FARSITE: Federated, Available and Reliable Storage for an Incompletely Trusted Environment

Alexandre Zua Caldeira

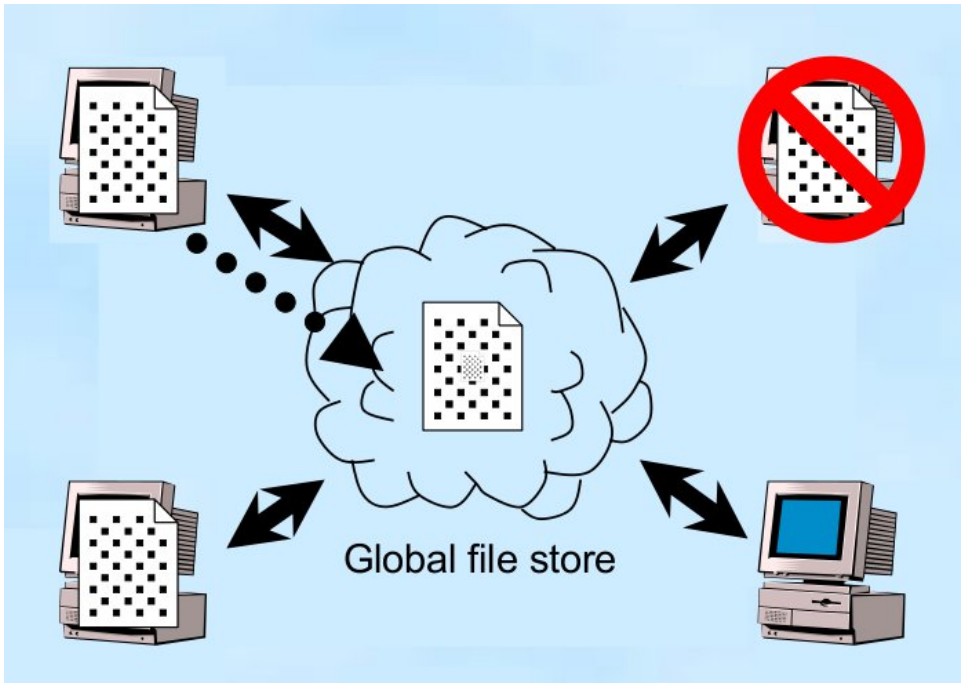
Tecnologias de Middleware 2006/07

17 de Novembro de 2006

- Introdução
- Overview
- Features
- Implementação
- Future Work
- Related Work
- Conclusão

FARSITE

- **FARSITE**
  - *Serverless Distributed File System*
  - **Desenho**: Logicamente centralizado
  - **Implementação**: Distribuída por vários computadores sem garantias de segurança



Porquê sem Servidor?

- *File Server*
  - Precisa de uma equipa para a sua administração ou manutenção
  - Caros
  - Pontos centralizados de falhas do sistema
  - Alvos preferenciais de ataques
- FARSITE vs File Server
  - **Mesmas Funcionalidades**
    - Espaço de nomes partilhado
    - *Write-sharing* para ficheiros e directorias
    - Oferece consistência forte
  - **Melhor Privacidade**: com auxílio de criptografia
  - **Maior Disponibilidade**: com auxílio de **protocolo tolerante a faltas bizantinas**
  - **Mais Barato**

Objectivos

- **Disponibilidade**
- **Fiabilidade**
- **Segurança**
- **Escalabilidade**
- **Performance**
- **Durabilidade**
- **Administração**
  - Esforço de administração apenas durante a configuração initial do sistema
  - Auto administração

Hipóteses de Desenho (1/2)

- Ambiente Alvo: Universidades e grandes empresas, até 10^5 máquinas
- Menor disponibilidade que servidores dedicados[1]
- Falhas permanentes das máquinas são independentes entre si
- Os ficheiros não são acedidos concorrentemente por um grande número de utilizadores
- Uma pequena parte dos utilizadores tem intenções maliciosas de destruir dados
- Uma grande parte dos utilizadores pode, se tiver oportunidade, tentar aceder a dados ou metadados (Directory Service) para os quais não tem permissões de acesso

