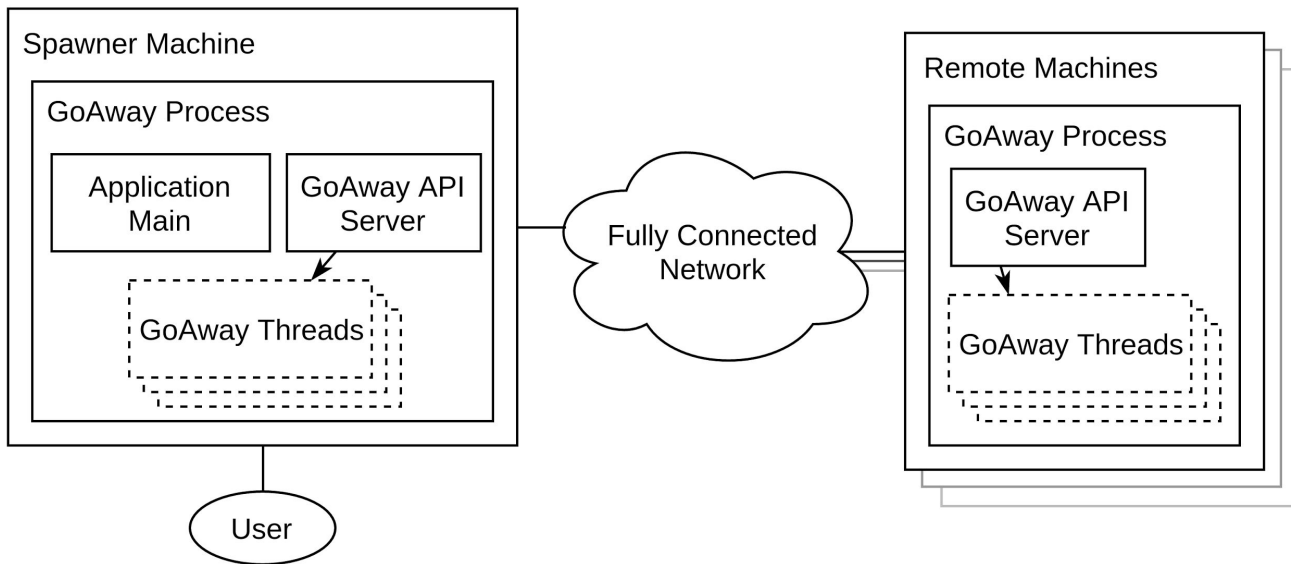


GoAway

Jessica Kenney, Andres Perez, Miles Steele

System

- User runs an application on a spawner machine.
- GoAway threads are sent to remote machines in the cluster.

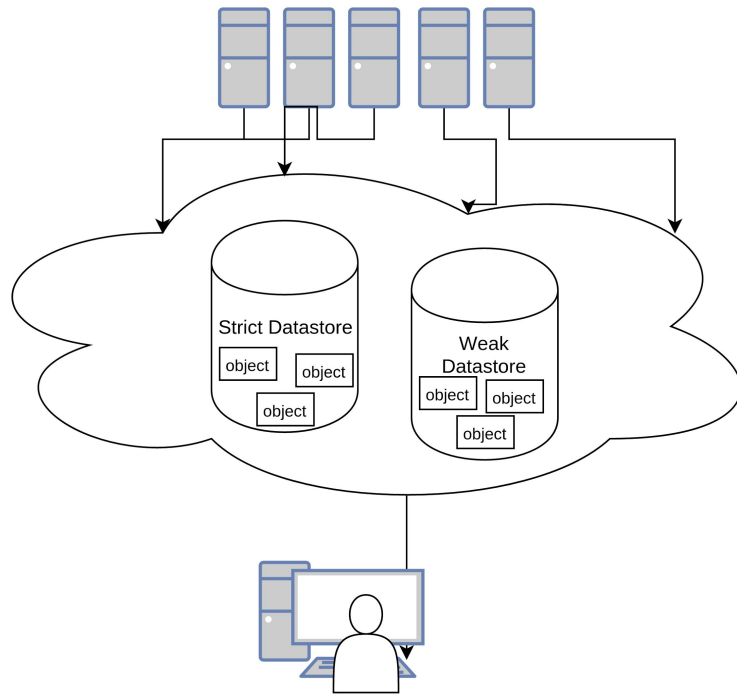


Data Stores

DataStores hold shared memory and provide a consistency model.

