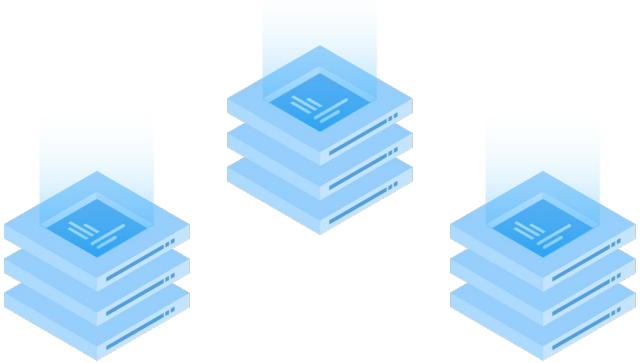
How We Run TiDB In Browser

Presented by Zhou Shuai

PingCAP TiDB



PingCAP

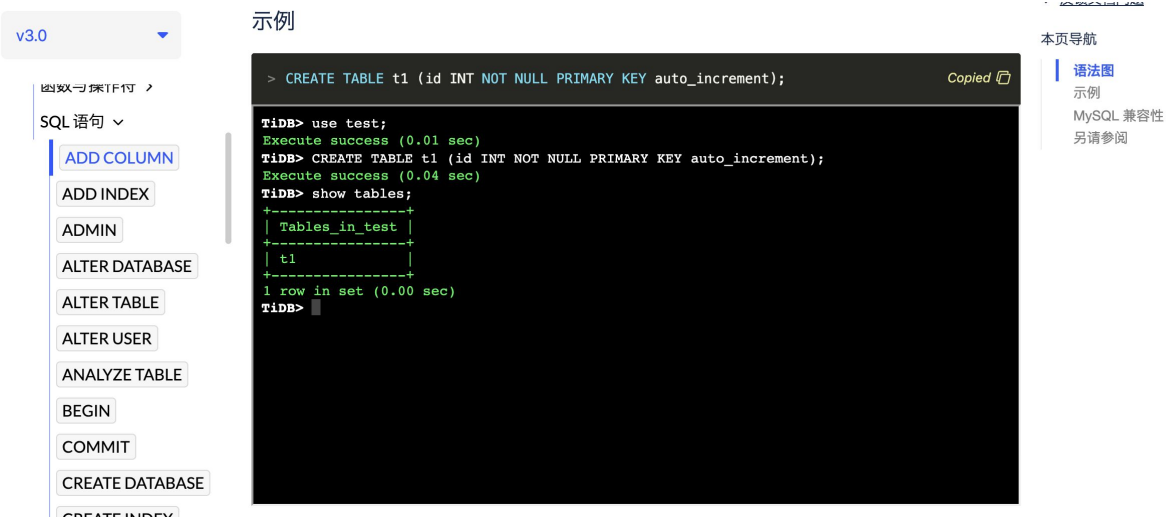
Part I - The purpose

Why

- What can webassembly do ?
  - Portable: it’s easy to port webassembly application
- Did TiDB(databases) run in browser matters ?
  - A first Golang DB in webassembly
  - All users can try TiDB very easy
  - Can store sql data for any other products in browser
- bad cases
  - http://play.etcd.io/install (so ugly)

All users can try TiDB very easy

- make it easy for user to study SQL online
- make it easy for user to try new features of TiDB