[1] Assume-se que as máquinas estão em funcionamento a maior parte do tempo

Hipóteses de Desenho (2/2)

- Cada máquina está sobre o controlo de um utilizador, e um grupo de utilizadores maliciosos só pode corromper máquinas sob o controlo dos membro do grupo.
- Pode-se considerar que a máquina de um utilizador é de confiança para esse mesmo utilizador.
- Nenhuma informação sensível do utilizador permanece depois do *logoff*[2]

[2]Provado que é falso a não ser que se usem técnicas especiais como por exemplo crypto-paging

Pressupostos Tecnológicos (1/2)

1. Aumento do espaço livre em disco

Ano	Espaço Livre em Disco
1998	49%
1999	50%
2000	58%

Tabela: Evolução do Espaço Livre (1998 e 2000). Teste realizado na Microsoft

Observação:

- Espaço livre permite o mecanismo de **replicação**

Pressupostos Tecnológicos (2/2)

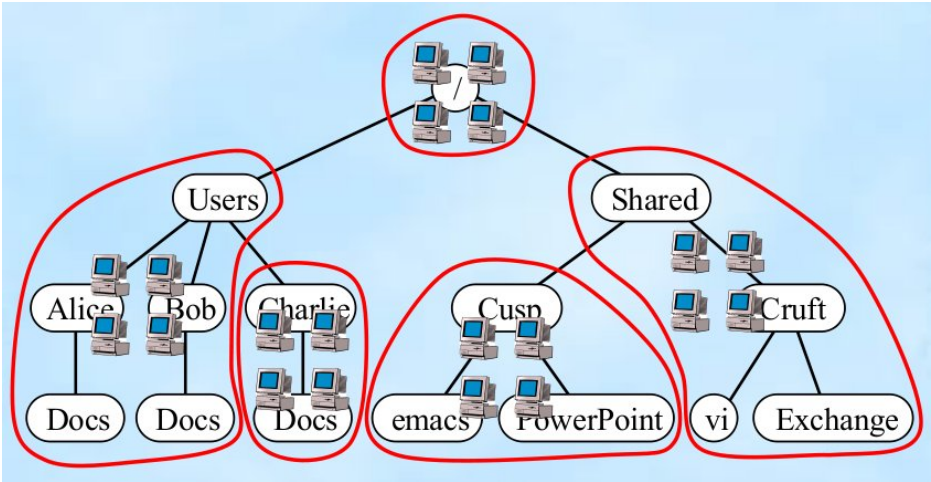
1. Redução do custo computacional das operações criptográficas associado às operações IO

Operação	Largura de Banda (MB/s)
Cifra Simétrica	72
Hashing	53
Disk Sequential IO	32

Tabela: Largura de banda para as operações criptográficas e para o acesso do disco

2. Observações:
   - Cifrar ou decifrar 32KB de dados adiciona aproximadamente 1ms de latência
   - Computar uma assinatura RSA com chave de 1024 bits adiciona 6.5ms (menos que o tempo de 1 rotação de um disco de 7200 rpm)
   - Baixo custo computacional das operações criptográficas em relação às operações de IO

Espaço de Nomes



- Árvore hierarquica de nomes (de ficheiros e directorias)
- Para aumentar a flexibilidade, FARSITE permite várias raízes para o espaço de nomes
- O administrador cria uma raíz do espaço de nomes especificando:
  - Nome único para a raíz do espaço de nomes
  - Máquinas responsáveis por gerir essa raíz (*Directory group*)
- As máquinas que gerem uma raíz do espaço de nomes formam um grupo **tolerante a faltas bizantinas**

Certificação (1/3)

- A gestão da confiança entre máquinas é feita através de **Certificados de Chave Pública**
- Tipos de Certificados no FARSITE:
  - **Certificado de Espaço de Nomes**: associa uma raíz dum EN com o conjunto de máquinas que gerem os metadados dessa raíz
  - **Certificado de Utilizador**: associa um utilizador com a sua chave pública
  - **Certificado de Máquina**: associa uma máquina à sua chave pública

