Go FUSE and store cookies in database

VISIONECT

Mihael Dimec

Joan and the need to scale

Joan and the need to scale

- Joan is digital door labeling and room scheduling solution.
- Device is an E Ink(R) e-paper display and WiFi connected thin client solution.

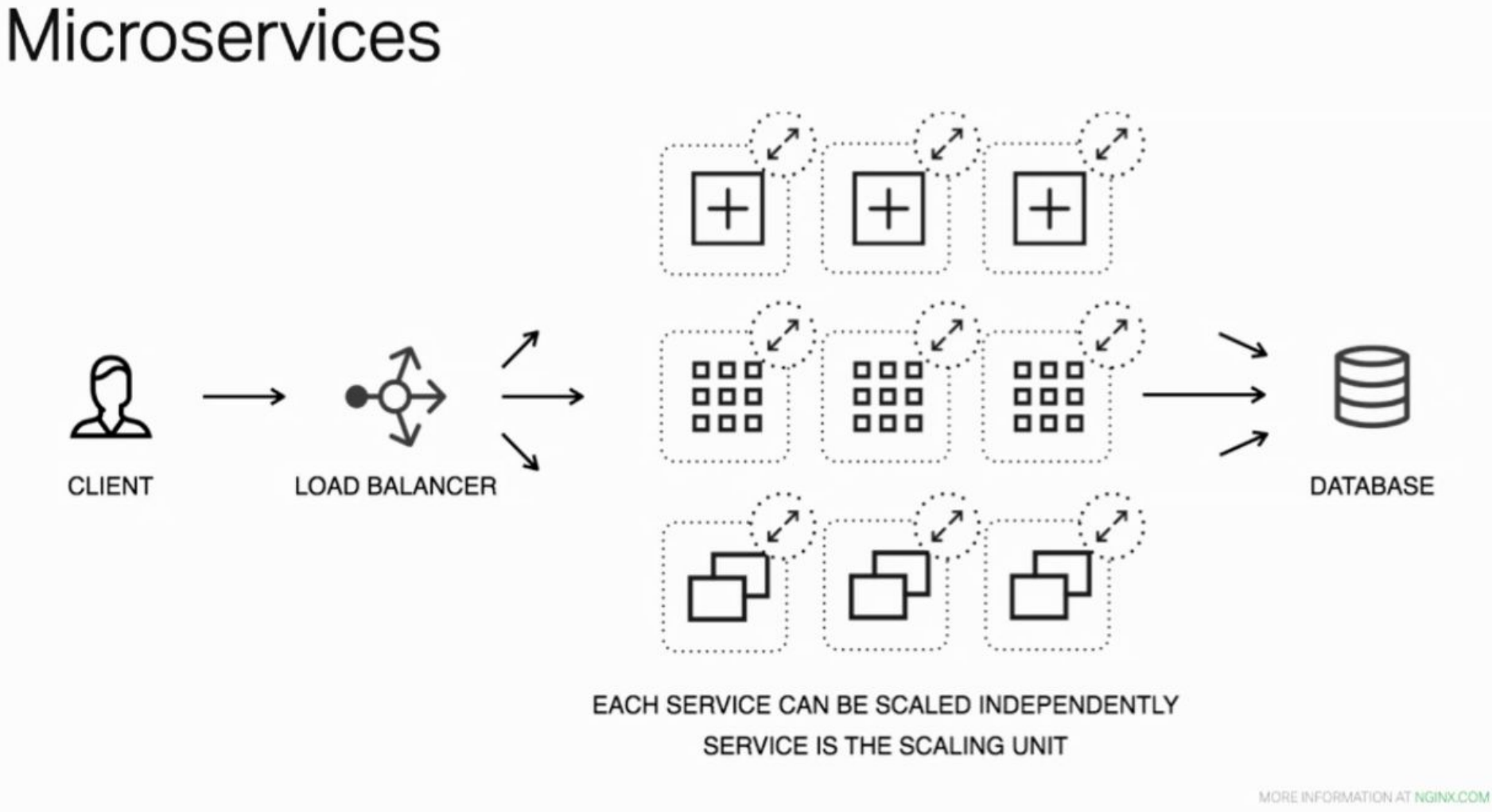
Joan and the need to scale

- Joan is actually web application supported by “Visionect server” WebKit instances.
- Joan app. relies on cookies. WebKit back-end stores persistent cookies trough file-system only.
  - Customers may choose to host in the cloud. which presents for us a scaling exercise.

Joan and the need to scale

- V-server was (mostly) architectured with scalability in mind. But until now it was used as monolith until now.
- Problem: Load balancer can migrate web sessions between different servers. → Cookies need global storage.
- Solution: We are already using database to store such data. → Implement FUSE interface between database and WebKit, that each server can mount.