PingCAP

PingCAP

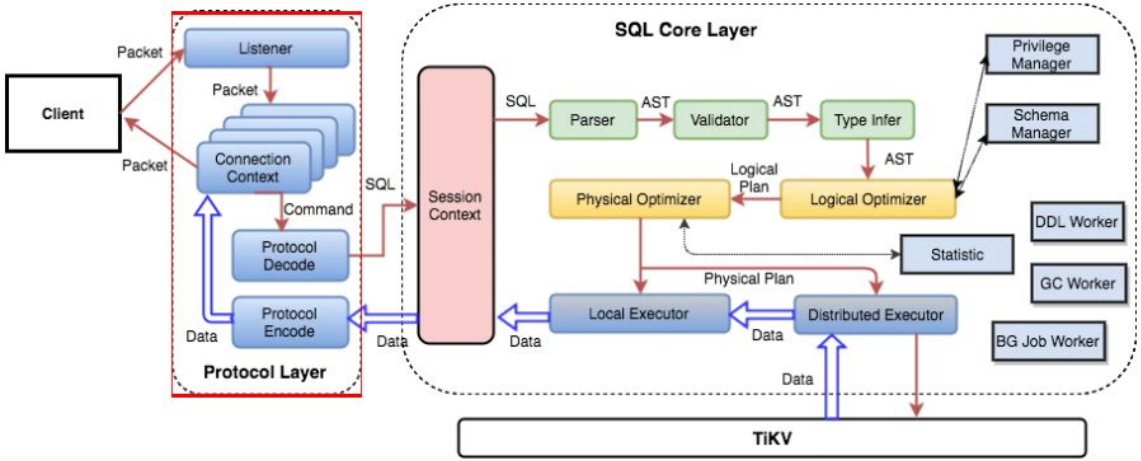
Part II - The implement

PingCAP

II-I - TiDB WIth WebAssembly

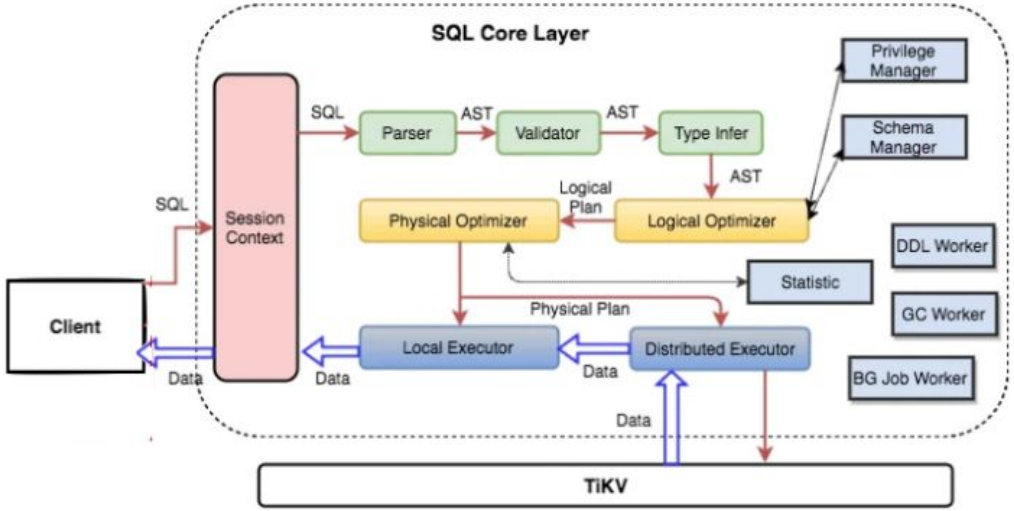
Security policy of browser

- Problem
  - Browser is a sandbox, port listening and file operations are not permitted
  - TiDB needs to listen on a port to provide service



Security policy of browser

- Solution
  - Remove protocol layer, integrated a SQL terminal with TiDB and send SQL command to TiDB session directly



Build Constraints In Golang

A built constraint (aka. build tag) lists the conditions under which a file should be included in the package.

- use comment
  - `// +build linux,386 darwin,!cgo`

    `(linux AND 386) OR (darwin AND (NOT cgo))`
  - `// +build linux darwin`

    `// +build 386`

    `(linux OR darwin) AND 386`
- with file name
  - `*_GOOS`
  - `*_GOARCH`
  - `*_GOOS_GOARCH`
    - `example: source_windows_amd64.go`

PingCAP

Code Compatibility Issues

3rd-party lib has platform-specific code but don't support Wasm platform

Example:

```
# github.com/pingcap/goleveldb/leveldb/storage
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:81:16: undefined: newFileLock
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:166:3: undefined: rename
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:252:11: undefined: rename
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:257:11: undefined: syncDir
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:354:14: undefined: rename
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:483:9: undefined: rename
../vendor/github.com/pingcap/goleveldb/leveldb/storage/file_storage.go:519:13: undefined: syncDir
```

