Fluent Bit

Extending Your Logging Pipeline with Go

Warren Fernandes

Pivotal

@warren_ff

Jason Keene

Pivotal

@jasonkeene

Why

Architecture

Go Interface

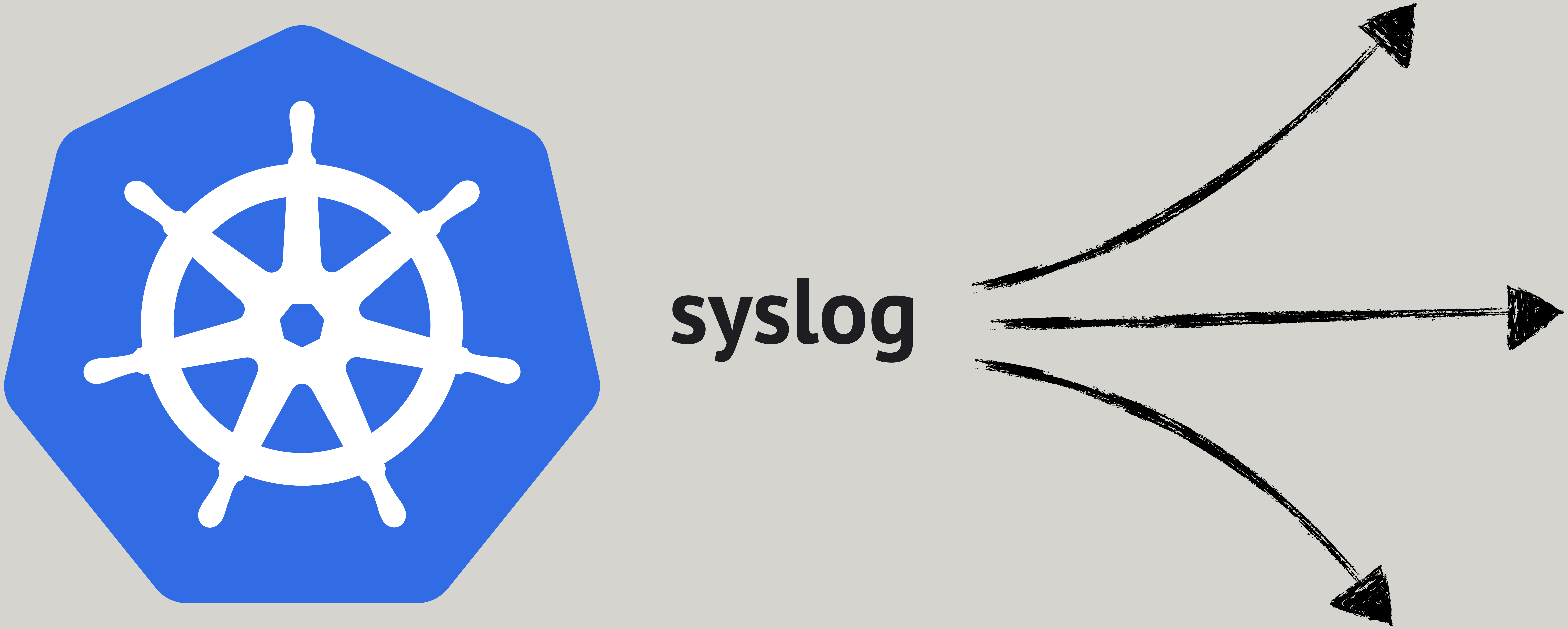
Moving Forward

Why

Architecture

Go Interface

Moving Forward



syslog

Message Reliability

was our **Primary Consideration**

Kubernetes Integration

CLOUD NATIVE
COMPUTING FOUNDATION

Go Support

RFC 5424 Support

Resource Usage

and

Performance

fluentd

fluentbit

fluentd	fluentbit
Implemented in Ruby/C	Implemented in C
Ecosystem of Plugins (900+)	Plugins are Included (54)
Extend with Ruby	Extend with C/Go/Lua
Memory Usage (40MB)	Reduced Memory Usage (500KB)
Higher CPU Usage	Lower CPU Usage
Better for Aggregation	Better as an DaemonSet
Support Forward Protocol	Support Forward Protocol

fluentd

fluentbit

	fluentd	fluentbit
Message Reliability	Buffering + Retries	Buffering + Retries
Kubernetes Support	Tail/Filter	Tail/Filter
CNCF Status	Graduated	Sub-project
Go Support	Nope	Output Plugins
RFC 5424	Not Compliant	No Output
Resource Usage	Not Bad	Great
Performance	Not Bad	Great