GoAway provides four:

- Strict Centralized
- Linearizable Fast Read
- Weakly Consistent
- Release Consistency

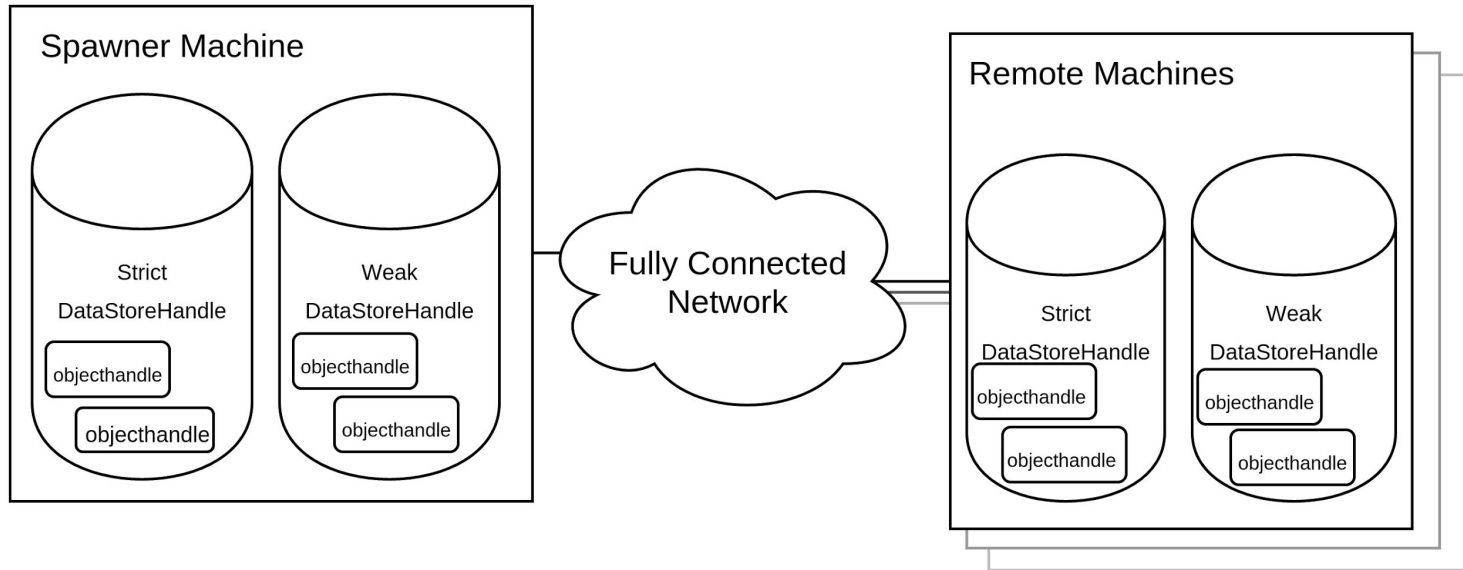


Usage

```
import goaway
# Initialize a shared lock.
lock = goaway.Lock("lock")
# Initialize a shared object in centralized linearizable datastore.
s = goaway.StrictCentralized("s")

def increase(value):
    with lock:
        s.counter += value

if __name__ == "__main__":
    # Initialize GoAway and start the remote servers.
    goaway.init(config_file_path)
    for i in range(n):
        # Run the function on a remote machine.
        goaway.goaway(increase, 1)
```