Certificação (2/3)

- CAs e Certificação:
  - A CA gera um par de chaves pública/privada para cada utilizador e assina um certificado associando o username do utilizador à sua chave pública
  - A CA gera pares chaves públicas/privadas para o conjunto de máquinas que gerem um determinado EN e assina um certificado associando às máquinas as respectivas chaves públicas

Certificação (3/3)

- **Instalação dos Certificados**: Para cada máquina na organização
  - A máquina gera um par de chaves pública/privada
  - CA assina um certificado associando a máquina com a chave pública
  - CA instala o certificado da máquina e do espaço de nomes nessa máquina
  - CA indica à máquina que deve aceitar qualquer certificado assinado por si

Certificação (3/3)

- **Armazenamento das Chaves Privadas**:
  - A chave privada de uma máquina é guardada na própria máquina
  - As chaves privadas dos clientes são cifradas com cifra simétrica criada a partir da respectivas passwords e guardadas e acedidas depois do *login* do utilizador
  - As chaves privadas da CA são guardadas offline: todo o sistema depende delas.
- A CA pode revogar os certificados.

Sistema Básico (1/2)

- Cada máquina pode assumir 3 papéis: *Client*, membro de um *Directory Group* ou *File Host*.
- **Client**: Máquina que interage directamente com utilizador
- **Directory Group**: Conjunto de máquinas que geram colectivamente informação sobre ficheiros através de um protocolo de tolerância a faltas bizantinas. Cada membro do grupo guarda uma réplica da informação e, para os pedidos dos clientes, todos os elementos do grupo
  - Processam os pedidos deterministicamente (pela mesma ordem)
  - Actualizam a informação sobre o ficheiro ou directoria
  - Enviam respostas aos pedidos dos clientes

Sistema Básico (2/2)

- Problemas:
  - **Performance**: Todos os acessos ao sistema de ficheiros utilizam operações bizantinas remotas
  - **Segurança 1**: Não matém a **privacidade** dos dados dos utilizadores que têm acesso directo às máquinas dos *directory goups*
  - **Segurança 2**: Não fornece mecanismos de **controlo de acessos**
  - **Escalabilidade**: Os dados estão replicados em todas as máquinas do *directory group*[3].

[3] Consome demasiado espaço

Optimizações ao Sistema Básico (1/3)

- **Performance** com *cache* local e *leases* sobre ficheiros
  - Usa mecanismo de **cache** local do conteúdo dos ficheiros
  - Directory groups lançam **leases** sobre os ficheiros, permitindo aos clientes que detenham um lease válido aceder ao ficheiro durante um determinado período de tempo.
  - **Obs**: Com estes dois mecanismos os acessos podem ser feitos localmente, sem necessidade de troca de mensagens intensivo com o exterior
  - Retarda o envio de actualizações[4]

[4] As escritas num ficheiro são frequentemente apagadas ou reescritas pouco depois de ocorrerem

Optimizações ao Sistema Básico (2/3)

- **Segurança** e **Tolerância a Faltas**
  - Privacidade e Controlo de Acessos: Os dados são cifrados com as chaves públicas dos utilizadores autorizados.
  - Directory Group verifica a autenticidade dos pedidos de escrita de dados antes de processar as actualizações.
  - Grupos Bizantinos falham se $\frac{1}{3} * N + 1$ dos seus elementos falharem. *Directory groups* querem um grande factor de replicação.
  - O grupo distribui o conteúdo dos ficheiros por um número menor de máquinas, os *file hosts*, e guarda um hash seguro desse conteúdo, para não serem corrompidos pelo *file host*
  - Os *file hosts* não realizam operações bizantinas logo o sistema não falha enquanto houver uma máquina a correr[5]

[5] Redução da taxa de replicação

Optimizações ao Sistema Básico (3/3)