Why

Architecture

Go Interface

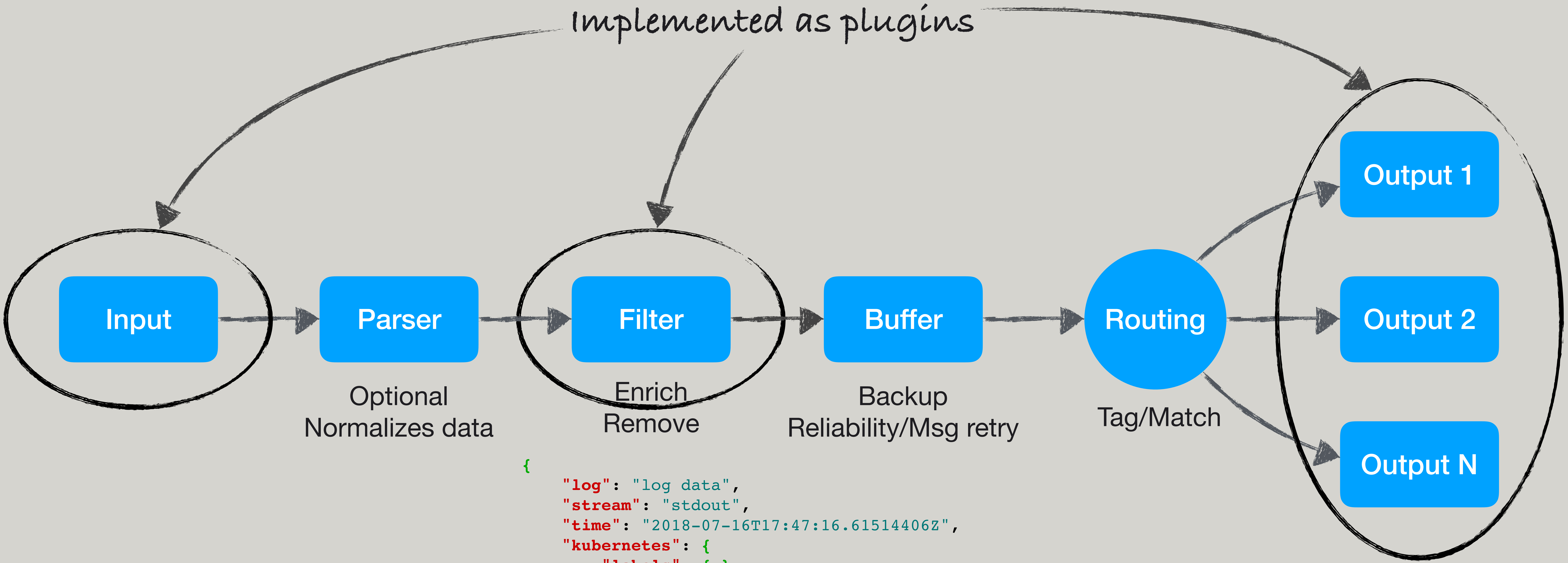
Moving Forward

Fluent Bit Deep Dive

KubeCon + CloudNativeCon, Seattle 2018

Eduardo Silva <eduardo@treasure-data.com>
@edsiper

Eduardo Silva's Deep Dive talk at KubeCon Seattle 2018



Implemented as plugins

Input

Parser

Optional
Normalizes data

Filter

Enrich
Remove

Buffer

Backup
Reliability/Msg retry

Routing

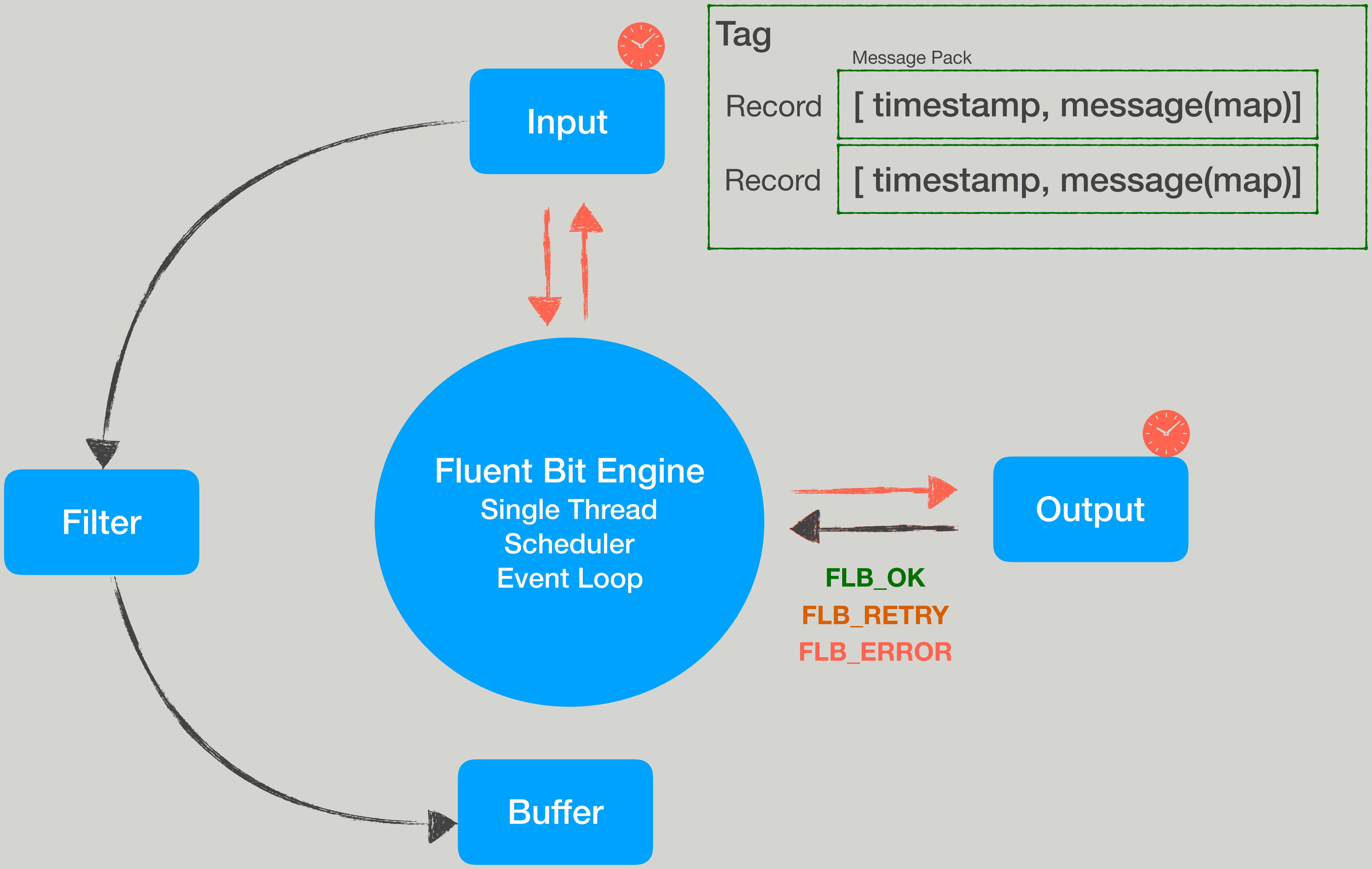
Tag/Match

Output 1

Output 2

Output N

```
{
    "log": "log data",
    "stream": "stdout",
    "time": "2018-07-16T17:47:16.61514406Z",
    "kubernetes": {
        "labels": {…},
        "annotations": {…},
        "host": "minikube",
        "container_name": "kube-addon-manager",
        "docker_id": "some-hash",
        "pod_name": "kube-addon-manager-minikube",
        "namespace_name": "kube-system",
        "pod_id": "some-hash"
    }
}
```



Input

Tag

Message Pack

Record [timestamp, message(map)]

Record [timestamp, message(map)]

