СБЕР

Когда нельзя, но очень хочется? GO!

Лазаренков Егор
Исполнительный директор, SberInfra

О спикере

Егор Лазаренков

Разработал свою первую ERP систему на Delphi 7 и MS SQL 2000 **18 лет назад** и с того времени не перестаю изучать новые технологии, языки программирования и писать коммерческий код.

- Лидер Golang community
- Исполнительный директор в SberInfra
  Управление развития облачных решений

SBER INFRA

golang community #SBERTEAM

Чем может быть полезно?

- Упрощает разработку Unit тестов, аналог mockito в Java
- Добавление нового функционала в стороннюю библиотеку, которая не поддерживает его
- Удаление из исполняемых файлов опасного и неиспользуемого кода
- Понять, как взламываются программы и разрабатывать более защищенные приложения с учетом этого

О чем речь?

	Языки с указателями	Языки без указателей
Примеры языков	C, C++, C#, Go, Rust, Swift	Python, Java, Kotlin
Правки оперативной памяти в обход ООП и прочих правил используемого языка		Получаем прямой доступ к памяти: • Используем низкоуровневые интерфейсы, например **JNI** • Либо используем специальные библиотеки, например **sun.misc.unsafe**
Правки исполняемого файла ABI без исходников		Компиляция JIT: 1. Декомпилируем в исходники - Dex2jar, Java Decompiler 2. Правим и собираем снова – ApkTool Компиляция AOT: • **GNU Compiler for Java** • **GraalVM** • **Cython**

Работаем с БД Postgres

```go
package examples

import (
    "context"
    "testing"

    "github.com/jackc/pgx/v5"
    "github.com/stretchr/testify/require"
)

var ctx = context.Background()

const connStr = "postgresql://postgres:*****@tvlds-nimbs0107.delta.sbrf.ru"

run test | debug test
func TestConnect(t *testing.T) {
    conn, err := pgx.Connect(ctx, connStr)
    require.NoError(t, err)

    defer conn.Close(ctx)

    rows, err := conn.Query(ctx, "SELECT 1")
    require.NoError(t, err)
    rows.Close()
}
```

```
PASS
Process 62707 has exited with status 0
```

В поисках трейсера

```go
// Connect establishes a connection with a PostgreSQL server with a connection string. See
// pgconn.Connect for details.
func Connect(ctx context.Context, connString string) (*Conn, error) {
    connConfig, err := ParseConfig(connString)
    if err != nil {
        return nil, err
    }
    return connect(ctx, connConfig)
}

func ParseConfig(connString string) (*ConnConfig, error) {
    return ParseConfigWithOptions(connString, ParseConfigOptions{})
}

// ConnConfig contains all the options used to establish a connection. It must be created by ParseConfig and
// then it can be modified. A manually initialized ConnConfig will cause ConnectConfig to panic.
type ConnConfig struct {
    pgconn.Config

    Tracer QueryTracer

    // Original connection string that was parsed into config.
    connString string

    // StatementCacheCapacity is maximum size of the statement cache used when executing a query with "cache_statement"
    // query exec mode.
    StatementCacheCapacity int
```

И где же ты?

```go
func ParseConfigWithOptions(connString string, options ParseConfigOptions) (*ConnConfi
    config, err := pgconn.ParseConfigWithOptions(connString, options.ParseConfigOptior
    if err != nil {
        return nil, err
    }

    statementCacheCapacity := 512
    if s, ok := config.RuntimeParams["statement_cache_capacity"]; ok {
        delete(config.RuntimeParams, "statement_cache_capacity")
        n, err := strconv.ParseInt(s, 10, 32)
        if err != nil {
            return nil, fmt.Errorf("cannot parse statement_cache_capacity: %w", err)
        }
        statementCacheCapacity = int(n)
    }

    descriptionCacheCapacity := 512
    if s, ok := config.RuntimeParams["description_cache_capacity"]; ok {
        delete(config.RuntimeParams, "description_cache_capacity")
        n, err := strconv.ParseInt(s, 10, 32)
        if err != nil {
            return nil, fmt.Errorf("cannot parse description_cache_capacity: %w", err)
        }
        descriptionCacheCapacity = int(n)
    }
```

```go
defaultQueryExecMode := QueryExecModeCacheStatement
if s, ok := config.RuntimeParams["default_query_exec_mode"]; ok {
    delete(config.RuntimeParams, "default_query_exec_mode")
    switch s {
    case "cache_statement":
        default
    case "cache
        default
    case "descr
        default
    case "exec"
        default
    case "simpl
        default
    default:
        return
    }
}

connConfig := &ConnConfig{
    Config:                 *config,
    createdByParseConfig:   true,
    StatementCacheCapacity:   statementCacheCapacity,
    DescriptionCacheCapacity: descriptionCacheCapacity,
    DefaultQueryExecMode:     defaultQueryExecMode,
    connString:               connString,
}
```

```go
type ConnConfig struct {
    pgconn.Config

    Tracer QueryTracer

    // Original connection string that was parsed in
    connString string

    // StatementCacheCapacity is maximum size of the
    // query exec mode.
    StatementCacheCapacity int

    // DescriptionCacheCapacity is the maximum size
    // "cache_describe" query exec mode
```

```go
// ConnectWithOptions behaves exactly like Connect with the addition of options. At the present options is only used to
// provide a GetSSLPassword function.
func ConnectWithOptions(ctx context.Context, connString string, options ParseConfigOptions) (*Conn, error) {
    connConfig, err := ParseConfigWithOptions(connString, options)
    if err != nil {
        return nil, err
    }
    return connect(ctx, connConfig)
}
// ConnectConfig establishes a connection with a PostgreSQL server with a configuration struct.
// connConfig must have been created by ParseConfig.
func ConnectConfig(ctx context.Context, connConfig *ConnConfig) (*Conn, error) {
    // In general this improves safety. In particular avoid the config.Config.OnNotification mutation from affecting other
    // connections with the same config. See https://github.com/jackc/pgx/issues/618.
    connConfig = connConfig.Copy()

    return connect(ctx, connConfig)
}
// New creates a new Pool. See [ParseConfig] for information on connString format.
func New(ctx context.Context, connString string) (*Pool, error) {
    config, err := ParseConfig(connString)
    if err != nil {
        return nil, err
    }

    return NewWithConfig(ctx, config)
}

func ParseConfig(connString string) (*ConnConfig, error) {
    return ParseConfigWithOptions(connString, ParseConfigOptions{})
}
```

Кто умеет работать с трейсером?

```
173  // NewWithConfig creates a new Pool. config must have been created by [ParseConfig].
174  func NewWithConfig(ctx context.Context, config *Config) (*Pool, error) {
175      // Default values are set in ParseConfig. Enforce initial creation by ParseConfig rather than setting defaults from
176      // zero values.
177      if !config.createdByParseConfig {
178          panic("config must be created by ParseConfig")
179      }
180
198      var err error
199      p.p, err = puddle.NewPool(
200          &puddle.Config[*connResource]{
201              Constructor: func(ctx context.Context) (*connResource, error) {
202                  atomic.AddInt64(&p.newConnsCount, 1)
203                  connConfig := p.config.ConnConfig.Copy()
204
205                  // Connection will continue in background even if Acquire is canceled. Ensure that a connect won't hang forever.
206                  if connConfig.ConnectTimeout <= 0 {
207                      connConfig.ConnectTimeout = 2 * time.Minute
208                  }
209
210                  if p.beforeConnect != nil {
211                      if err := p.beforeConnect(ctx, connConfig); err != nil {
212                          return nil, err
213                      }
214                  }
215
216                  conn, err := pgx.ConnectConfig(ctx, connConfig)
217                  if err != nil {
218                      return nil, err
219                  }
220
221                  if p.afterConnect != nil {
222                      err = p.afterConnect(ctx, conn)
223                      if err != nil {
224                          conn.Close(ctx)
225                          return nil, err
226                      }
227                  }
```

Опять ты?!

Кого патчить будем?

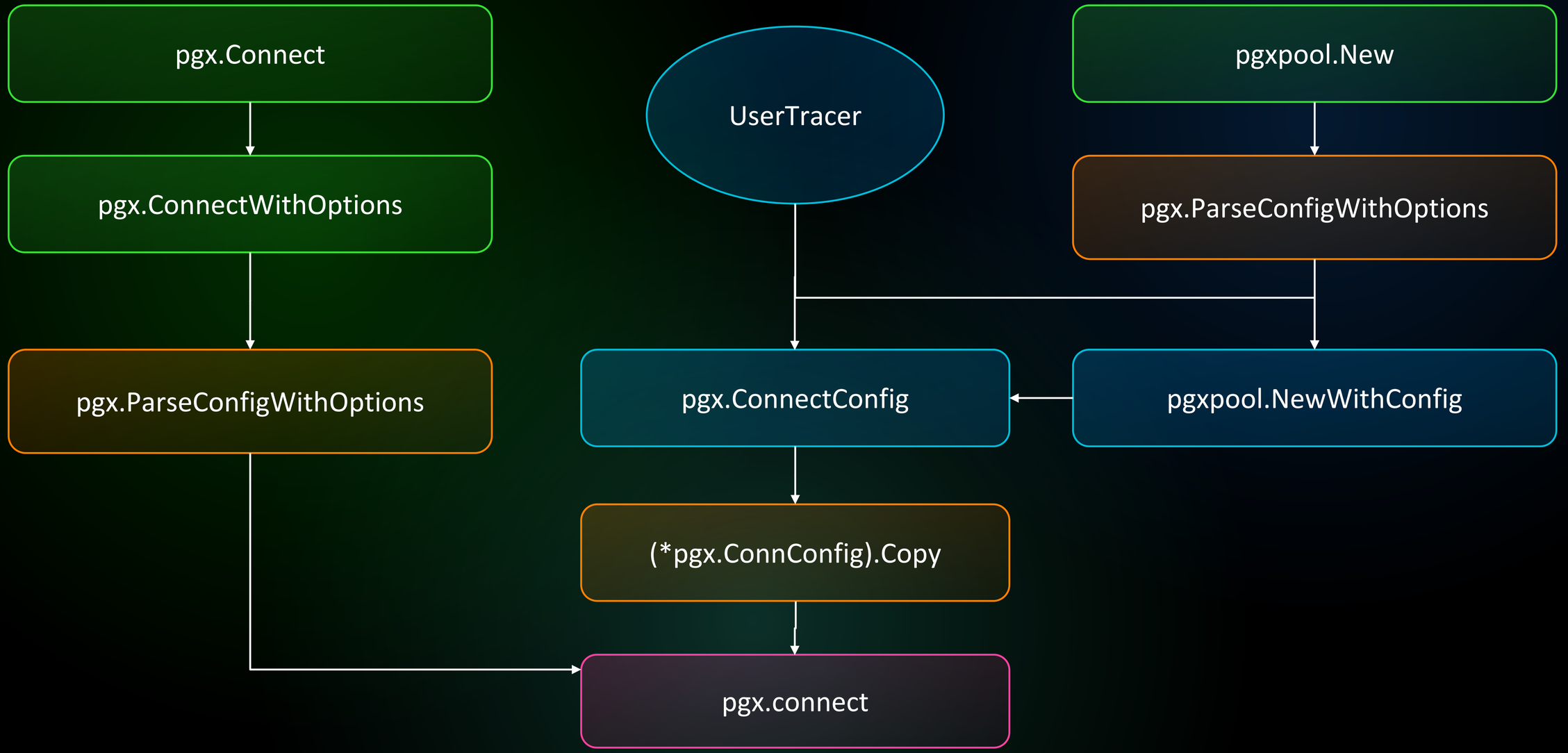


Таблица символов исполняемого файла

ОС	Команда вывода	Парсер в Go sdk
Linux	readelf -St –W file	pkg.go.dev/debug/elf
MacOS	gobjdump -t file	pkg.go.dev/debug/macho
Windows	dumpbin /symbols file	pkg.go.dev/debug/pe