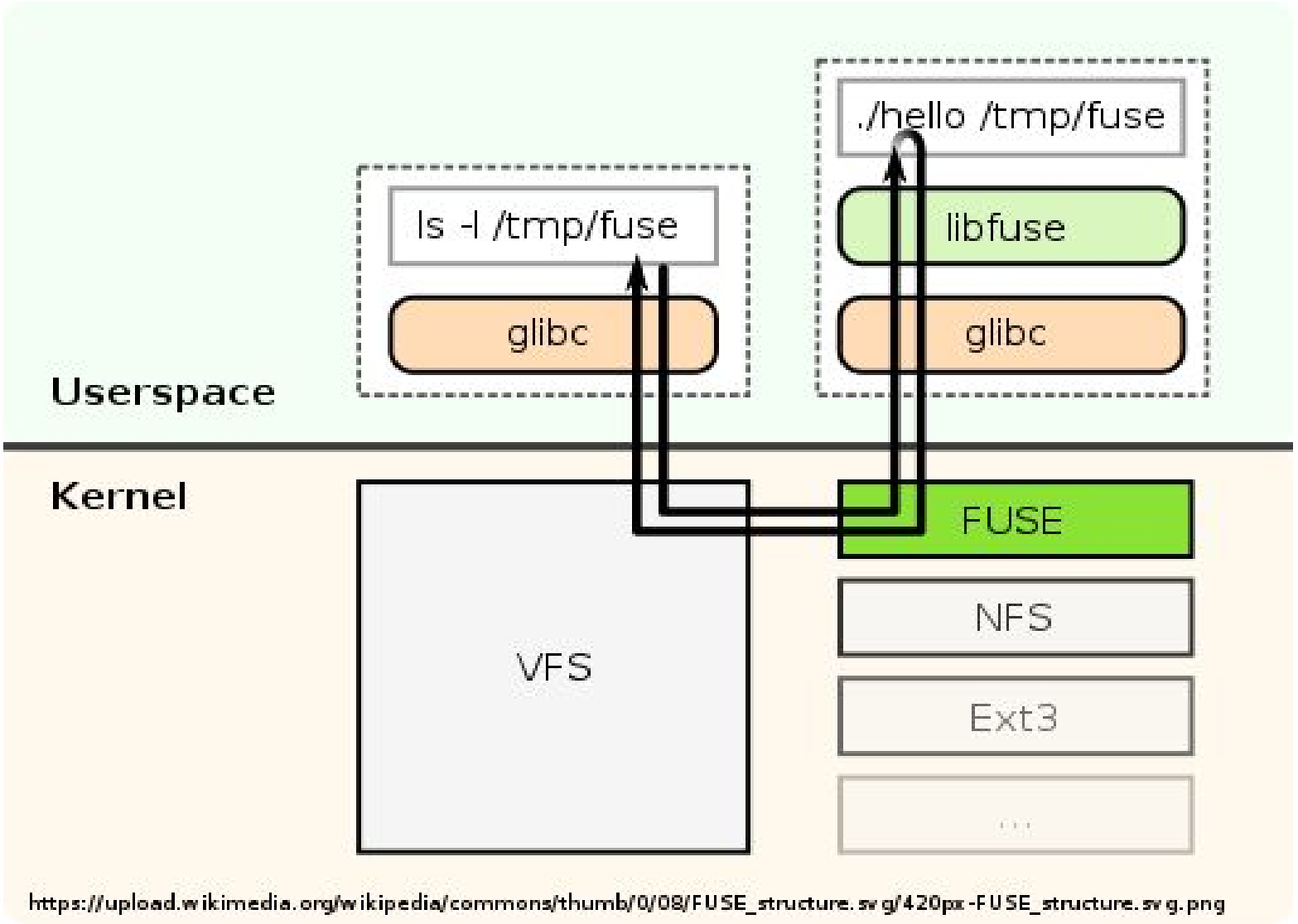
Joan and the need to scale



FUSE: What is it?

- File system in User SpacE
- Allows us to run file system “drivers” in user space.
- Acts like dispatcher between user space and kernel space. Specifically through C lib called “libfuse”, or its replacement written in Go.

FUSE: What is it?



https://upload.wikimedia.org/wikipedia/commons/thumb/0/08/FUSE_structure.svg/420px-FUSE_structure.svg.png

FUSE: How is this useful?

- FUSE is an abstraction layer between data source and “real” file-system.
- Many data sources can be interpreted as file-system. (directories, files, read, write, move, rename, …)
  - WikipediaFS
  - Cloud storage (Google drive, Dropbox, …)
  - Real file-sytems like NTFS-3G

bazil.org/fuse -- Filesystems in Go

- Pure-Go implementation of user-space server for the Linux and OS X kernel protocols.
- It is a Go replacment for C “libfuse” library

bazil.org/fuse -- Filesystems in Go

- It embraces Go fully for safety and ease of programming.
- To support file or dir. operation, simply implement it's interface for file node or open file handle object.
- Implement only what you need. (func Attr())

bazil.org/fuse -- Filesystems in Go

type Node

```
type Node interface {
    // Attr fills attr with the standard metadata for the node.
    Attr(ctx context.Context, attr *fuse.Attr) error
}
```

A Node is the interface required of a file or directory. See the documentation for type FS for general information pertaining to all methods.

A Node must be usable as a map key, that is, it cannot be a function, map or slice.

Other FUSE requests can be handled by implementing methods from the Node* interfaces, for example NodeOpener.

Methods returning Node should take care to return the same Node when the result is logically the same instance. Without this, each Node will get a new NodeID, causing spurious cache invalidations, extra lookups and aliasing anomalies. This may not matter for a simple, read-only filesystem.