Filter

Fluent Bit Engine
Single Thread
Scheduler
Event Loop

Output

FLB_OK
FLB_RETRY
FLB_ERROR

Buffer

Why

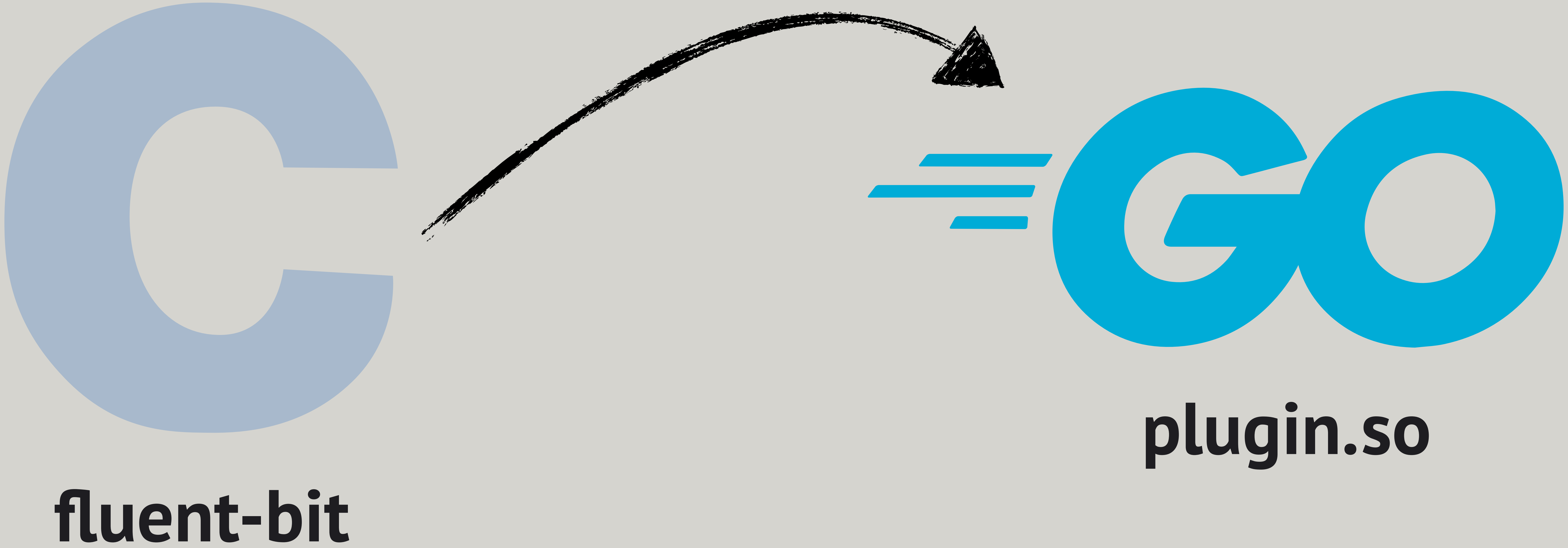
Architecture

Go Interface

Moving Forward

How does **Fluent Bit** interface with **Go?**

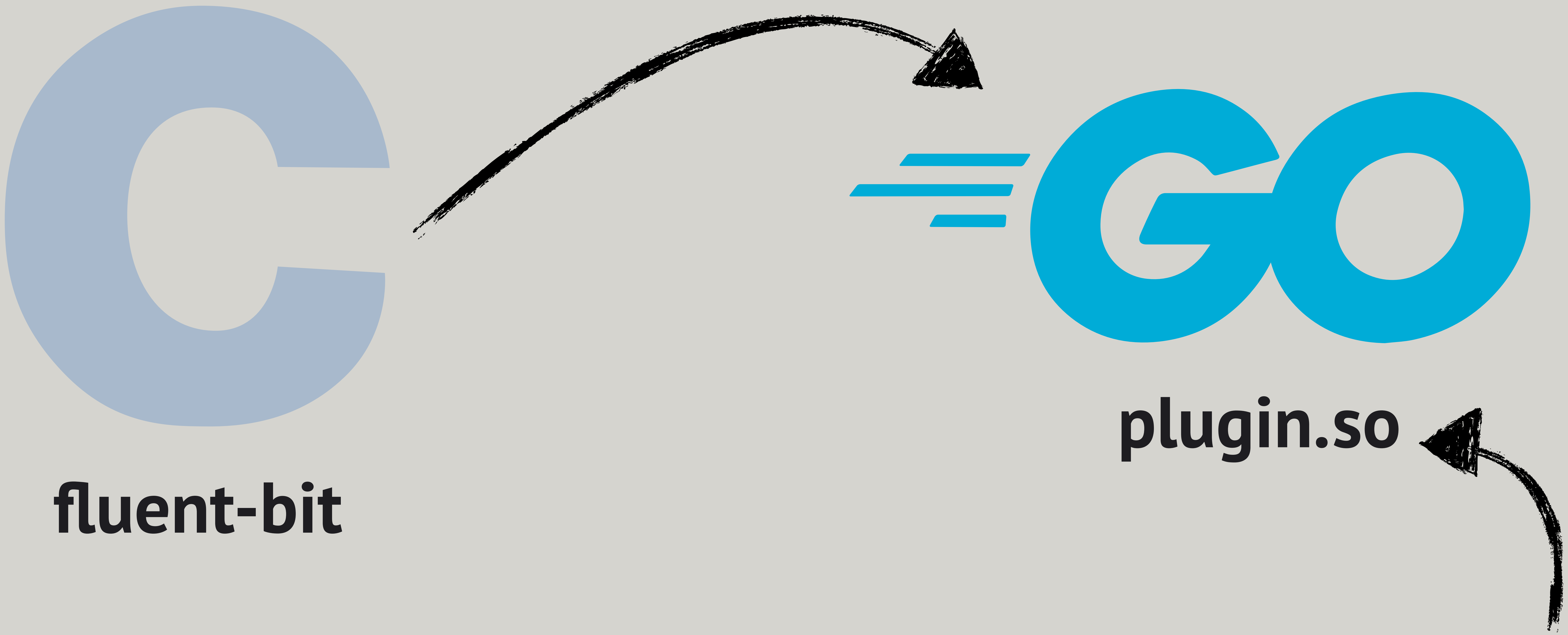
dynamic linking



fluent-bit

plugin.so

dynamic linking



fluent-bit

plugin.so

Build Modes

```
go build –buildmode ...
```

- default
- archive
- exe
- pie
- shared
- plugin
- c-archive
- c-shared

Build Modes

```
go build -buildmode ...