Какое у Вас полное имя?

```
20047779@cab-wsm-0092006 examples % go test -c .
20047779@cab-wsm-0092006 examples % ls
examples.test   main_test.go    pgxtracer       querylogger
20047779@cab-wsm-0092006 examples % gobjdump -t examples.test | grep connect
000000000124f500 l       0e SECT   01 0000 [.text] _crypto/tls.(*Conn).connectionStateLocked
00000000013975a0 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5.connect
0000000001397cc0 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5.connect.func1
0000000001280e20 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.(*connectError).Error
0000000001281000 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.(*connectError).Unwrap
00000000012831a0 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect
0000000001284c00 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect.func5
0000000001284d60 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect.newContextWatcher.func1
0000000001284d20 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect.newContextWatcher.func2
0000000001284ca0 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect.newContextWatcher.func3
0000000001284c60 l       0e SECT   01 0000 [.text] _github.com/jackc/pgx/v5/pgconn.connect.newContextWatcher.func4
0000000001535420 l       0e SECT   03 0000 [__TEXT.__rodata] _go:itab.*github.com/jackc/pgx/v5/pgconn.connectError,error
0000000001201640 l       0e SECT   01 0000 [.text] _net.(*netFD).connect
00000000012022c0 l       0e SECT   01 0000 [.text] _net.(*netFD).connect.func1
00000000012021a0 l       0e SECT   01 0000 [.text] _net.(*netFD).connect.func2
0000000001202400 l       0e SECT   01 0000 [.text] _net.(*netFD).connect.func3
000000000182d880 l       0e SECT   0b 0000 [.data] _net.connectFunc
0000000001078720 l       0e SECT   01 0000 [.text] _syscall.connect
000000000107a100 l       0e SECT   01 0000 [.text] _syscall.libc_connect_trampoline.abi0
000000000128bf40 l       0e SECT   01 0000 [.text] _type:.eq.github.com/jackc/pgx/v5/pgconn.connectError
0000000000000000 g       01 UND    00 0000 _connect
```

```go
package pgxtracer

import (
    "context"
    "sync/atomic"

    "github.com/jackc/pgx/v5"

    _ "unsafe"

    "gitlab.ocp.delta.sbrf.ru/godev/core.git/function"
)

var (
    globalTracer  atomic.Pointer[pgx.QueryTracer]
    originConnect func(ctx context.Context, config *pgx.ConnConfig) (c *pgx.Conn, err error)
)

//go:linkname pgxConnect github.com/jackc/pgx/v5.connect
func pgxConnect(ctx context.Context, config *pgx.ConnConfig) (c *pgx.Conn, err error)

func newConnect(ctx context.Context, config *pgx.ConnConfig) (c *pgx.Conn, err error) {
    if pTracer := globalTracer.Load(); pTracer != nil {
        if config.Tracer == nil {
            config.Tracer = *pTracer
        } else {
            config.Tracer = QueryTracerList{*pTracer, config.Tracer}
        }
    }

    return originConnect(ctx, config)
}

func init() {
    function.Replace(pgxConnect, newConnect, &originConnect)
}

func SetGlobalTracer(tracer pgx.QueryTracer) {
    globalTracer.Store(&tracer)
}
```

Знакомьтесь, это Ваша новая реализация

```go
// QueryTracer traces Query, QueryRow, and Exec.
type QueryTracer interface {
    // TraceQueryStart is called at the beginning of Query, QueryRow, and Exec calls. The returned context is used for the
    // rest of the call and will be passed to TraceQueryEnd.
    TraceQueryStart(ctx context.Context, conn *Conn, data TraceQueryStartData) context.Context

    TraceQueryEnd(ctx context.Context, conn *Conn, data TraceQueryEndData)
}
```

Трейсы будем писать в лог

```go
package querylogger

import (
    "context"
    "log"
    "sync/atomic"

    "github.com/jackc/pgx/v5"
)

type QueryLogger struct {
    QueryNumber uint64
}

const queryNumberKey = "query-number"

func (s *QueryLogger) TraceQueryStart(ctx context.Context, conn *pgx.Conn, data pgx.TraceQueryStartData) context.Context {
    number := atomic.AddUint64(&s.QueryNumber, 1)

    log.Println("start", number, "query:", data.SQL)

    return context.WithValue(ctx, queryNumberKey, number)
}

func (s *QueryLogger) TraceQueryEnd(ctx context.Context, conn *pgx.Conn, data pgx.TraceQueryEndData) {
    log.Println("end", ctx.Value(queryNumberKey), "query")
}
```

Подключаем наш трейс-логгер

```go
package examples

import (
    "context"
    "sync"
    "testing"

    "github.com/jackc/pgx/v5"
    "github.com/jackc/pgx/v5/pgxpool"
    "github.com/stretchr/testify/require"

    "gitlab.ocp.delta.sbrf.ru/godev/grpcmock.git/examples/pgxtracer"
    "gitlab.ocp.delta.sbrf.ru/godev/grpcmock.git/examples/querylogger"
)

var ctx = context.Background()

const connStr = "postgresql://postgres:******@tvlds-nimbs0107.delta.sbrf.ru"

func init() {
    pgxtracer.SetGlobalTracer(&querylogger.QueryLogger{})
}

run test | debug test
func TestConnect(t *testing.T) {
    conn, err := pgx.Connect(ctx, connStr)
    require.NoError(t, err)

    defer conn.Close(ctx)

    rows, err := conn.Query(ctx, "SELECT 1")
    require.NoError(t, err)
    rows.Close()
}
```

```
2023/10/02 14:49:40 start 1 query: SELECT 1
2023/10/02 14:49:40 end 1 query
PASS
Process 37714 has exited with status 0
```

```go
func TestPool(t *testing.T) {
    pool, err := pgxpool.New(ctx, connStr)
    require.NoError(t, err)

    defer pool.Close()

    var wg sync.WaitGroup

    wg.Add(100)
    for i := 0; i < 100; i++ {
        go func() {
            defer wg.Done()
            rows, err := pool.Query(ctx, "SELECT 2")
            require.NoError(t, err)
            rows.Close()
        }()
    }

    wg.Wait()
}
```

```
2023/10/02 14:50:48 end 79 query
2023/10/02 14:50:48 end 80 query
2023/10/02 14:50:48 start 91 query: SELECT 2
2023/10/02 14:50:48 start 92 query: SELECT 2
2023/10/02 14:50:48 end 82 query
2023/10/02 14:50:48 end 81 query
2023/10/02 14:50:48 start 93 query: SELECT 2
2023/10/02 14:50:48 start 94 query: SELECT 2
2023/10/02 14:50:48 end 84 query
2023/10/02 14:50:48 start 95 query: SELECT 2
2023/10/02 14:50:48 end 86 query
2023/10/02 14:50:48 end 83 query
2023/10/02 14:50:48 start 97 query: SELECT 2
2023/10/02 14:50:48 start 96 query: SELECT 2
2023/10/02 14:50:48 end 85 query
2023/10/02 14:50:48 end 87 query
2023/10/02 14:50:48 start 98 query: SELECT 2
2023/10/02 14:50:48 end 88 query
2023/10/02 14:50:48 end 91 query
2023/10/02 14:50:48 start 99 query: SELECT 2
2023/10/02 14:50:48 start 100 query: SELECT 2
2023/10/02 14:50:48 end 89 query
2023/10/02 14:50:48 end 90 query
2023/10/02 14:50:48 end 92 query
2023/10/02 14:50:48 end 94 query
2023/10/02 14:50:48 end 93 query
2023/10/02 14:50:48 end 95 query
2023/10/02 14:50:48 end 96 query
2023/10/02 14:50:48 end 99 query
2023/10/02 14:50:48 end 97 query
2023/10/02 14:50:48 end 100 query
2023/10/02 14:50:48 end 98 query
PASS
Process 37815 has exited with status 0
```

Победа!

```go
func TestConnectConfig(t *testing.T) {

    config := pgx.ConnConfig{
        Config: pgconn.Config{
            Host:     "tvlds-nimbs0107.delta.sbrf.ru",
            Port:     5432,
            Database: "postgres",
            User:     "postgres",
            Password: "*****",
        },
        Tracer: nil,
    }

    conn, err := pgx.ConnectConfig(ctx, &config)
    require.NoError(t, err)

    defer conn.Close(ctx)

    rows, err := conn.Query(ctx, "SELECT 1")
    require.NoError(t, err)
    rows.Close()
}
```

Попробуем передать свой конфиг

```
--- FAIL: TestConnectConfig (0.00s)
panic: config must be created by ParseConfig [recovered]
        panic: config must be created by ParseConfig
```

```go
// connect connects to a database. connect takes ownership of config. The caller must not use
func connect(ctx context.Context, config *ConnConfig) (c *Conn, err error) {
    if connectTracer, ok := config.Tracer.(ConnectTracer); ok {
        ctx = connectTracer.TraceConnectStart(ctx, TraceConnectStartData{ConnConfig: config})
        defer func() {
            connectTracer.TraceConnectEnd(ctx, TraceConnectEndData{Conn: c, Err: err})
        }()
    }

    // Default values are set in ParseConfig. Enforce initial creation by ParseConfig rather
    // zero values.
    if !config.createdByParseConfig {
        panic("config must be created by ParseConfig")
    }

func ParseConfig(connString string) (*Config, error) {
    connConfig, err := pgx.ParseConfig(connString)
    if err != nil {
        return nil, err
    }

    config := &Config{
        ConnConfig:           connConfig,
        createdByParseConfig: true,
    }
```

Преступление раскрыто!