the file_storage_xxx.go is platform-specific but it doesn't has a Wasm version:

```
-rw-r--r--  1 joshua  wheel  12991 10 27 08:50 file_storage.go
-rw-r--r--  1 joshua  wheel    631 10 27 08:50 file_storage_nacl.go
-rw-r--r--  1 joshua  wheel   1161 10 27 08:50 file_storage_plan9.go
-rw-r--r--  1 joshua  wheel   1442 10 27 08:50 file_storage_solaris.go
-rw-r--r--  1 joshua  wheel   3855 10 27 08:50 file_storage_test.go
-rw-r--r--  1 joshua  wheel   1605 10 27 08:50 file_storage_unix.go
-rw-r--r--  1 joshua  wheel   1824 10 27 08:50 file_storage_windows.go
-rw-r--r--  1 joshua  wheel   3867 10 27 09:51 mem_storage.go
-rw-r--r--  1 joshua  wheel   1522 10 27 08:50 mem_storage_test.go
-rw-r--r--  1 joshua  wheel   4682 10 27 08:50 storage.go
```

Solution: replace incompatible packages with modified ones in go.mod

Golang WebAssembly Issue

- Some os functions are missing in golang's wasm_exec.js (master fix this)

```
panic: syscall/js: Value.Call: property stat is not a function, got undefined

goroutine 1 [running]:
syscall/js.Value.Call(0x7ff8000000000012, 0x46d67f, 0x4, 0xc00b5e0, 0x2, 0x2, 0x1)
    /Users/joshua/.gvm/gos/go1.12.5/src/syscall/js/js.go:299 +0x87
syscall.fsCall(0x46d67f, 0x4, 0xc0c39f8, 0x1, 0x1, 0x10b10004, 0xa0, 0x358ea0)
    /Users/joshua/.gvm/gos/go1.12.5/src/syscall/fs_js.go:477 +0xc
syscall.Stat(0x46d933, 0x5, 0xc1c0218, 0x25e8248, 0x18d60004)
    /Users/joshua/.gvm/gos/go1.12.5/src/syscall/fs_js.go:185 +0x5
os.statNolog(0x46d933, 0x5, 0x1a, 0x17870006, 0x6eec80, 0xc00b5a0)
    /Users/joshua/.gvm/gos/go1.12.5/src/os/stat_unix.go:31 +0x3
os.Stat(0x46d933, 0x5, 0x0, 0x555500a5, 0x6eec80, 0xc00b5a0)
    /Users/joshua/.gvm/gos/go1.12.5/src/os/stat.go:13 +0x3
github.com/prometheus/procfs.NewFS(0x46d933, 0x5, 0x1, 0x0, 0x0, 0x0)
    /Users/joshua/.gvm/pkgsets/go1.12.5/global/pkg/mod/github.com/prometheus/procfs@v0.0.0-20181005140218-185b4288413d/fs.go:35 +0x2
github.com/prometheus/procfs.NewStat(0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, ...)
    /Users/joshua/.gvm/pkgsets/go1.12.5/global/pkg/mod/github.com/prometheus/procfs@v0.0.0-20181005140218-185b4288413d/stat.go:83 +0x2
github.com/prometheus/client_golang/prometheus.NewProcessCollector(0x0, 0x0, 0x0, 0xc039a00, 0x1, 0x1)
    /Users/joshua/.gvm/pkgsets/go1.12.5/global/pkg/mod/github.com/prometheus/client_golang@v0.9.0/prometheus/process_collector.go:129 +0x18
github.com/prometheus/client_golang/prometheus.init.0()
    /Users/joshua/.gvm/pkgsets/go1.12.5/global/pkg/mod/github.com/prometheus/client_golang@v0.9.0/prometheus/registry.go:56 +0x2
exit code: 2
```