- Problema: Introdução de novas máquinas/utilizadores ao sistema aumenta a quantidade de metadados[6]
  - FARSITE permite que um grupo delegue parte do seu EN a outro *directory group*. Como?
    - O grupo selecciona aleatoriamente um conjunto de máquinas para que formem um novo directory group
    - O grupo assina um certificado de EN delegando autoridade ao novo grupo sobre parte do seu espaço de nomes[7]

[6] Pode ultrapassar a capacidade de processamento do grupo

[7] uma sub-árvore da sua sub-árvore

Fiabilidade e Disponibilidade (1/2)

- Garantidos através da replicação dos dados
  - Metadados do serviço de directórios são replicados nos *directory groups* com protocolos tolerantes a faltas bizantinas.
  - Conteúdo dos ficheiros são replicados nos *file hosts* com protocolos de replicação tradicionais
- Com este modelo de replicação
  - Os dados nos *directory groups* estão acessíveis desde que $\frac{1}{3} * R_D - 1$ membros do grupo não falhem
  - Os dados nos *file hosts* estão acessíveis enquanto houver uma máquina em funcionamento

Fiabilidade e Disponibilidade (2/2)

- As funções das máquinas que fiquem indisponíveis migram para outras máquinas
  - Migração das funções dum membro de *directory groups*
    - As máquinas que mais frequentemente vão ver os seus dados migraos são as que têm menor índice de disponibilidade. Então o sistema faz convergir os membros de directory groups para máquinas com maior disponibilidade.
  - Migração de *file host*
    - Evidência: As máquinas com baixa dispobilidade têm mais impacto no sistema do que as máquinas com alta disponibilidade.
    - A disponibilidade é maximizada se as réplicas estiverem bem distribuídas entre as máquinas com maior e menor disponibilidade.
    - FARSITE utiliza um mecanismo de swap de réplicas entre máquinas de menor e maior disponibilidade
    - Experiência mostra que, por dia, apenas é feito swap a 1% das réplicas.

Controlo de Acessos

- Os metadados, além dos nomes dos ficheiros e directorias, contém também uma lista de controlo de acessos (ACL) com as chaves públicas dos clientes autorizados a aceder à informação no *directory group*
- Os clientes assinam os pedidos com a sua chave privada.
- Os membros do directory group autenticam o cliente antes de conceder o acesso.

Privacidade

- O conteúdo dos ficheiros e os metadados são ambos cifrados para garantir a sua privacidade. Ao criar um ficheiro, o cliente:
- *Convergent Encryption*
  - O mesmo texto cifrado por utilizadores diferentes converge para a mesma cifra
  - Usado para detectar ficheiros duplicados
- *Exclusive Encryption*
  - Reforço do sistema criptográfico para garantir que o resultado da *decryption* é sempre um nome válido
  - Previne de um ataque em que o atacante cifra um nome inválido

Integridade

- Metadados: garantida pelo protocolo tolerante a faltas bizantinas
- Dados: integridade garantida usando uma *Merkle hash tree*[8] sobre os blocos dos ficheiros
  - Criação da árvore
  - Armazenamento da árvore juntamente com o ficheiro na máquina
  - Armazena uma cópia do *hash* da raíz da árvore no *directory group*

[8] Caracteísticas???

Durabilidade dos Metadados(1/2)

- Log
  - Operações de criar, apagar, renomear ficheiros ou directorias são operações apenas sobre os metadados
  - Quando uma aplicação efectua uma destas operações a actualização é feita apenas localmente e não através de uma operação bizantina (custos elevados)
  - Os updates são registados num log.
  - O log é enviado para o *directory group* periodicamente, ou quando um *lease* for libertado
  - Os directory groups verificam a validade e autenticação de cada entrada no log antes de procederem às actualizações

Durabilidade dos Metadados (2/2)