Да нас же предупреждали...

```go
type ConnConfig struct {
    pgconn.Config

    Tracer QueryTracer

    // Original connection string that was parsed into config.
    connString string

    // StatementCacheCapacity is maximum size of the statement cache used when executing a query with "cache_statement"
    // query exec mode.
    StatementCacheCapacity int

    // DescriptionCacheCapacity is the maximum size of the description cache used when executing a query with
    // "cache_describe" query exec mode.
    DescriptionCacheCapacity int

    // DefaultQueryExecMode controls the default mode for executing queries. By default pgx uses the extended protocol
    // and automatically prepares and caches prepared statements. However, this may be incompatible with proxies such as
    // PGBouncer. In this case it may be preferrable to use QueryExecModeExec or QueryExecModeSimpleProtocol. The same
    // functionality can be controlled on a per query basis by passing a QueryExecMode as the first query argument.
    DefaultQueryExecMode QueryExecMode

    createdByParseConfig bool // Used to enforce created by ParseConfig rule.
}
```

```go
func TestConnectConfig(t *testing.T) {

    config := pgx.ConnConfig{
        Config: pgconn.Config{
            Host:     "tvlds-nimbs0107.delta.sbrf.ru",
            Port:     5432,
            Database: "postgres",
            User:     "postgres",
            Password: "*****",
        },
        Tra
}

    config.createdByParseConfig = true

    conn, err := pgx.ConnectConfig(ctx, &config)
    require.NoError(t, err)

    defer conn.Close(ctx)

    rows, err := conn.Query(ctx, "SELECT 1")
    require.NoError(t, err)
    rows.Close()
}
```

config.createdByParseConfig undefined (type pgx.ConnConfig has no field or method createdByParseConfig) compiler(MissingFieldOrMethod)

View Problem (⌥F8) No quick fixes available

И что, прям вообще никак?

И так тоже нельзя?!

```
reflect.ValueOf(config).FieldByName("createdByParseConfig").Set(reflect.ValueOf(true))
```

```
panic: reflect: reflect.Value.Set using value obtained using unexported field [recovered]
        panic: reflect: reflect.Value.Set using value obtained using unexported field
```

```go
config := pgx.ConnConfig{
    Config:                pgconn.Config{
        Host: "tvlds-nimbs0107.delta.sbrf.ru",
        Port: 5432,
        Database: "postgres",
        User: "postgres",
        Password: "*****"},
    Tracer:                nil,
    StatementCacheCapacity:   0,
    DescriptionCacheCapacity: 0,
    DefaultQueryExecMode:     0,
}

tp := reflect.ValueOf(config).Type()
for i := 0; i < tp.NumField(); i++ {
    fmt.Println(tp.Field(i).Name, tp.Field(i).Offset)
}
```

```
Config 0
Tracer 216
connString 232
StatementCacheCapacity 248
DescriptionCacheCapacity 256
DefaultQueryExecMode 264
createdByParseConfig 268
```

```go
type ConnConfig struct {
    pgconn.Config

    Tracer QueryTracer

    createdByParseConfig bool // Used to enforce created by ParseConfig rule.
}
```

А где
Вы живете?

```
75      tp := reflect.TypeOf(config.Config)
76      for i := 0; i < tp.NumField(); i++ {
77          field := tp.Field(i)
78          fmt.Println(field.Name, field.Offset)
79      }
```

```
31  type Config struct {
32      Host            string // host (e.g. localhost
33      Port            uint16
34      Database        string
35      User            string
36      Password        string

63      createdByParseConfig bool // Used to e
```

```
Host 0
Port 16
Database 24
User 40
Password 56
TLSConfig 72
ConnectTimeout 80
DialFunc 88
LookupFunc 96
BuildFrontend 104
RuntimeParams 112
KerberosSrvName 120
KerberosSpn 136
Fallbacks 152
ValidateConnect 176
AfterConnect 184
OnNotice 192
OnNotification 200
createdByParseConfig 208
```

И с Вами приятно познакомиться

```go
func TestConnectConfig(t *testing.T) {
    connConfig := pgx.ConnConfig{
        Config: pgconn.Config{
            Host:           "tvlds-nimbs0107.delta.sbrf.ru",
            Port:           5432,
            Database:       "postgres",
            User:           "postgres",
            Password:       "*****",
            TLSConfig:      nil,
            ConnectTimeout: time.Second,
            DialFunc:       (&net.Dialer{KeepAlive: 5 * time.Minute}).DialContext,
            LookupFunc:     net.DefaultResolver.LookupHost,
            BuildFrontend: func(r io.Reader, w io.Writer) *pgproto3.Frontend {
                return pgproto3.NewFrontend(r, w)
            },
        },
        Tracer:                   nil,
        StatementCacheCapacity:   512,
        DescriptionCacheCapacity: 512,
        DefaultQueryExecMode:     pgx.QueryExecModeCacheStatement,
    }

    *(*bool)(unsafe.Pointer(uintptr(unsafe.Pointer(&connConfig)) + 268)) = true
    *(*bool)(unsafe.Pointer(uintptr(unsafe.Pointer(&connConfig.Config)) + 208)) = true

    conn, err := pgx.ConnectConfig(ctx, &connConfig)
    require.NoError(t, err)

    defer conn.Close(ctx)

    rows, err := conn.Query(ctx, "SELECT 1")
    require.NoError(t, err)
    rows.Close()
}
```

А говорили , что нельзя....

```
2023/10/03 11:10:06 start 1 query: SELECT 1
2023/10/03 11:10:06 end 1 query
PASS
Process 24267 has exited with status 0
```

И даже так можно!

```
*(*bool)(reflect.ValueOf(&connConfig).Elem().FieldByName("createdByParseConfig").
    Addr().UnsafePointer()) = true

*(*bool)(reflect.ValueOf(&connConfig.Config).Elem().FieldByName("createdByParseConfig").
    Addr().UnsafePointer()) = true
```

```
2023/10/03 11:20:20 start 1 query: SELECT 1
2023/10/03 11:20:20 end 1 query
PASS
Process 28748 has exited with status 0
```

Смотря, как попросить...

Нельзя	Но можно
Читать изменять/приватные поля объекта	1. Вычисляем адрес поля в памяти, где хранится это поле 2. Кастим полученный указатель в указатель на тип его значения
Читать изменять/приватные поля объекта имеющие приватный тип	1. Определяем размер приватного типа 2. Создаем новый совместимый тип такого же размера 3. Вычисляем адрес поля в памяти, где хранится целевое значение 4. Кастим полученный указатель в указатель на совместимый-тип
Получить доступ к приватному методу или функции другого пакета	1. Находим адрес в памяти, где начинается эта функция при помощи любого из вариантов: symbols table, runtime.moduledata , linkname, disasm caller func 2. Создаем на основе точки входа в функцию переменную с функциональным типом
Изменять неизменяемое. Например иммутабельный string, либо тело функции.	1. Получаем адрес в памяти, где хранится константа, либо код 2. Снимаем защиту от записи со страниц виртуальной памяти, где расположена эта константа 3. Кастим указатель на эту память в массив байт либо любой совместимый тип и изменяем его элементы.

```go
func TestString(t *testing.T) {
    const str = "hello world"

    str[1] = 'X'
    fmt.Println(str)
}
```

```
cannot assign to str[1] (neither addressable nor a map index expression)
const str untyped string = "hello world"
View Problem (⌥F8)    No quick fixes available
```

Ну вот так точно нельзя...

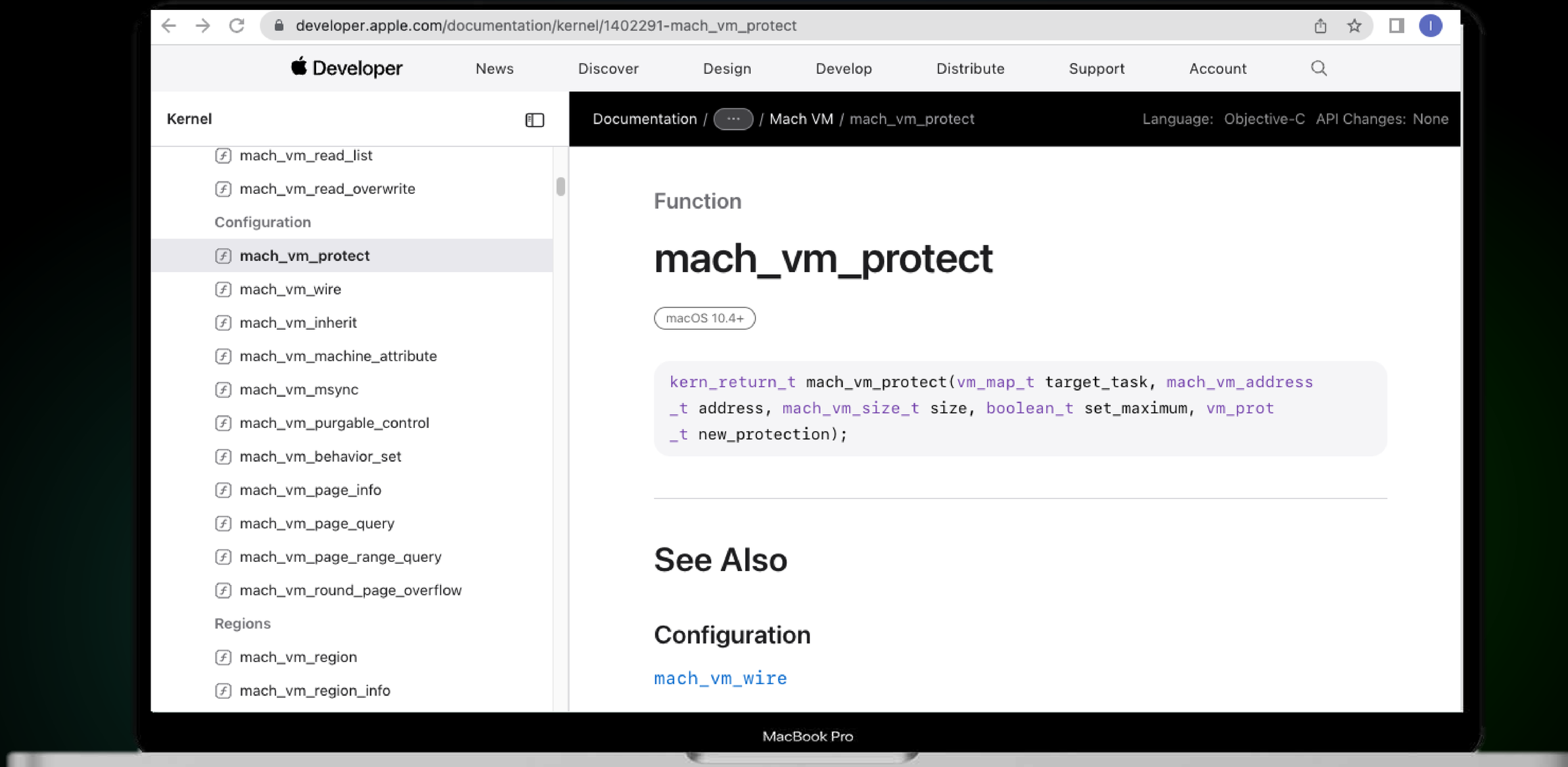
```go
func TestString(t *testing.T) {
    const str string = "hello world"

    stringAsBytes := unsafe.Slice(unsafe.StringData(str), len(str))
    stringAsBytes[1] = byte('X')

    fmt.Println(str)
}
```

```
unexpected fault address 0x1720b47
fatal error: fault
[signal SIGBUS: bus error code=0x2 addr=0x1720b47 pc=0x163f247]
```