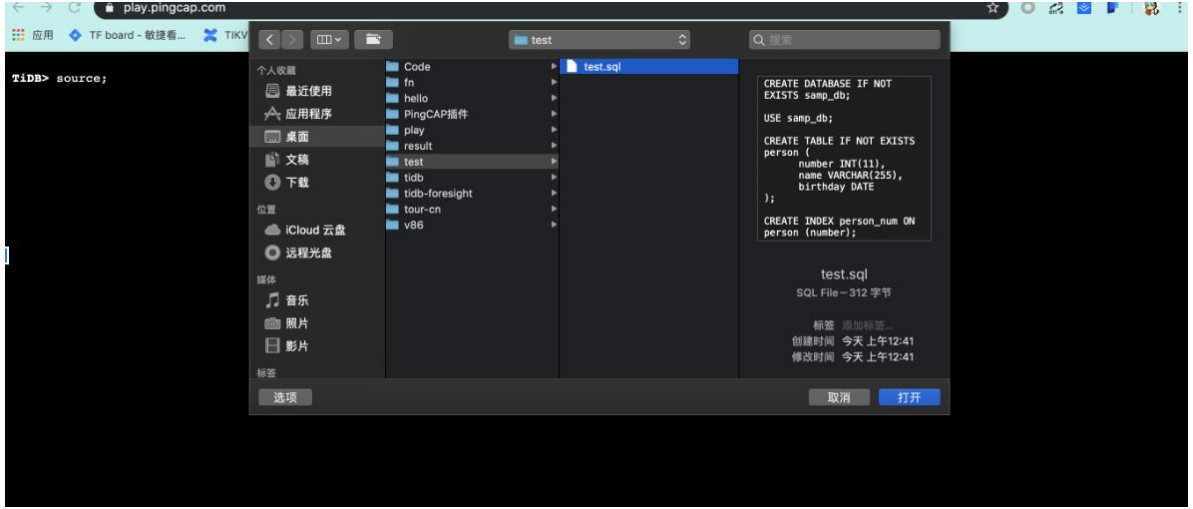
- Solution: mock these operations before run wasm

```
fs.stat = unimplemented1;
fs.lstat = unimplemented1;
fs.unlink = unimplemented1;
fs.rmdir = unimplemented1;
fs.mkdir = unimplemented2;
go.run(result.instance);
```

PingCAP

File System Access

- Some SQL statement need read file on client side
  - load stats
  - load data
  - run script file
- Implement a file uploader



PingCAP

II - II - TiDB With WASI

WASI

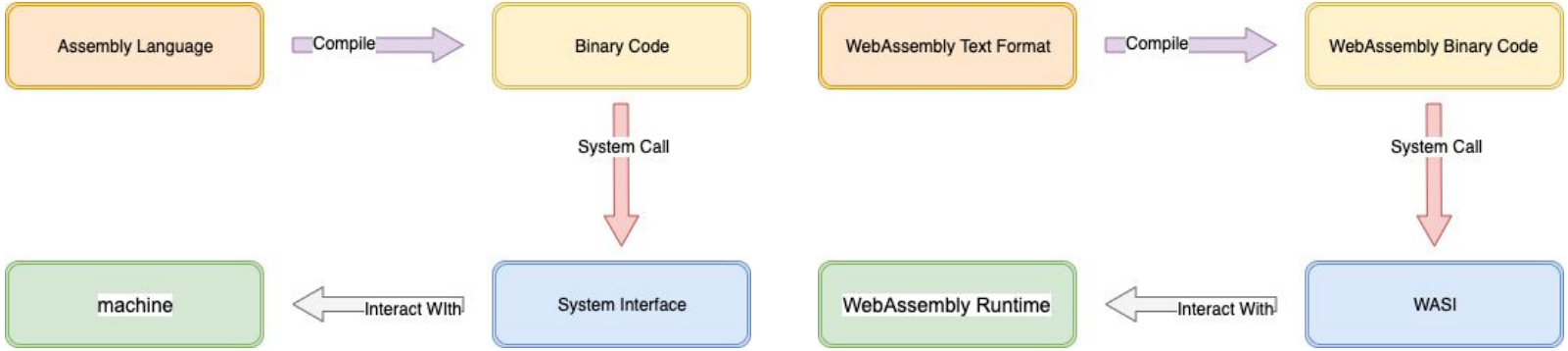
WASI is a modular system interface for WebAssembly.

Why

- Developers are starting to push WebAssembly beyond the browser
- Code outside of a browser needs a way to talk to the system—a system interface

What

- WebAssembly is an assembly language for a conceptual machine
- So it needs a system interface for a conceptual operating system
- This is what WASI is—a system interface for the WebAssembly platform.