- Quando o cliente contacta pela primeira vez o *directory group*:
  - Cliente gera um *authenticator key*
  - Divide essa chave em *secret shares* que distribui aos membros do *directory group*
  - A chave não é guardada na máquina do cliente, logo um intruso não pode utilizá-la durante um crash
  - O cliente assina as actualizações com esta chave criando MACs
- CRASH!!! E houve *commits* de *updates* locais que não foram comunicados ao DG.
- Após a reinicialização[9]:
  - Cliente envia MAC e updates realizados localmente, numa única transação.
  - Membros do DG primeiro obtém os updates, reconstroem conjuntamente a chave, validam os updates e descartam a chave
  - Não são aceites mais updates desde o momento em que a chave é reconstruída

Durabilidade do Conteúdo dos ficheiros

- Modificar um ficheiro afecta tanto os metadados como o conteúdo do ficheiro.
- É necessário que os metadados e o conteúdo sejam actualizados atomicamente.
- Se um bloco de um ficheiro for reescrito, o novo conteúdo é registado no log juntamente com os metadados.
- Se um cliente concatena novos conteúdos no fim de um ficheiro, apenas se actualiza atomicamente o tamanho do ficheiro e o seu *hash*.

Consistência

- A responsabilidade de manter a coerência dos dados pertence aos directory groups.
  - Os DG delegam algum controlo aos clientes através da atribuição de *leases* sobre os ficheiros
  - Existem 4 classes de leases no FARSITE
    - **content leases**
    - **name leases**
    - **mode leases**
    - **access leases**

Leases sobre o Conteúdo dos Ficheiros

- Existe, 2 tipos de *Content Leases*:
  - Read/write Leases
  - Read-Only Leases
- *Content Leases* podem ter várias granularidades: sobre ficheiro ou sibre uma directoria
- *Leases* são válidos apenas temporariamente. Caso contrário, a falha de uma máquina poderia tornar um ficheiro indisponível porque o *lease* nunca devolvido ao DG
- Para os DG a expiração de um *lease* é idêntico ao fechar de um ficheiro, sem realizar mais nenhuma actualização
- Por questões de performance o número de *leases* lançados sobre um ficheiro é limitado

Leases sobre o Espaço de Nomes

- Os **name leases** definem qual o cliente que está com controlo sobre um ficheiro.
- 1 tipo 2 semânticas:
  - Se o nome (de ficheiro ou directoria) não existe então o name lease autoriza o cliente a criar um ficheiro ou directoria com esse nome
  - Se existe uma directoria com esse nome então o lease autoriza o cliente a criar ficheiros ou subdirectorias nessa directoria
  - A dupla semântica permite ao cliente criar uma directoria e de seguida criar subdirectorias ou ficheiros

Suporte à Semântica de Partilha do Windows

- Windows suporta consistência ao nível aplicacional, através de modos de acesso e de partilha
- **Modos de Acesso**: *read access, write access, delete access*
- **Modos de Partilha**: *read sharing, write sharing, delete sharing*. Permitem que outras aplicações possam abrir os ficheiros para leitura, escrita ou remoção (delete access)
- Para suportar esta semântica FARSITE utiliza 6 tipos de *mode leases*:
  - **read mode lease, write mode lease, delete mode lease, exclude-read mode lease, exclude-write mode lease e exclude-delete mode lease**
  - Se uma aplicação abre um ficheiro para acesso de leitura então o sistema atribui-lhe um *read mode lease*.
  - Se além disso desejar que a partilha com outras aplicações seja apenas de leitura, então é-lhe atribuído também um exclude-write mode lease e um exclude-delete mode lease.
- Conflitos possíveis: read e exclude-read. O DG deve resolver

Suporte à Semântica de Remoção do Windows

- O Windows tem uma semântica de remoção complexa.
  - Para apagar um ficheiro: (1º) abrir o ficheiro, (2º) Marcar o ficheiro para remoção e (3º) fechar o ficheiro.
  - Quando um ficheiro for marcado para remoção, nenhum novo handle é disponibilizado para manipular o ficheiro.
  - Se existir alguma aplicação que tenha um handle com permissão de remoção, essa aplicação pode remover a marca de remoção, e desse modo, permitir que novos *handles* sejam atribuídas a outras aplicações.
- Para suportar esta semântica FARSITE utiliza 3 tipos de **access lease**
  - **Public** access lease: Indica que o detentor do *lease* tem o ficheiro aberto.
  - **Protected** access lease: Public access lease + não será concedido outro lease sem antes contactar o detentor do lease.
  - **Private** access lease: Protected access lease + nunhum outro cliente pode ter leases de acesso sobre o ficheiro