Произносим заклинание

```
97  //go:linkname setMemProtect gitlab.ocp.delta.sbrf.ru/godev/core.git/function.setMemProtect
98  func setMemProtect(ptr uintptr, size int, allowWrite bool)
    run test | debug test
99  func TestString(t *testing.T) {
100     const str string = "hello world"
101
102     ptr := unsafe.StringData(str)
103     stringAsBytes := unsafe.Slice(ptr, len(str))
104
105     setMemProtect(uintptr(unsafe.Pointer(ptr)), len(str), true)
106
107     stringAsBytes[1] = 'X'
108     fmt.Println(str)
109 }
```

```
hXllo world
PASS
Process 41205 has exited with status 0
```


developer.apple.com/documentation/kernel/1402291-mach_vm_protect
Developer
News
Discover
Design
Develop
Distribute
Support
Account
Kernel
mach_vm_read_list
mach_vm_read_overwrite
Configuration
mach_vm_protect
mach_vm_wire
mach_vm_inherit
mach_vm_machine_attribute
mach_vm_msync
mach_vm_purgable_control
mach_vm_behavior_set
mach_vm_page_info
mach_vm_page_query
mach_vm_page_range_query
mach_vm_round_page_overflow
Regions
mach_vm_region
mach_vm_region_info
Documentation / ... / Mach VM / mach_vm_protect
Language: Objective-C API Changes: None
Function
mach_vm_protect
macOS 10.4+
kern_return_t mach_vm_protect(vm_map_t target_task, mach_vm_address_t address, mach_vm_size_t size, boolean_t set_maximum, vm_prot_t new_protection);
See Also
Configuration
mach_vm_wire
MacBook Pro

Подключаем СИ

function/protect_darwin.go

```go
//go:build darwin
// +build darwin

package function

/*
#include <mach/mach.h>
*/
import "C"

func setMemProtect(ptr uintptr, size int, allowWrite bool) {
        prot := C.VM_PROT_READ | C.VM_PROT_EXECUTE | C.VM_PROT_COPY

        if allowWrite {
                prot |= C.VM_PROT_WRITE
        }

        if C.vm_protect(C.mach_task_self_, C.ulong(ptr), C.ulong(size), 0, C.int(prot)) != C.KERN_SUCCESS {
                panic("can't setMemProtect memory protect on macos")
        }
}
```

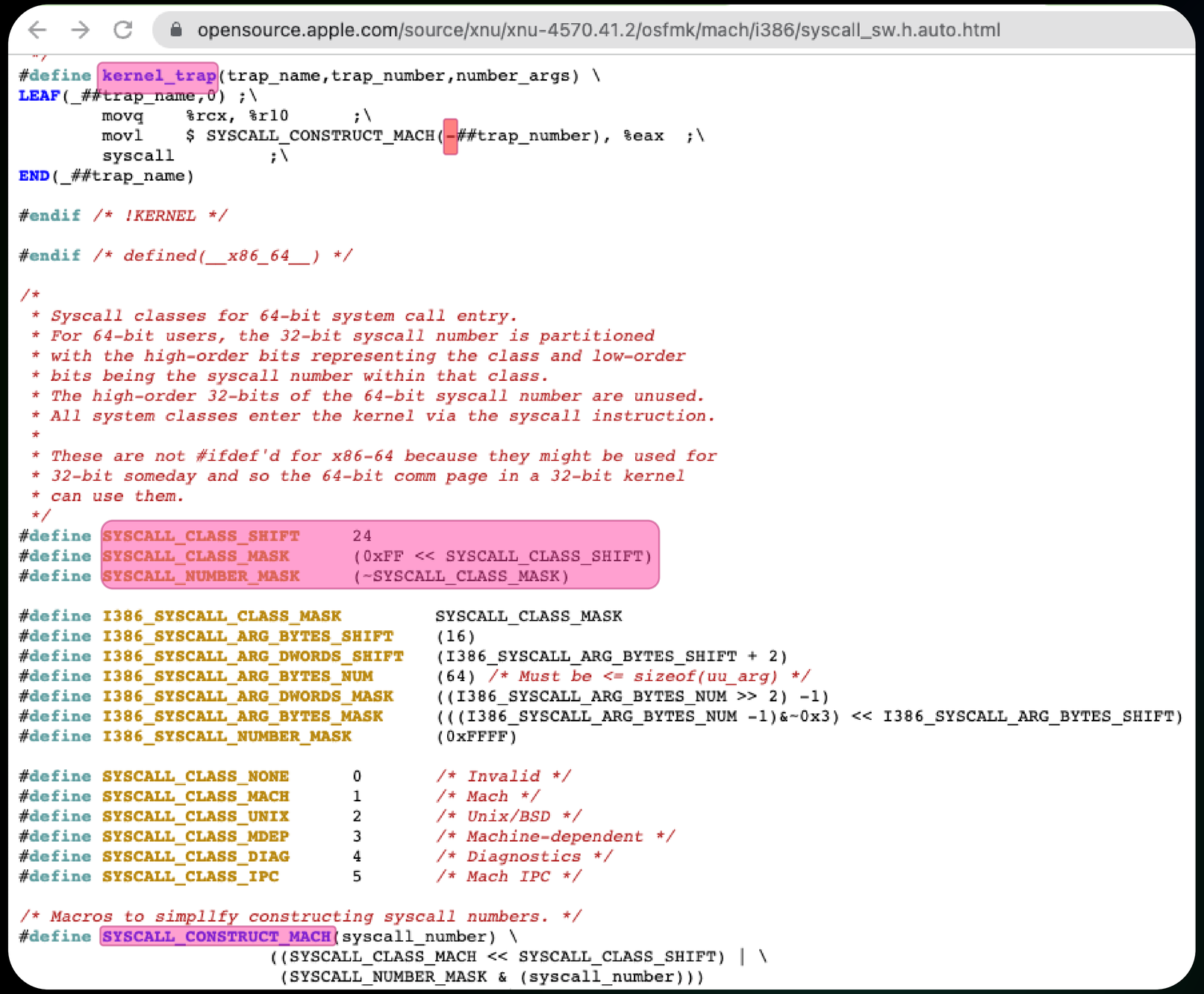
Заглянем в кроличью нору

opensource.apple.com/source/xnu/xnu-7195.50.7.100.1/libsyscall/mach/mach_vm.c.auto.html

```
kern_return_t
mach_vm_protect(
        mach_port_name_t task,
        mach_vm_address_t address,
        mach_vm_size_t size,
        boolean_t set_maximum,
        vm_prot_t new_protection)
{
        kern_return_t rv;

        rv = _kernelrpc_mach_vm_protect_trap(task, address, size, set_maximum,
            new_protection);

        if (rv == MACH_SEND_INVALID_DEST) {
                rv = _kernelrpc_mach_vm_protect(task, address, size,
                    set_maximum, new_protection);
        }

        return rv;
}
```

opensource.apple.com/source/xnu/xnu-6153.11.26/osfmk/mach/mach_traps.h.auto.html

```
struct _kernelrpc_mach_vm_protect_args {
        PAD_ARG_(mach_port_name_t, target);          /* 1 word */
        PAD_ARG_(mach_vm_address_t, address);        /* 2 words */
        PAD_ARG_(mach_vm_size_t, size);              /* 2 words */
        PAD_ARG_(boolean_t, set_maximum);            /* 1 word */
        PAD_ARG_(vm_prot_t, new_protection);         /* 1 word */
};                                                   /* Total: 7 */
extern kern_return_t _kernelrpc_mach_vm_protect_trap(
        struct _kernelrpc_mach_vm_protect_args *args);
```



В поисках системного вы

syscall_number = (1 << 24) | (~(1 << 24) & (-trap_number))

syscall_number = 0x1000000 - trap_number

Ах вот же ты!