```

- default
- archive
- exe
- pie
- shared
- **plugin**
- c-archive
- c-shared

Build Modes

```
go build –buildmode ...
```

- default
- archive
- exe
- pie
- shared
- plugin
- **c-archive** ←
- c-shared

Build Modes

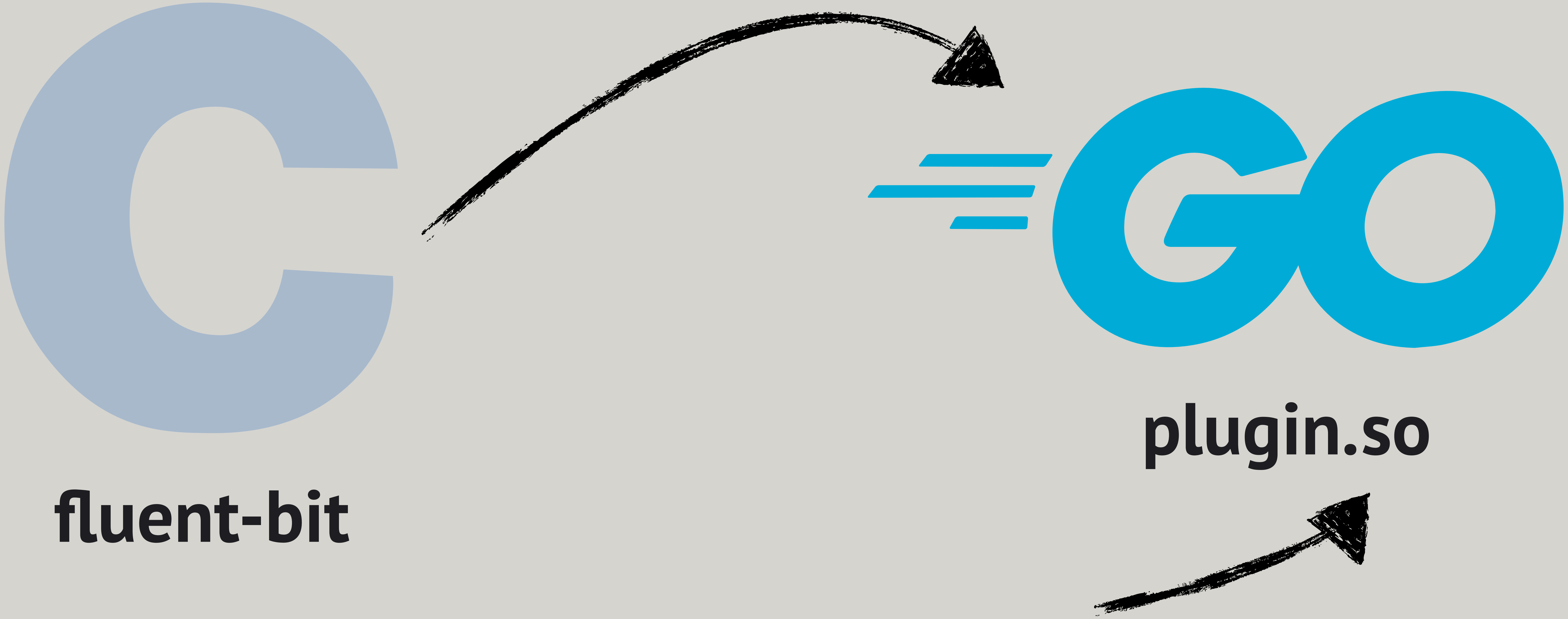
```
go build —buildmode ...
```

- default
- archive
- exe
- pie
- shared
- plugin
- c-archive
- **c-shared**

```go
package main

import "C"

//export MyAwesomeFunction
func MyAwesomeFunction() {
  println("You are awesome!")
}

func main() {}
```