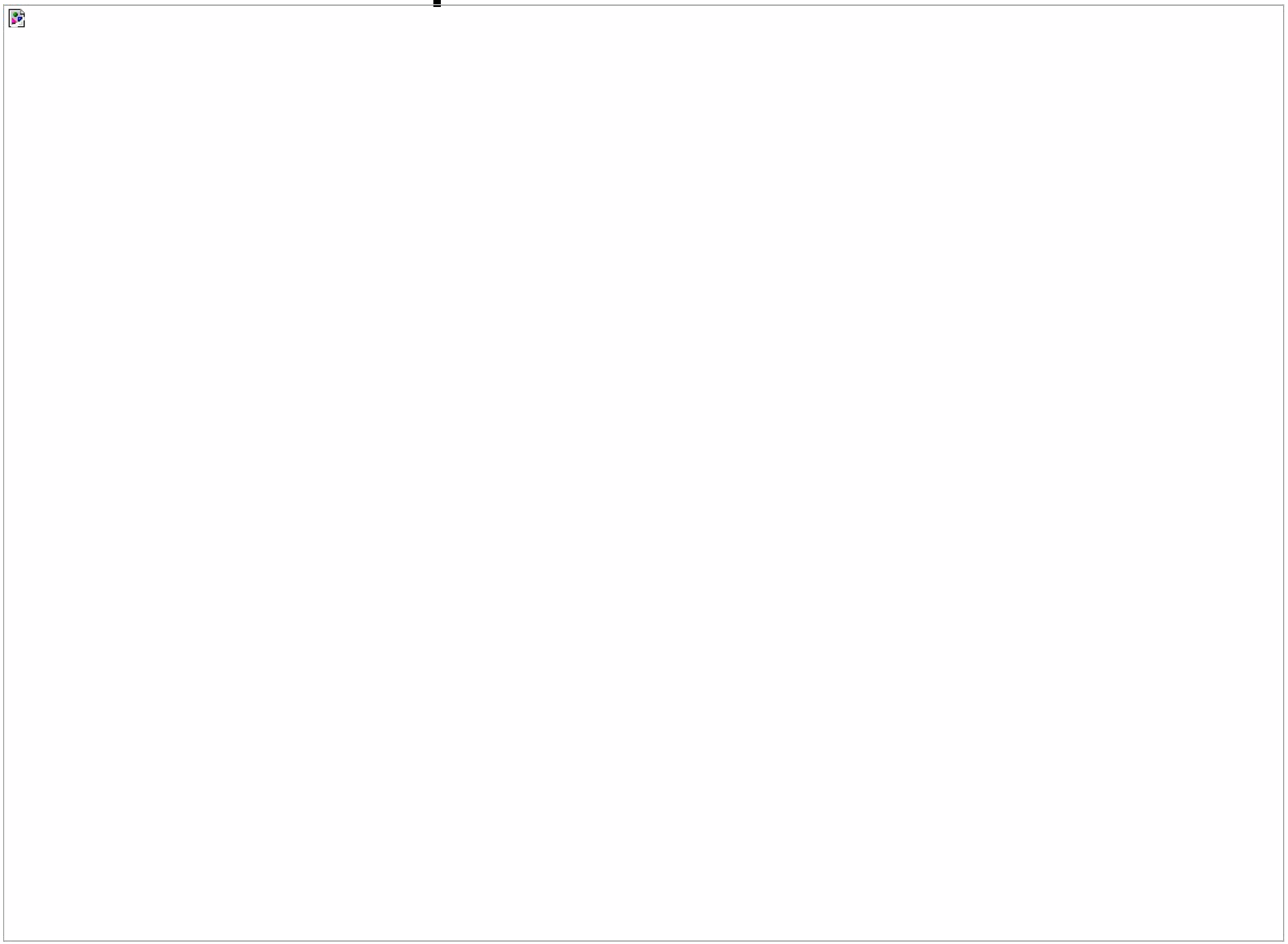
type HandleWriter

```
type HandleWriter interface {
    // Write requests to write data into the handle at the given offset.
    // Store the amount of data written in resp.Size.
    //
    // There is a writeback page cache in the kernel that normally submits
    // only page-aligned writes spanning one or more pages. However,
    // you should not rely on this. To see individual requests as
    // submitted by the file system clients, set OpenDirectIO.
    //
    // Writes that grow the file are expected to update the file size
    // (as seen through Attr). Note that file size changes are
    // communicated also through Setattr.
    Write(ctx context.Context, req *fuse.WriteRequest, resp *fuse.WriteResponse) error
}
```

Implementation

- Engine manages WebKit sessions on the server. (also does image processing, touch manip.)
- Each engine mounts cookie files to machine local file system.
- “cookiefs” is now interface between “storage service” and WebKit instances.
  - Networkmanager acts as load balancer and assignees load to the engines.

Implementation



Potential issues

- There is no cookie file locking, we rely, that NM will assign each session to one WK instance only.
- Bottlenecks : IPC, Database
- Single point of failure

Questions?