opensource.apple.com/source/xnu/xnu-6153.11.26/osfmk/mach/syscall_sw.h.auto.html

```
/*
 * i386 and x86_64 just load of the stack or use
 * registers in order; no munging is required,
 * and number of args is ignored.  ARM loads args
 * into registers beyond r3, unlike the normal
 * procedure call standard; we pad for 64-bit args.
 */
kernel_trap(_kernelrpc_mach_vm_allocate_trap,-10,5) /* 4 args, +1 for mach_vm_size_t */
kernel_trap(_kernelrpc_mach_vm_purgable_control_trap,-11,5) /* 4 args, +1 for mach_vm_offset_t */
kernel_trap(_kernelrpc_mach_vm_deallocate_trap,-12,5) /* 3 args, +2 for mach_vm_size_t and mach_vm_address_t */
kernel_trap(_kernelrpc_mach_vm_protect_trap,-14,7) /* 5 args, +2 for mach_vm_address_t and mach_vm_size_t */
kernel_trap(_kernelrpc_mach_vm_map_trap,-15,9)
kernel_trap(_kernelrpc_mach_port_allocate_trap,-16,3)
kernel_trap(_kernelrpc_mach_port_destroy_trap,-17,2)
kernel_trap(_kernelrpc_mach_port_deallocate_trap,-18,2)
kernel_trap(_kernelrpc_mach_port_mod_refs_trap,-19,4)
kernel_trap(_kernelrpc_mach_port_move_member_trap,-20,3)
kernel_trap(_kernelrpc_mach_port_insert_right_trap,-21,4)
kernel_trap(_kernelrpc_mach_port_insert_member_trap,-22,3)
kernel_trap(_kernelrpc_mach_port_extract_member_trap,-23,3)
kernel_trap(_kernelrpc_mach_port_construct_trap,-24,5)
kernel_trap(_kernelrpc_mach_port_destruct_trap,-25,5)
```

syscall_number = 0x1000000 -(-14) = 0x1000000 + 14 = 0x100000E

```
1  //go:build darwin && amd64
2  // +build darwin,amd64
3
4  #include "go_asm.h"
5  #include "textflag.h"
6
7  // func taskSelfTrap() (ret uint32)
8  TEXT ·taskSelfTrap(SB), $8-0
9      PUSHQ AX
10     MOVL  $(0x1000000+28), AX // task_self_trap
11     SYSCALL
12     MOVL  AX, ret+0(FP)
13     POPQ AX
14     RET
15
16 // func vmProtect(targetTask uint32, address uintptr, size int, setMaximum, newProtection uint32) (ret uint32)
17 TEXT ·vmProtect(SB), $56-40
18     PUSHQ AX
19     PUSHQ DI
20     PUSHQ SI
21     PUSHQ DX
22     PUSHQ R10
23     PUSHQ R8
24     PUSHQ R9
25     MOVL  targetTask+0(FP), DI
26     MOVQ  address+8(FP), SI
27     MOVQ  size+16(FP), DX
28     MOVL  set_maximum+24(FP), R10
29     MOVL  new_protection+28(FP), R8
30     XORQ  R9, R9
31     MOVQ  $(0x1000000+14), AX // mach_vm_protect
32     SYSCALL
33     MOVL  AX, ret+32(FP)
34     POPQ  R9
35     POPQ  R8
36     POPQ  R10
37     POPQ  DX
38     POPQ  SI
39     POPQ  DI
40     POPQ  AX
41     RET
```

Слезаем с зависимостей от СИ

function/protect_darwin_amd64.s

Мы больше не на СИ!

function/protect_darwin.go

```go
//go:build darwin
// +build darwin

package function

import "fmt"

const (
    vmProtRead uint32 = 1 << iota
    vmProtWrite
    vmProtExecute
    _ // vmProtNoChange
    vmProtCopy
)

//go:noescape
func taskSelfTrap() (id uint32)

//go:noescape
func vmProtect(targetTask uint32, address uintptr, size int, setMaximum, newProtection uint32) (ret uint32)

func setMemProtect(ptr uintptr, size int, allowWrite bool) {
    prot := vmProtRead | vmProtExecute | vmProtCopy

    if allowWrite {
        prot |= vmProtWrite
    }

    if retCode := vmProtect(taskSelfTrap(), ptr, size, 0, prot); retCode != 0 {
        panic(fmt.Sprint("can't setMemProtect memory protect on macos, mach_vm_protect return ", retCode))
    }
}
```

```go
package main

import "fmt"

func main() {
    var (
        a = 5
        b = 6
    )

    c := a + b

    fmt.Println(c)
}
```

```
20047779@cab-wsm-0092006 demo % go build .
20047779@cab-wsm-0092006 demo % ls
demo    demo.go
20047779@cab-wsm-0092006 demo % ./demo
11
```

А теперь серьез
задачка...

Знакомьтесь, Ида

IDA - The Interactive Disassembler

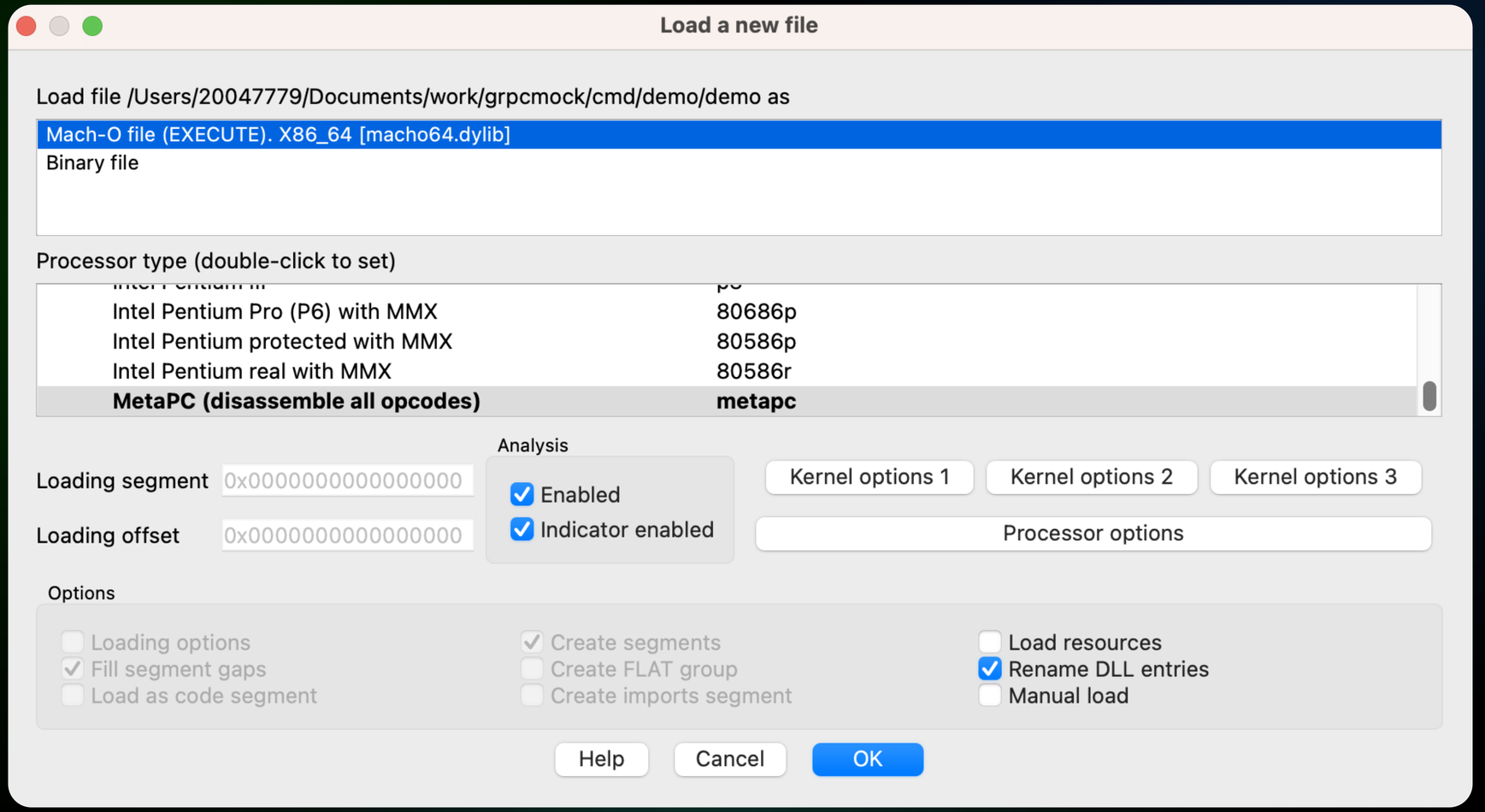
Version 8.2.230124 macOS x86_64 (64-bit address size)

Freeware version with the following limitations:
1. Only for non-commercial use
2. Without technical support
3. Only supports x86/x64 code
4. Only PE/ELF/Mach-O files are supported
5. IDAPython is not available

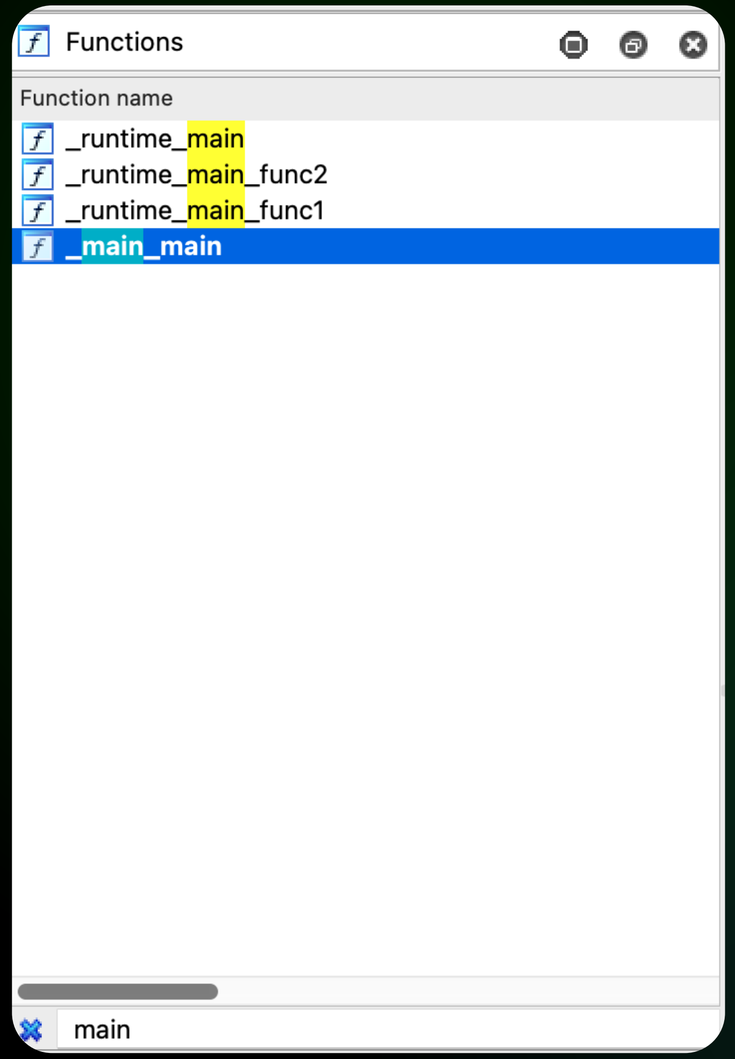
For commercial use please acquire the full version