Escalabilidade (1/3)

- FARISTE utiliza 2 mecanismos para evitar que a computação, armazenamento e comunicação cresça com o tamanho do sistema:
  - Hint-based pathname translation: consiste em determinar qual o DG responsável por um determinado nome.
  - Notificação retardada de mudanças nas directorias.

Escalabilidade (2/3)

- **Hint-based pathname translation**.
  - Cada cliente usa uma cache de nomes e o respectivo DG.
  - Encontra na cache a entrada do maior prefixo.
  - O cliente contacta o respectivo DG
    - O mais provável é que o grupo contactado gira o nome
    - Se o grupo gere apenas um prefixo do pathname então responde com todos os certificados delegados a outros grupos e o cliente guarda estes certificados na *hint* cache.
    - Se o grupo não gere um prefixo do nome então informa o cliente que retira a sua entrada da cache.
  - O processo é repetido até que se encontre o grupo responsável pelo nome.
  - Não são necessárias operações Bizantinas porque os certificados de delegação são seguros

Escalabilidade (3/3)

- **Directory-Change Notification**
  - As aplicações podem registar-se (através de *callbacks*) para serem avisadas de alterações numa determinada directoria.
  - O DG difunde retardadamente as notificações de alteração na directoria.

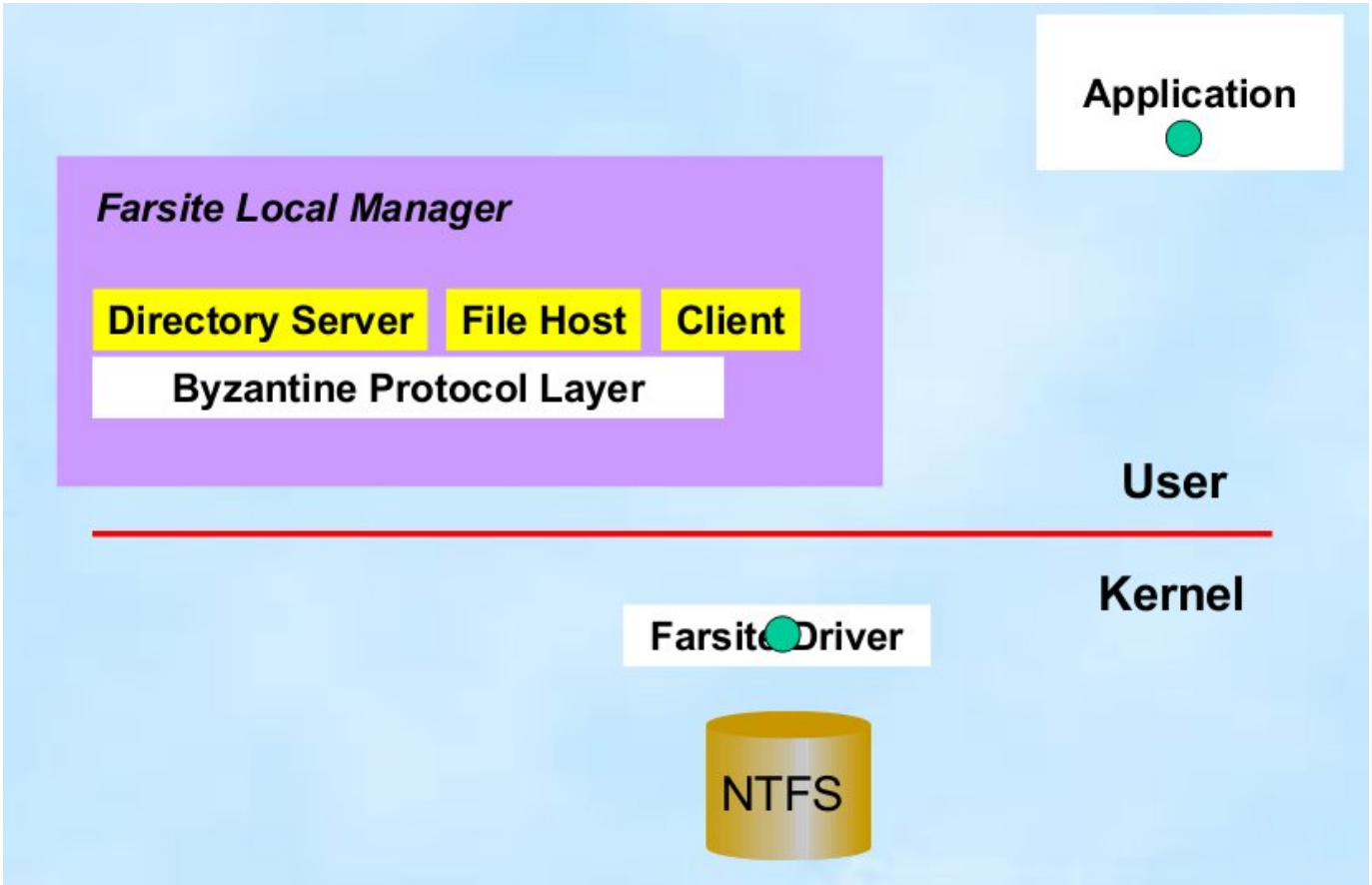
Eficiência no aproveitamento do espaço livre