PingCAP

WASI Hello World

```
(module
    ;; type iov struct { iov_base, iov_len int32 }
    ;; func fd_write(id *iov, iovs_len int32, nwritten *int32) (written int32)
    (import "wasi_unstable" "fd_write" (func $fd_write (param i32 i32 i32 i32) (result i32)))

    (memory 1)(export "memory" (memory 0))

    ;; The first 8 bytes are reserved for the iov array, starting with address 8
    (data (i32.const 8) "hello world\n")

    ;; _start is similar to main function, will be executed automatically
    (func $main (export "_start")
        (i32.store (i32.const 0) (i32.const 8))  ;; iov.iov_base - The string address is 8
        (i32.store (i32.const 4) (i32.const 12)) ;; iov.iov_len  - String length

        (call $fd_write
            (i32.const 1)  ;; 1 is stdout
            (i32.const 0)  ;; *iovs - The first 8 bytes are reserved for the iov array
            (i32.const 1)  ;; len(iovs) - Only 1 string
            (i32.const 20) ;; nwritten - Pointer, inside is the length of the data to be written
        )
        drop ;; Ignore return value
    )
)
```

PingCAP

wasmer.io

- wasmer: a command line tool to run .wasm file out of browser

```
JoshuadeMacBook-Pro:aabbcc joshua$ wat2wasm hello.wat -o hello.wasm
JoshuadeMacBook-Pro:aabbcc joshua$ wasmer run hello.wasm
hello world
```

- wapm: WebAssebmly package manager, like yum on CentOS, or Cargo for rust.
- webassembly.sh: a web shell to run wasm/wasi in browser

Benefit

- Integration into the WASM ecosystem
- Compile once, run anywhere
- Package management

```
JoshuadeMacBook-Pro:wasm-dist joshua$ wapm install lucklove/tidb
[INFO] Installing lucklove/tidb@0.1.6
Package installed successfully to wapm_packages!
JoshuadeMacBook-Pro:wasm-dist joshua$ wapm run tidb
TiDB [pingcap]> show databases;
+--------------------+
| Database           |
+--------------------+
| INFORMATION_SCHEMA |
| PERFORMANCE_SCHEMA |
| mysql              |
| test               |
+--------------------+
4 row in set (0.02 sec)
```

Benefit

- Another way to run TiDB in browser

webassembly.sh

应用 | TF board - 敏捷看... | TIKV-Efficiency-T... | case study - 性能... | Netflix/flamescop... | Brendan Gregg's...

```
WebAssembly.sh v0.0.2
Powered by Wasmer-JS
    @wasmer/wasm-terminal v0.5.1

QUICK START:
    • Try a command: `cowsay hello`.
    • Run a JS Engine in Wasm: `quickjs`.
    • Manage Wasm modules: `wapm`.

ADDITIONAL COMMANDS:
    All WASI packages published on WAPM.io are automatically
    available in this shell: https://wapm.io/interface/wasi

MORE INFO:
    • Usage information: `help`
    • About the project: `about`

$ tidb
TiDB [pingcap]> show databases;
+--------------------+
| Database           |
+--------------------+
| INFORMATION_SCHEMA |
| PERFORMANCE_SCHEMA |
| mysql              |
| test               |
+--------------------+
4 row in set (0.01 sec)
TiDB [pingcap]>
```

PingCAP

Challenge: WASI target is not support in Golang

- Official golang doesn’t support WASI target yet
- There is an unofficial fork implements it, however, it produce wrong binary on TiDB

```
JoshuadeMacBook-Pro:wasi joshua$ wasmer-js run main.wasm --backend singlepass
LinkError: WebAssembly.instantiate(): data segment is out of bounds
```