www.hex-rays.com

Перейдем сразу к делу



Знакомый код

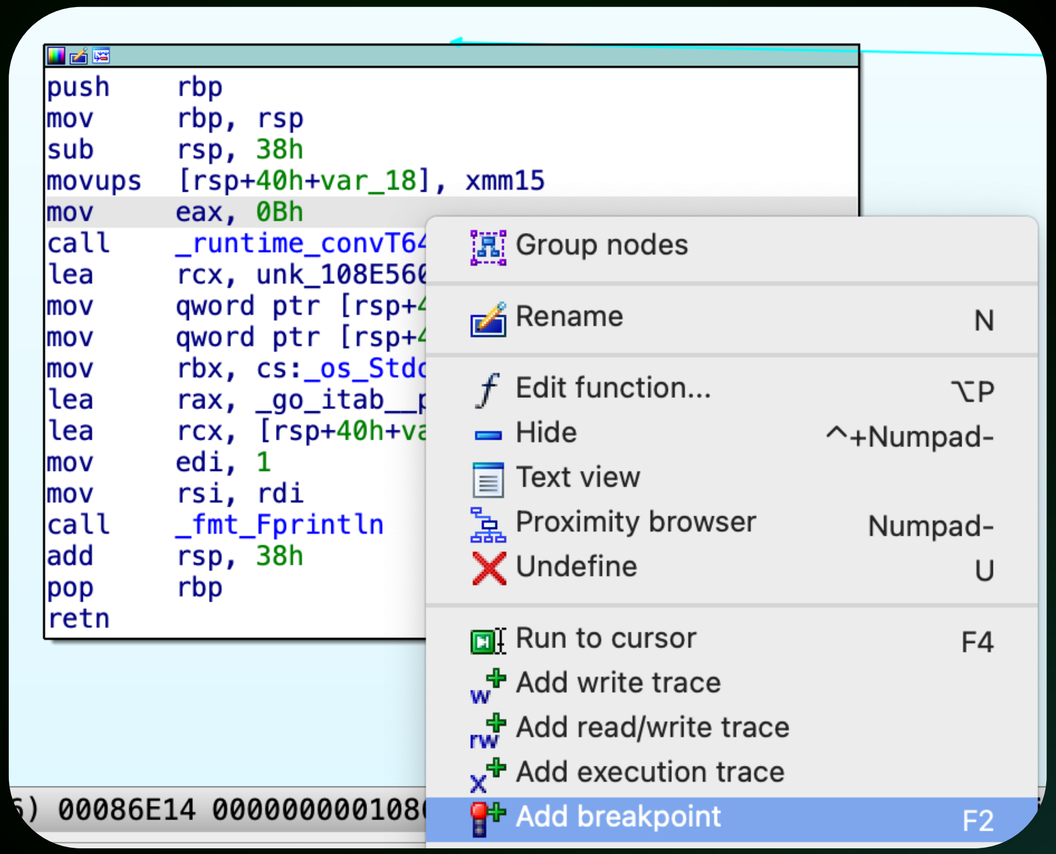


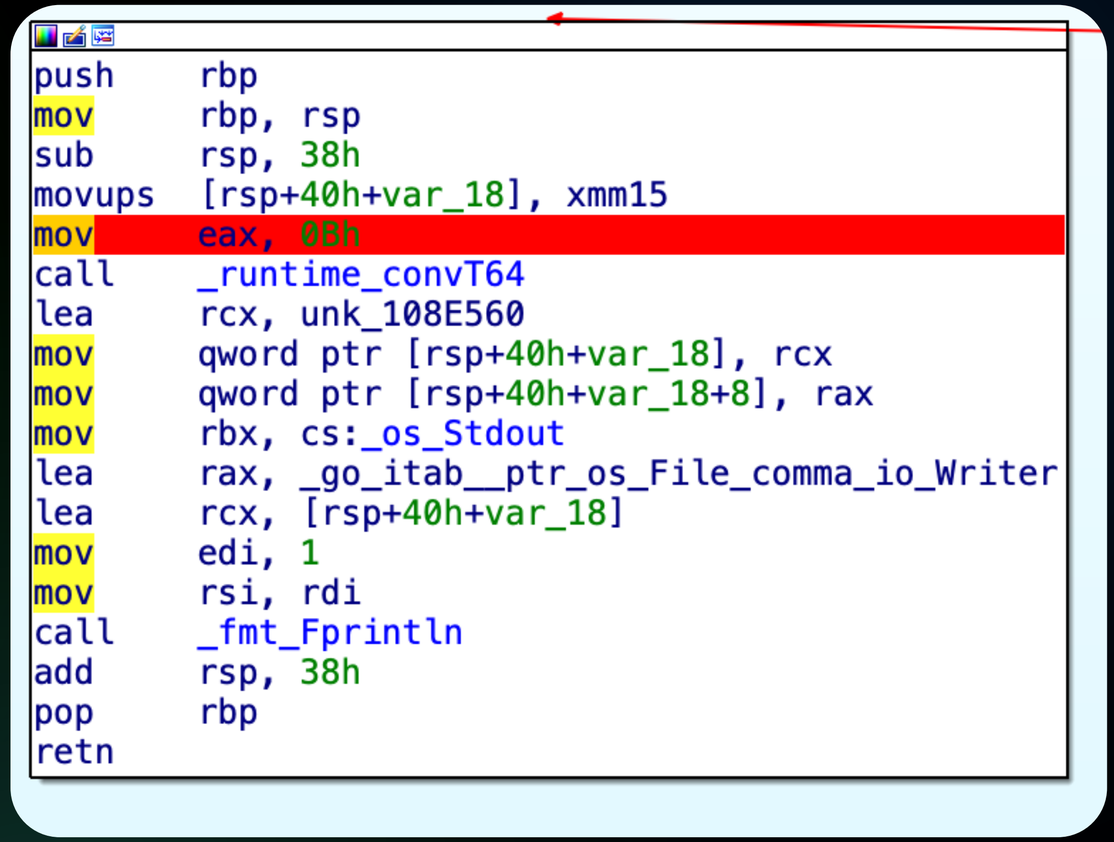

```
; void __cdecl main_main()
_main_main proc near

var_18= xmmword ptr -18h

cmp     rsp, [r14+10h]
jbe     short loc_1086E55
```

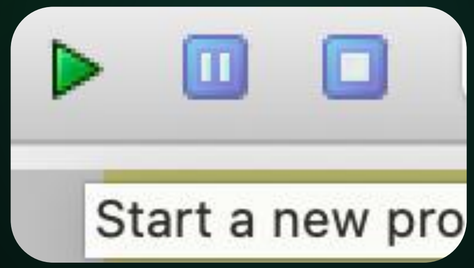
```
push    rbp
mov     rbp, rsp
sub     rsp, 38h
movups  [rsp+40h+var_18], xmm15
mov     eax, 0Bh
call    _runtime_convT64
lea     rcx, unk_108E560
mov     qword ptr [rsp+40h+var_18], rcx
mov     qword ptr [rsp+40h+var_18+8], rax
mov     rbx, cs:_os_Stdout
lea     rax, _go_itab__ptr_os_File_comma_io_Writer
lea     rcx, [rsp+40h+var_18]
mov     edi, 1
mov     rsi, rdi
call    _fmt_Fprintln
add     rsp, 38h
pop     rbp
retn
```

```
loc_1086E55:
call    _runtime_morestack_noctxt_abi0
_main_main endp
```