- A replicação requer espaço disponível em disco
- Testes realizados na Microsoft mostram que cerca de $\frac{1}{2}$ pode ser recuperado apenas pela eliminação dos ficheiros duplicados incidentalmente, por exemplo para serem partilhado por aplicações.
- FARSITE usa uma base de dados distribuída, escalável e tolerante a faltas para descobrir os duplicados.
- As duas cópias são colocadas no mesmo *file host*. *Window's Single Instance Store* trata do resto.

Administração do Sistema

- Actividades de administração locais pretendidas pelos utilizadores
  - Substituição do Disco ou upgrade do hardware
    - A fiabilidade pode ser melhorada.
    - FARSITE permite que utlizador indique que tenciona desligar um disco ou uma máquina.
    - As funções da máquina migram para outras máquinas.
  - Upgrade do FARSITE ou *bug fixes*
    - FARSITE mantém a **compatibilidade** entre versões
    - Importante para a **interoperabilidade** entre máquinas que correm versões diferentes do FARSITE.
- Nao são necessárias outras tarefas administrativas além da configuração inicial do sistema (criação e instalação de certificados)

Componentes



- Componentes do FARSITE
  - FLM: *FARSITE Mini Redirector* (Driver ao nível do kernel)
  - FMR: *FARSITE Local Manager*

Performance

- FARSITE vs NTFS (sistema de ficheiros local) e CIFS(servidor de ficheiros centralizado):

Operação	FARSITE/NTFS	FARSITE/CIFS
Open	9	-
Close	2-4	-
Read	2-4	0.4
Write	2-4	2
Querys (ls)	20	0.7
Total	5.5	0.8

- **FARSITE**: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment
  Atul Adya, William J. Bolosky, Miguel Casto, Gerald Cermak, Ronie Chaiken, John R. Douceur, John Howell, Jacob R. Lorch, Marvin Theimer, Roger P. Wattenhofer
  Microsoft Research, Redmond, WA 98052
- **Distributed Systems for System Architects**
  Paulo Veríssimo, Luís Rodrigues
  Kluwer Academic Publishers
- www.cs.berkeley.edu/ kubitron/courses/cs294-4-F03/slides/lec15-farsite.ppt
  Atul Adya, William J. Bolosky, Miguel Casto, Gerald Cermak, Ronie Chaiken, John R. Douceur, John Howell, Jacob R. Lorch, Marvin Theimer, Roger P. Wattenhofer
  Microsoft Research, Redmond, WA 98052