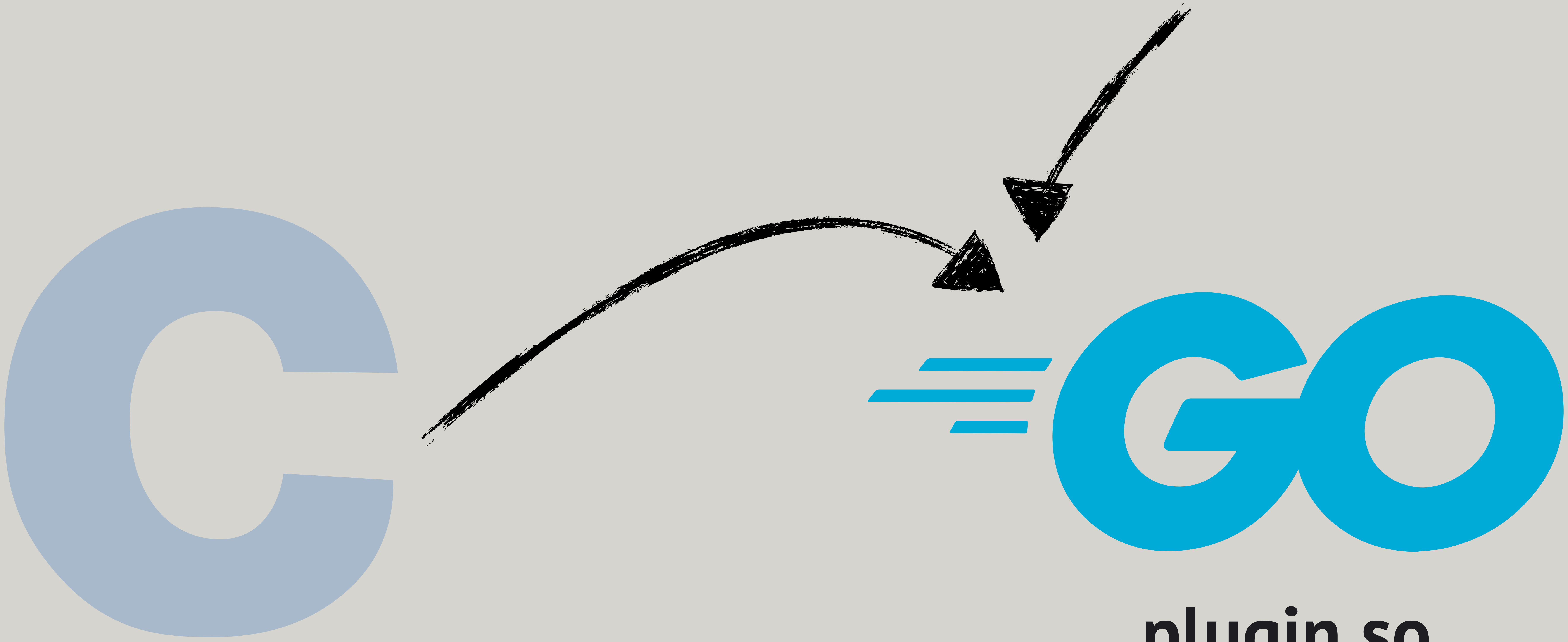
```
$ go build -o plugin.so -buildmode c-shared
```

```
$ readelf --dyn-syms plugin.so | grep MyAwesomeFunction
    42: 0000000000095470    51 FUNC    GLOBAL DEFAULT   12 MyAwesomeFunction
```



fluent-bit

plugin.so



fluent-bit

plugin.so

C Types

void *	unsafe.Pointer
char *	*C.char
int	C.int
unsigned long long	C.ulonglong
struct foo	C.struct_foo
union foo	C.union_foo
enum foo	C.enum_foo
__int128_t	[16]byte

Go Types

unsafe.Pointer	void *
string	GoString
[]byte	GoSlice
int	GoInt
uint64	GoUint64
complex128	GoComplex128

What about functions that return

Multiple Values?

```go
//export MultipleReturns
func MultipleReturns() (int, *int, string, []byte) {
  return 0, nil, "", nil
}
```

```c
/* Return type for MultipleReturns */
struct MultipleReturns_return {
  GoInt r0;
  GoInt* r1;
  GoString r2;
  GoSlice r3;
};
```

How do you write a **Fluent Bit Go** plugin?

```go
//export FLBPluginRegister
func FLBPluginRegister(def unsafe.Pointer) int {
  // Gets called only once when the .so is loaded.
}

//export FLBPluginInit
func FLBPluginInit(plugin unsafe.Pointer) int {
  // Gets called once for each instance you have configured.
}

//export FLBPluginFlushCtx
func FLBPluginFlushCtx(ctx, data unsafe.Pointer, length C.int, tag *C.char) int {
  // Gets called once for each message to be written to an instance.
}

//export FLBPluginExit
func FLBPluginExit() int {
  // Gets called on teardown.
}
```

```go
//export FLBPluginRegister
func FLBPluginRegister(def unsafe.Pointer) int {
 // Gets called only once when the .so is loaded.

 return output.FLBPluginRegister(
     def, "multiinstance", "Testing multiple instances.")
}
```

```go
//export FLBPluginInit
func FLBPluginInit(plugin unsafe.Pointer) int {
  // Gets called once for each instance you have configured.

  // Read configuration values.
  hostname := output.FLBPluginConfigKey(plugin, “hostname")

  // Set the context to point to any Go variable.
  // This is used to know what instance to flush messages for.
  output.FLBPluginSetContext(plugin, unsafe.Pointer(&hostname))

  // Return FLB_OK or FLB_ERROR.
  return output.FLB_OK
}
```

```go
//export FLBPluginFlushCtx
func FLBPluginFlushCtx(ctx, data unsafe.Pointer, length C.int, tag *C.char) int {
  // Gets called once for each message to be written to an instance.

  // Cast context back into the original type.
  hostname := *(*string)(ctx)

  dec := output.NewDecoder(data, int(length))
  for {
    ret, _, record := output.GetRecord(dec)
    if ret != 0 {
      break
    }
    log.Printf(“Flushing to hostname: %s, data: %v”, hostname, record)
    // ...
  }

  // Return FLB_OK or FLB_ERROR.
  return output.FLB_OK
}
```

```
//export FLBPluginExit
func FLBPluginExit() int {
  // Gets called on teardown.

  // Return FLB_OK or FLB_ERROR.
  return output.FLB_OK
}
```