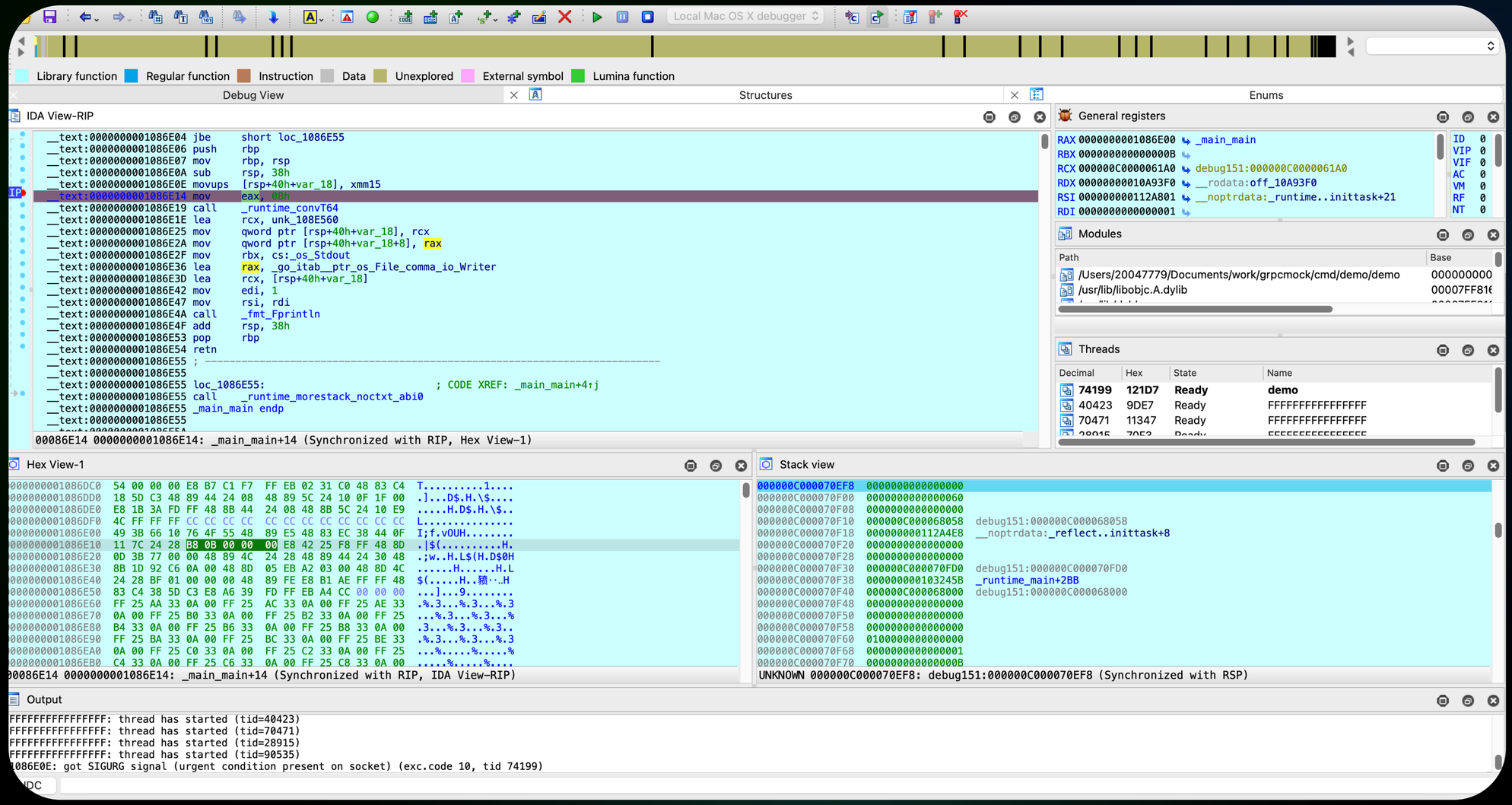
На следующей остановите



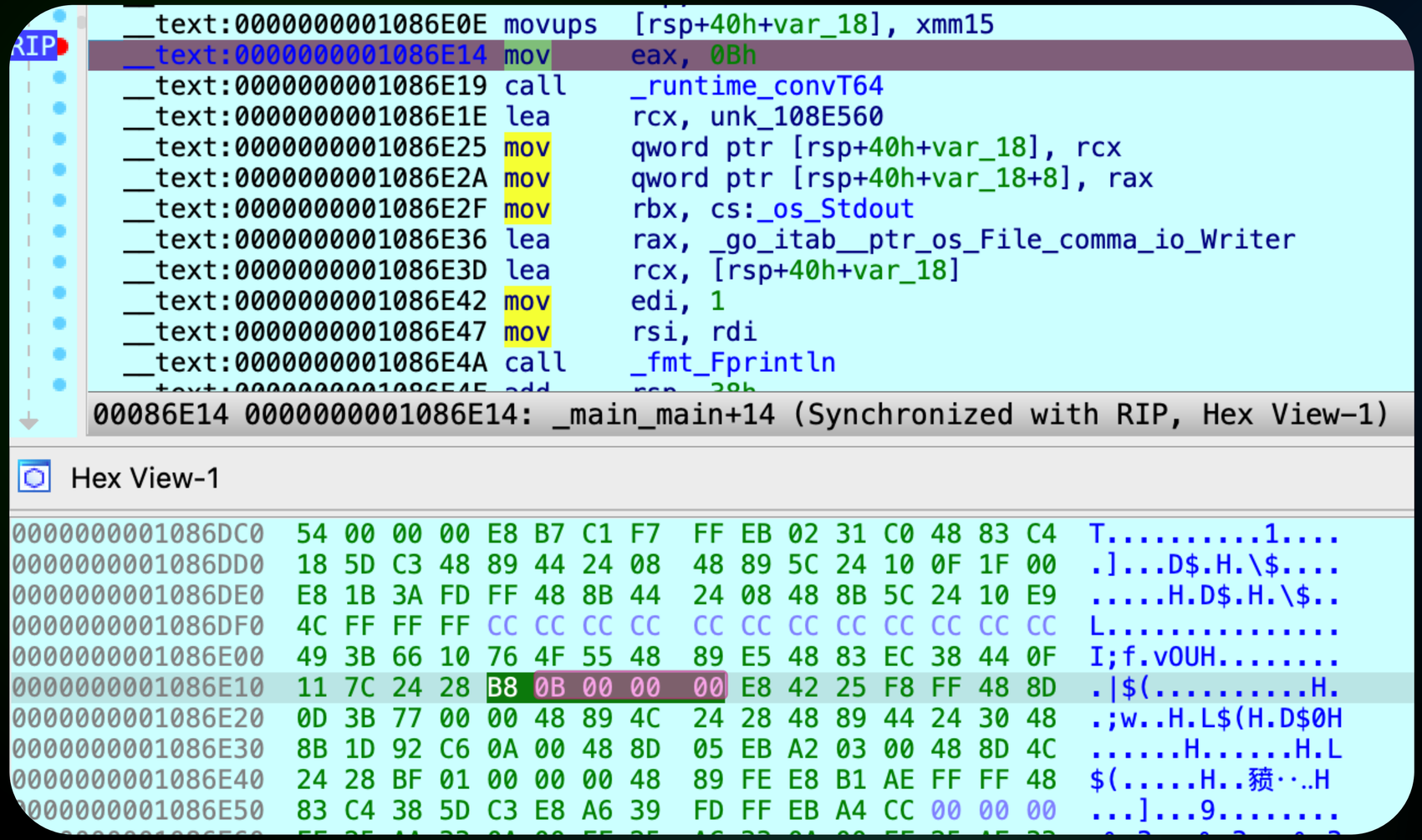




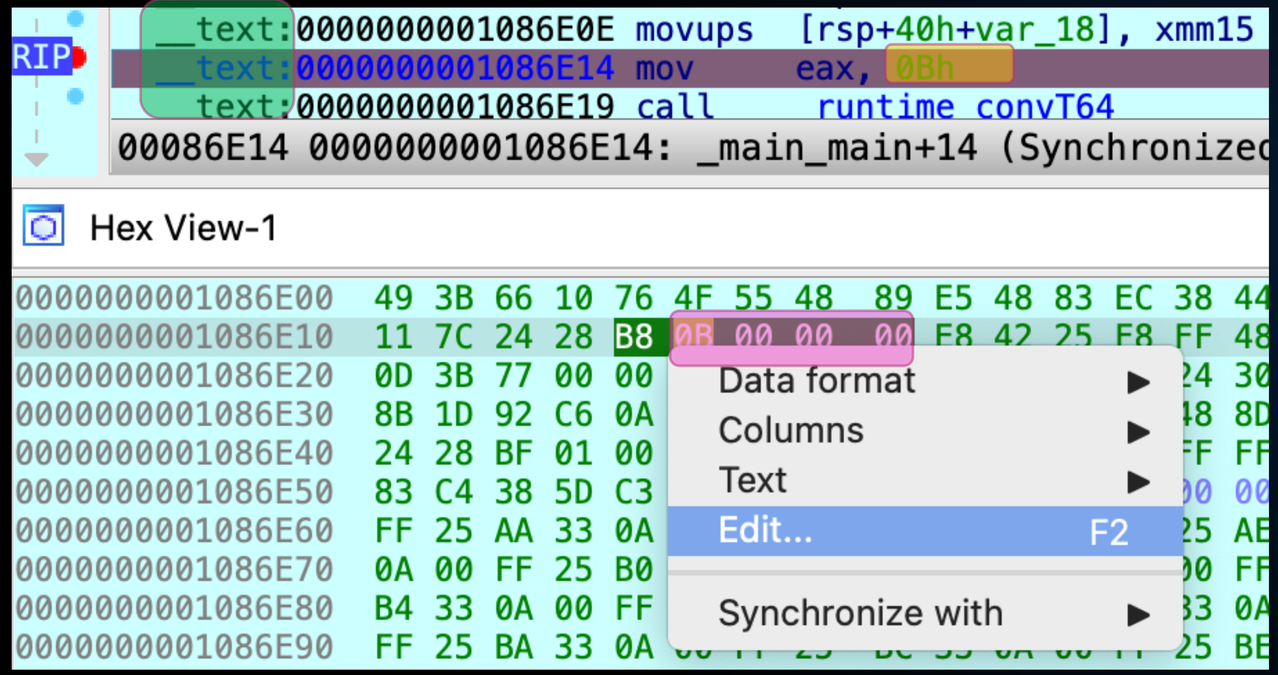
Чего желаете?

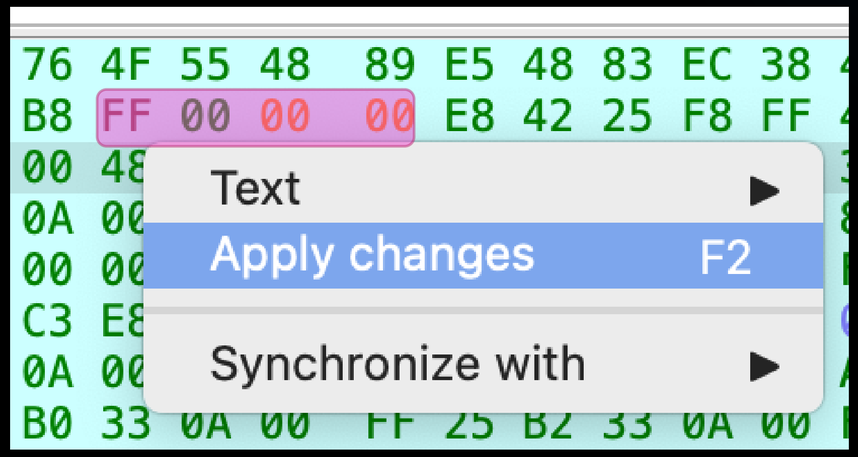


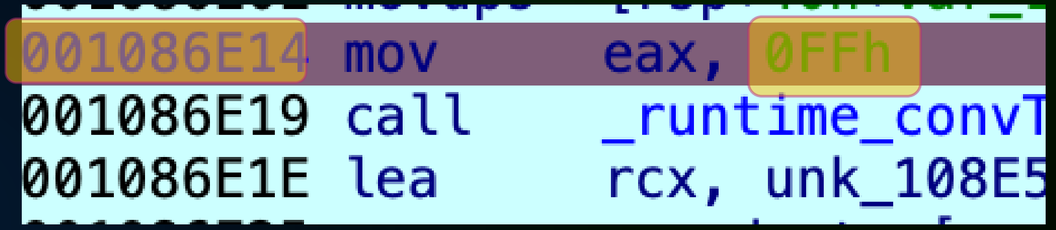
Я хочу...