Solution

- Cherry pick all commits related to Wasm from official repo

Changes on Golang

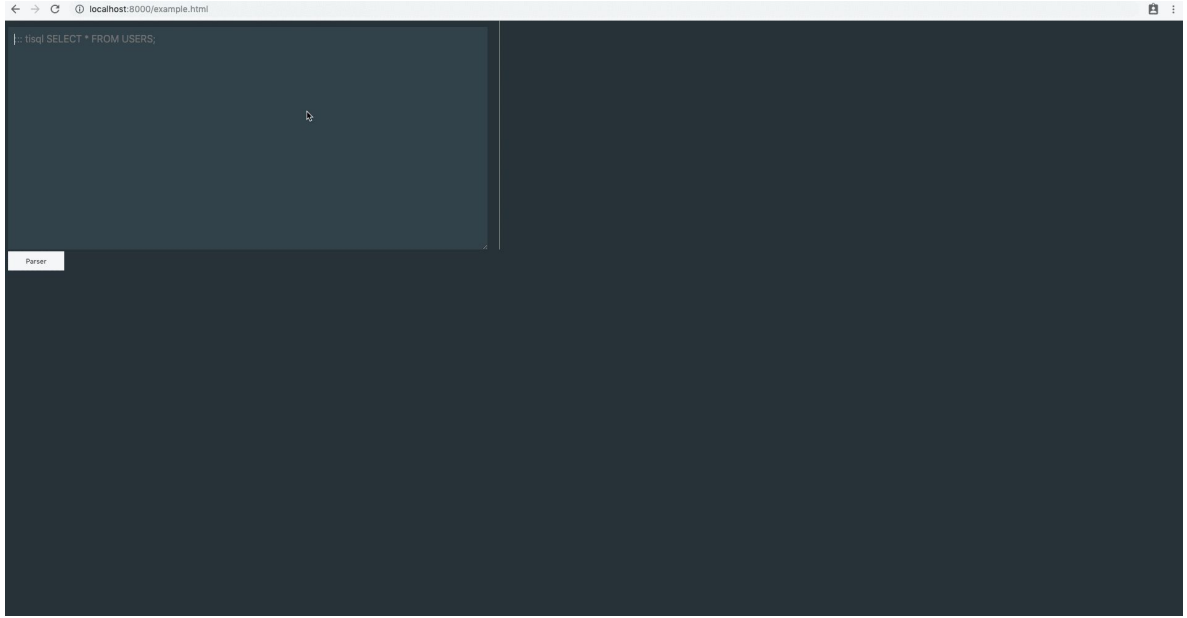
- Add wasi/wasm pair so that user can compile with GOOS=wasi GOARCH=wasm go build ...
- Implement WASI:
  - src/crypto/rand/rand_wasi.go
  - src/crypto/rand/root_wasi.go
  - src/internal/poll/fd_wasi.go
  - src/internal/syscall/unix/nonblocking_wasi.go
  - src/os/dir_wasi.go
  - src/os/exec/lp_wasi.go
  - src/os/stat_wasi.go
  - src/os/sys_wasi.go
  - src/runtime/lock_wasi.go
  - src/runtime/mem_wasi.go
  - src/runtime/os_wasi.go
  - src/syscall/net_wasi.go
  - src/syscall/syscall_wasi.go
  - src/syscall/tables_wasi.go
  - src/time/zoneinfo_wasi.go
  - src/runtime/rt0_wasi_wasm.s

PingCAP

Part III - The result

Products on tidb.wasm

- tidb playground: https://play.pingcap.com/
- tidb tour: https://tour.pingcap.com/
- tidb in webassembly shell: https://webassembly.sh/?run-command=tidb
- tidb-wasm-markdown (by community): https://github.com/imiskolee/tidb-wasm-markdown



PingCAP

PingCAP

Part IV - The Future

TodoList

- Persist data in the browser (implement storage backend for browser)

```go
// Storage defines the interface for storage.
// Isolation should be at least SI(SNAPSHOT ISOLATION)
type Storage interface {
    // Begin transaction
    Begin() (Transaction, error)
    // BeginWithStartTS begins transaction with startTS.
    BeginWithStartTS(startTS uint64) (Transaction, error)
    // GetSnapshot gets a snapshot that is able to read any data which data is <= ver.
    // if ver is MaxVersion or > current max committed version, we will use current version for this snapshot.
    GetSnapshot(ver Version) (Snapshot, error)
    // GetClient gets a client instance.
    GetClient() Client
    // Close store
    Close() error
    // UUID return a unique ID which represents a Storage.
    UUID() string
    // CurrentVersion returns current max committed version.
    CurrentVersion() (Version, error)
    // GetOracle gets a timestamp oracle client.
    GetOracle() oracle.Oracle
    // SupportDeleteRange gets the storage support delete range or not.
    SupportDeleteRange() (supported bool)
    // Name gets the name of the storage engine
    Name() string
    // Describe returns of brief introduction of the storage
    Describe() string
    // ShowStatus returns the specified status of the storage
    ShowStatus(ctx context.Context, key string) (interface{}, error)
}
```

- Provide service for application in other browsers via p2p protocol (eg. WebRTC)

- Push PD and TiKV into WebAssembly world

PingCAP

PingCAP

Thank You !