confidencialidade ou integridade dos mesmos
- c) Todos os ficheiros são guardados de modo que a sua dimensão é sempre igual. Isso não permite a um adversário do tipo HbC poder inferir nada sobre o tamanho dos mesmos
- d) Todos os ficheiros, são guardados obedecendo a a), b) e c) e garantindo que os nomes dos ficheiros e diretorias são igualmente cifrados, pelo que um adversário do tipo HbC não poderá inferir nada sobre os nomes e a ligação dos mesmos aos eventuais conteúdos.
- e) Todas as garantias a), b) , c) e d) com os ficheiros guardados particionados (ou fragmentados) em blocos (sempre do mesmo tamanho), sendo o armazenamento operado como se fossem conteúdos de blocos de ficheiros num disco.
- f) Todas as garantias a), b), c) e d), e do lado do servidor um adversário do tipo HbC só vê uma única diretoria que é usada como uma tabela única (Big Table) de blocos. O adversário não consegue inferir nada sobre a estrutura hierárquica que possa estar a ser gerida pelo cliente
- g) Outras garantias que possam ser propostas e implementadas pelos alunos como fatores diferenciadores e valorativos do trabalho em conjugação ou em complementaridade com as anteriores

Interessante em outras projeções da Fase 2

(não para implementação)

- O módulo de armazenamento poderia ser estendido por um módulo proxy (**FServerProxy**) que permitiria ao componente **FServerStorage** usar repositórios de dados *outsourced*
 - *Cloud Storage aaS* (ex., *Dropbox*, *AmazonEC3*, *GoogleDrive*, ou mesmo um ou mais serviço de Cloud Storage) em vez do sistema de ficheiros local usado pelo módulo **FServerStorage**.
 - Poderia evoluir para uma solução Multicloud, com diversidade
 - Interessante: Outsourced Storage sem outsourcing do controlo de segurança e privacidade
 - Interessante: Possível implementação para um ambiente do tipo “Serachable Encrypted File Storage”)
- Para esse efeito, o módulo trataria de replicar e manter os ficheiros em diferente repositórios, aumentando ainda as garantias de disponibilidade e tolerância a falhas do armazenamento.
 - Interessante: projeção possível para uma solução Multicloud Searchable RAID ?

Projeção da Arquitetura (Fase 2):

Multicloud Searchable Encrypted File Storage