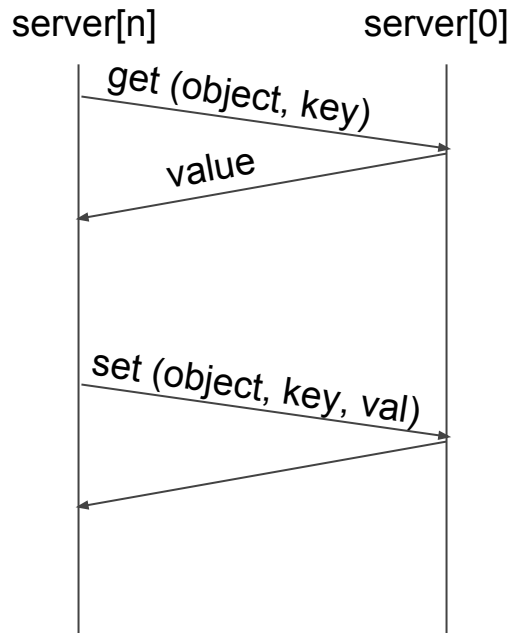


Strict Centralized DataStore

All of the data lives on server[0]

```
get(object, key):  
    return rpc(server[0], "get", object, key)
```

```
set(object, key, value):  
    rpc(server[0], "set", object, key, value)
```



Linearizable Fast-Read DataStore

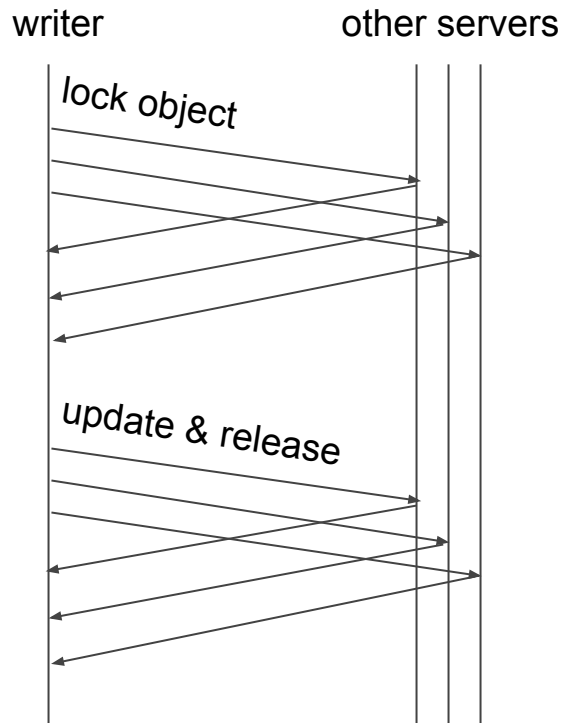
Optimized for fast reads at the expense of slow writes.
Guarantees linearizability.

Get

- Local read (fast)
- Reads occur even if the object is write-locked

Set

- 2-phase locking commit (slow)
 - Lock object on all servers in order
 - Update and unlock object on all servers

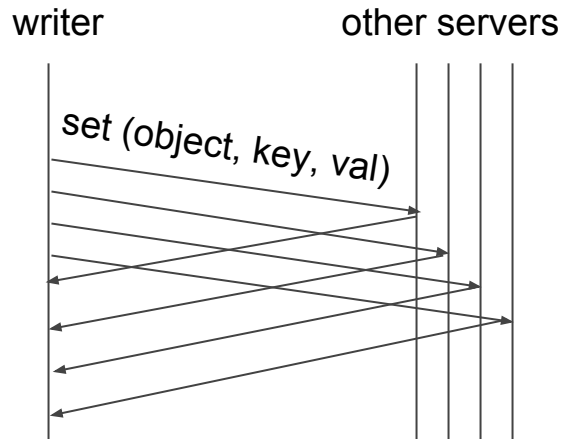


Weakly-consistent DataStore

```
get(object, key):  
    return self.data[object][key]
```

```
set(object, key, value):  
    for server in servers:  
        spawn thread(rpc(server, "set", object, key, value))
```

```
sync(object):  
    while outgoing RPCs for object:  
        # wait  
    return
```



P0: W(x, 1)		S	
P1:	W(x, 2)	S	
P2:			R(x, 1)
P3:			R(x, 2)
P4:	_____ R(x, 0)		

Demo (Hash Puzzle)

- Program that iterates through strings to find a hash with leading 0's.
- On 7 machines.

```
# call run_chunk with the arguments a, b on seperate machines.  
for a, b in bounds:  
    goaway.goaway(run_chunk, seed, a, b)
```


GoAway

github.com/anpere/goaway