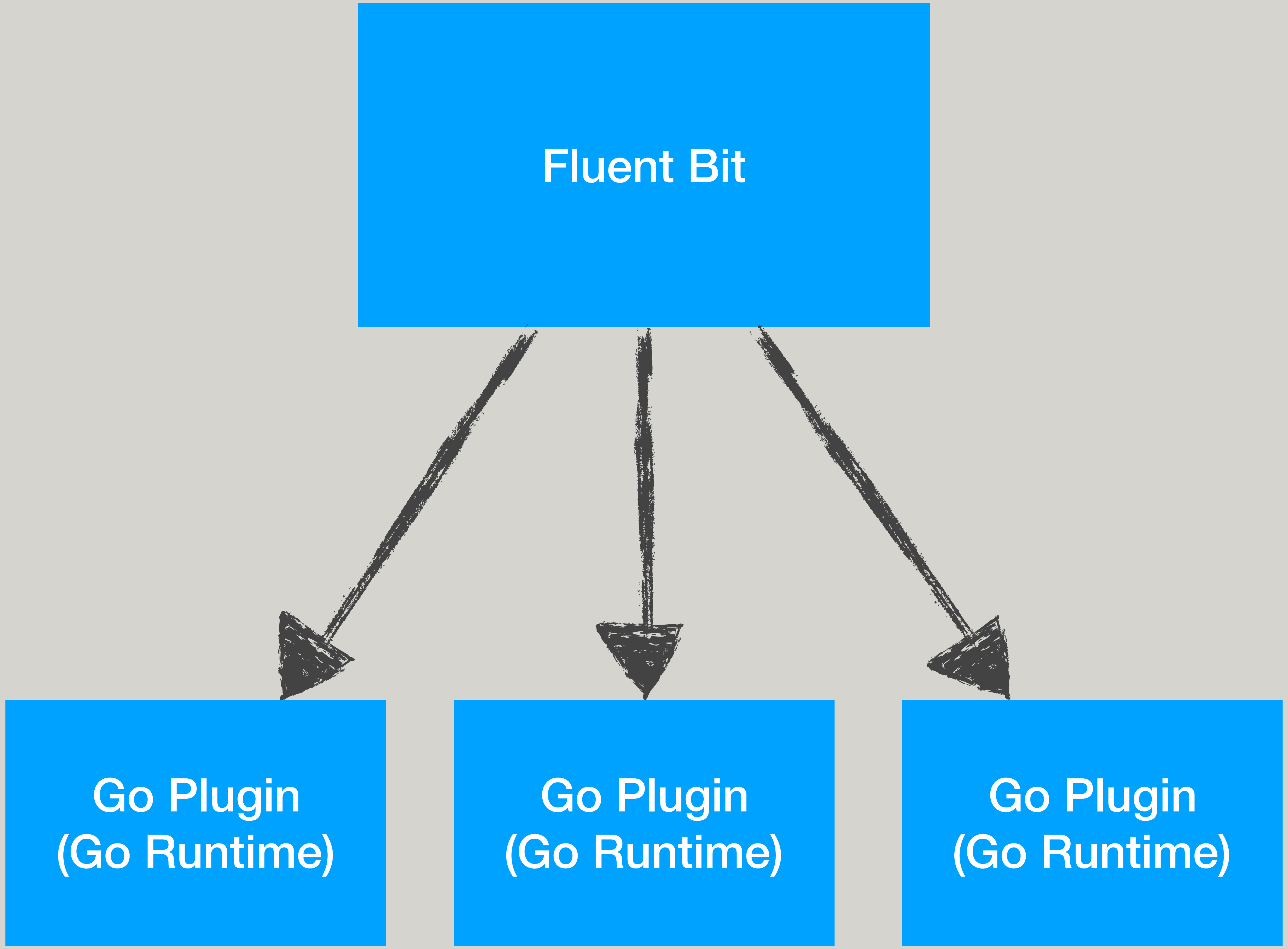
Why

Architecture

Go Interface

Moving Forward

Better Support for Multiple Go Plugins

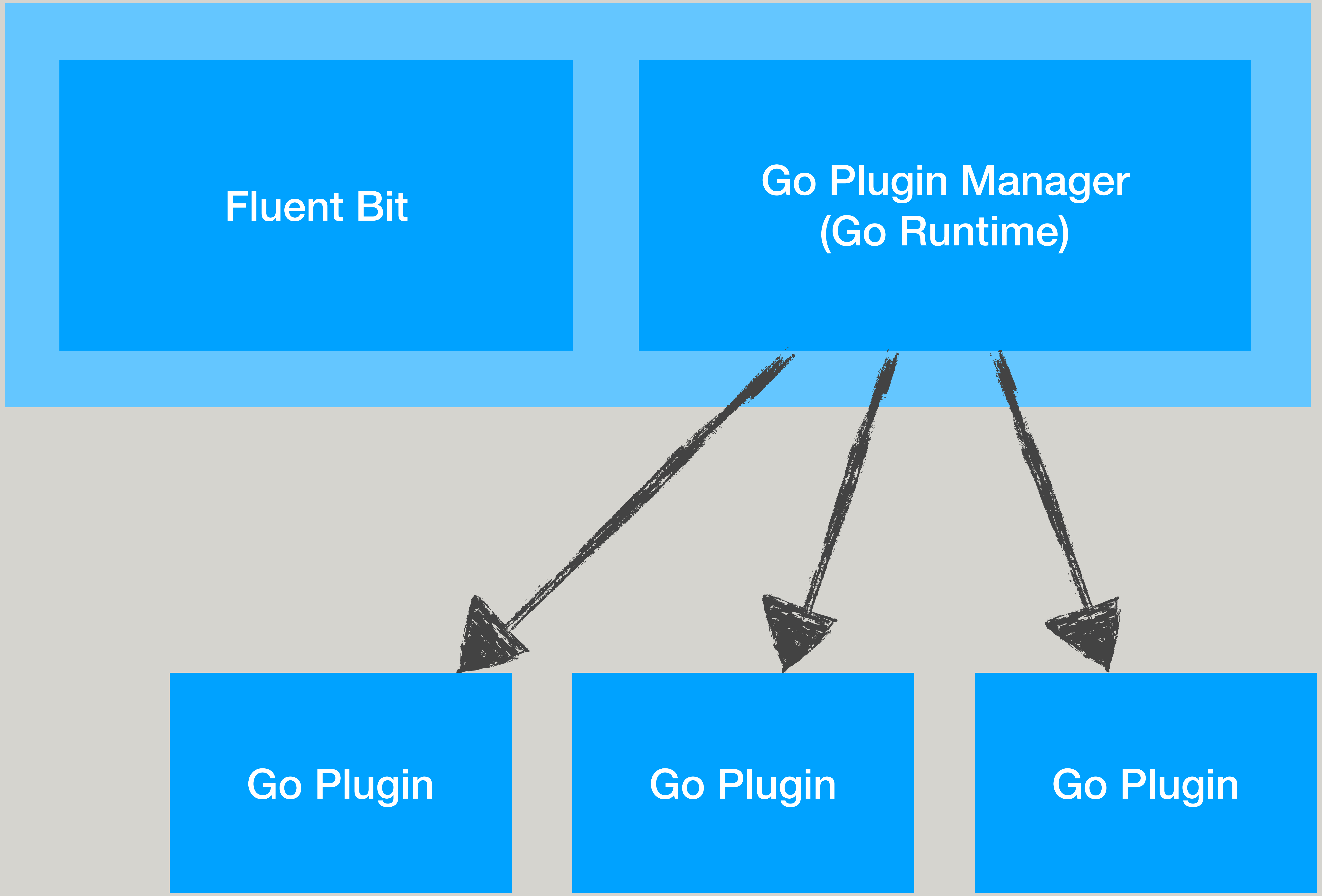


-buildmode c-shared

Build Modes

```
go build –buildmode ...
```

- default
- archive
- exe
- pie
- shared
- **plugin** ←
- **c-archive** ←
- c-shared



Fluent Bit

Go Plugin Manager
(Go Runtime)

Go Plugin

Go Plugin

Go Plugin

-buildmode c-archive

-buildmode plugin

Establish a Versioned ABI

Input and Filter plugins in Go

Expose logging and metrics for Go plugins

Thanks!

Questions?