__text:0000000001086E0E movups [rsp+40h+var_18], xmm15
RIP
__text:0000000001086E14 mov eax, 0Bh
__text:0000000001086E19 call _runtime_convT64
__text:0000000001086E1E lea rcx, unk_108E560
__text:0000000001086E25 mov qword ptr [rsp+40h+var_18], rcx
__text:0000000001086E2A mov qword ptr [rsp+40h+var_18+8], rax
__text:0000000001086E2F mov rbx, cs:_os_Stdout
__text:0000000001086E36 lea rax, _go_itab__ptr_os_File_comma_io_Writer
__text:0000000001086E3D lea rcx, [rsp+40h+var_18]
__text:0000000001086E42 mov edi, 1
__text:0000000001086E47 mov rsi, rdi
__text:0000000001086E4A call _fmt_Fprintln
00086E14 0000000001086E14: _main_main+14 (Synchronized with RIP, Hex View-1)
Hex View-1
0000000001086DC0 54 00 00 00 E8 B7 C1 F7 FF EB 02 31 C0 48 83 C4 T..........1....
0000000001086DD0 18 5D C3 48 89 44 24 08 48 89 5C 24 10 0F 1F 00 .]...D$.H.\$....
0000000001086DE0 E8 1B 3A FD FF 48 8B 44 24 08 48 8B 5C 24 10 E9 .....H.D$.H.\$..
0000000001086DF0 4C FF FF FF CC CC CC CC CC CC CC CC CC CC CC CC L...............
0000000001086E00 49 3B 66 10 76 4F 55 48 89 E5 48 83 EC 38 44 0F I;f.vOUH.....8D.
0000000001086E10 11 7C 24 28 B8 0B 00 00 00 E8 42 25 F8 FF 48 8D .|$(.....B%..H.
0000000001086E20 0D 3B 77 00 00 48 89 4C 24 28 48 89 44 24 30 48 .;w..H.L$(H.D$0H
0000000001086E30 8B 1D 92 C6 0A 00 48 8D 05 EB A2 03 00 48 8D 4C ......H.....H.L
0000000001086E40 24 28 BF 01 00 00 00 48 89 FE E8 B1 AE FF FF 48 $(.....H..猢···H
0000000001086E50 83 C4 38 5D C3 E8 A6 39 FD FF EB A4 CC 00 00 00 ...]...9........

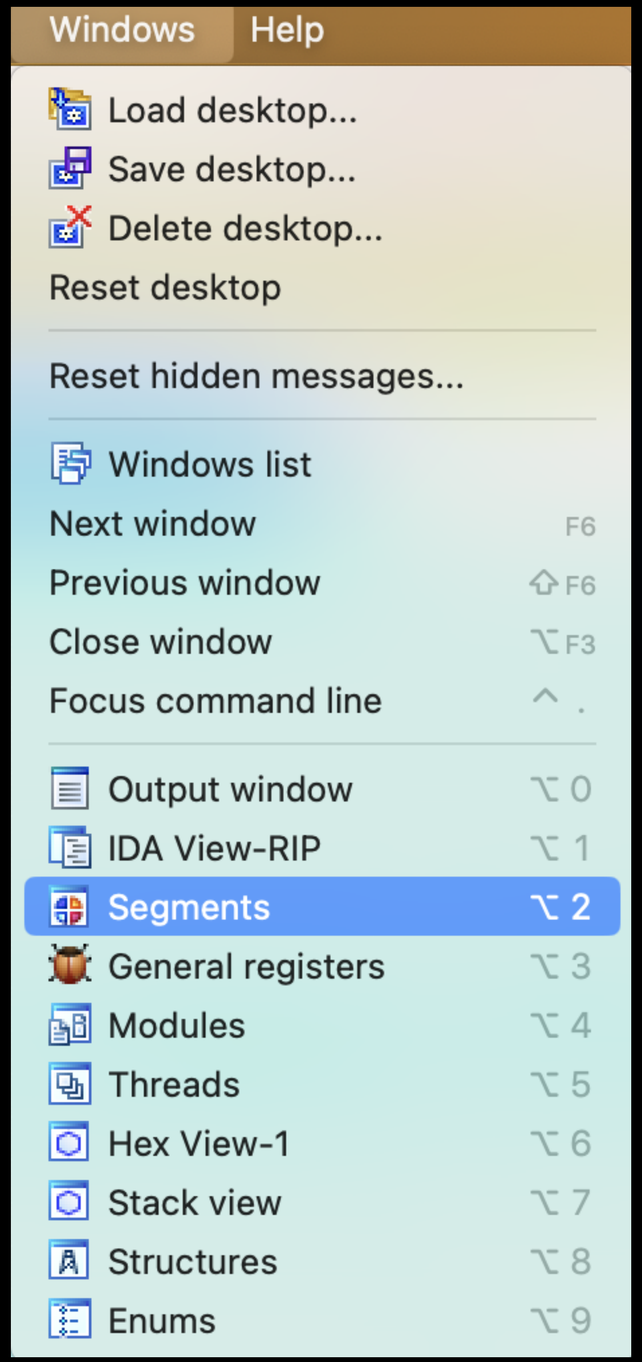
Изменить ответ!








```
20047779@cab-wsm-0092006 ~ % sudo ida64
Password:
qt.qpa.fonts: Populating font family aliases
st.
2023-10-03 17:30:34.317 ida64[95092:1810035]
11
11
qt.qpa.window: Window position QRect(-3,631 1
11
255
4294967295
11
255
```



Name	Start	End
HEADER	0000000001000000	0000000001001000
__text	0000000001001000	0000000001086E60
__symbol_stub1	0000000001086E60	0000000001086FA0
__rodata	0000000001086FA0	00000000010C2CC0
__typelink	00000000010C2CC0	00000000010C3280
__itablink	00000000010C3280	00000000010C32F0
demo:__gosymtab	00000000010C32F0	00000000010C3300
__gopclntab	00000000010C3300	000000000112A000
__go_buildinfo	000000000112A000	000000000112A210

```
20047779@cab-wsm-0092006 demo % gobjdump -h demo_100

demo_100:     file format mach-o-x86-64

Sections:
Idx Name          Size      VMA               LMA               File off  Algn
  0 .text         00085e5d  0000000001001000  0000000001001000  00001000  2**5
                  CONTENTS, ALLOC, LOAD, CODE
  1 __TEXT.__symbol_stub1 00000126  0000000001086e60  0000000001086e60  00086e60  2**5
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 __TEXT.__rodata 0003bd17  0000000001086fa0  0000000001086fa0  00086fa0  2**5
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  3 __TEXT.__typelink 000005a4  00000000010c2cc0  00000000010c2cc0  000c2cc0  2**5
```

OffsetInFile = VMAddress - TextVMAStart + TextFileOffset

= 0x1086E15 – 0x1000000

= 0x86E15

А можно навсегда?

```
20047779@cab-wsm-0092006 demo % printf '\x64' | dd of=demo_100 bs=1 seek=$((0x86E15)) conv=notrunc
1+0 records in
1+0 records out
1 bytes transferred in 0.000292 secs (3425 bytes/sec)
20047779@cab-wsm-0092006 demo % ./demo_100
100
20047779@cab-wsm-0092006 demo % printf '\xFF' | dd of=demo_100 bs=1 seek=$((0x86E15)) conv=notrunc
1+0 records in
1+0 records out
1 bytes transferred in 0.000106 secs (9434 bytes/sec)
20047779@cab-wsm-0092006 demo % ./demo_100
255
```

*Порядок байт на архитектуре x86/x86_64 - **little-endian***

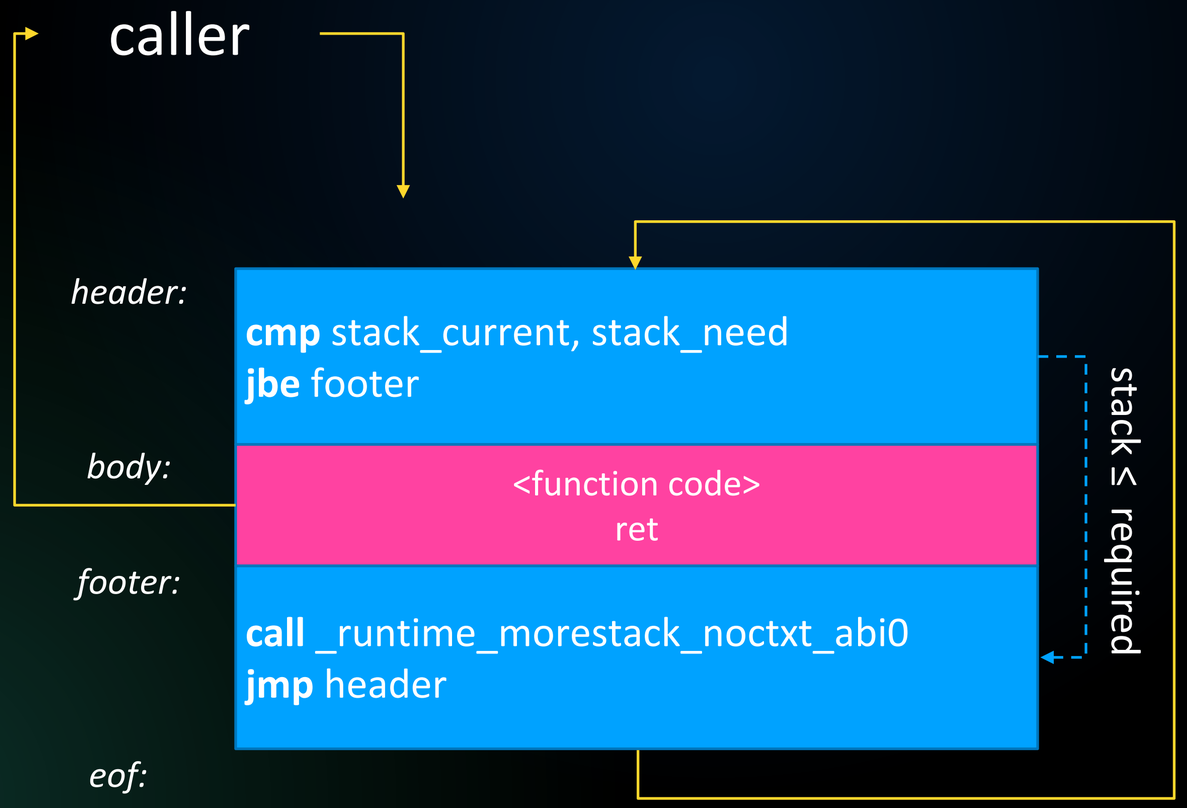
```
20047779@cab-wsm-0092006 demo % printf '\xd2\x02\x96\x49' | dd of=demo_100 bs=1 seek=$((0x86E15)) conv=notrunc
4+0 records in
4+0 records out
4 bytes transferred in 0.023235 secs (172 bytes/sec)
20047779@cab-wsm-0092006 demo % ./demo_100
1234567890
```

Можно!

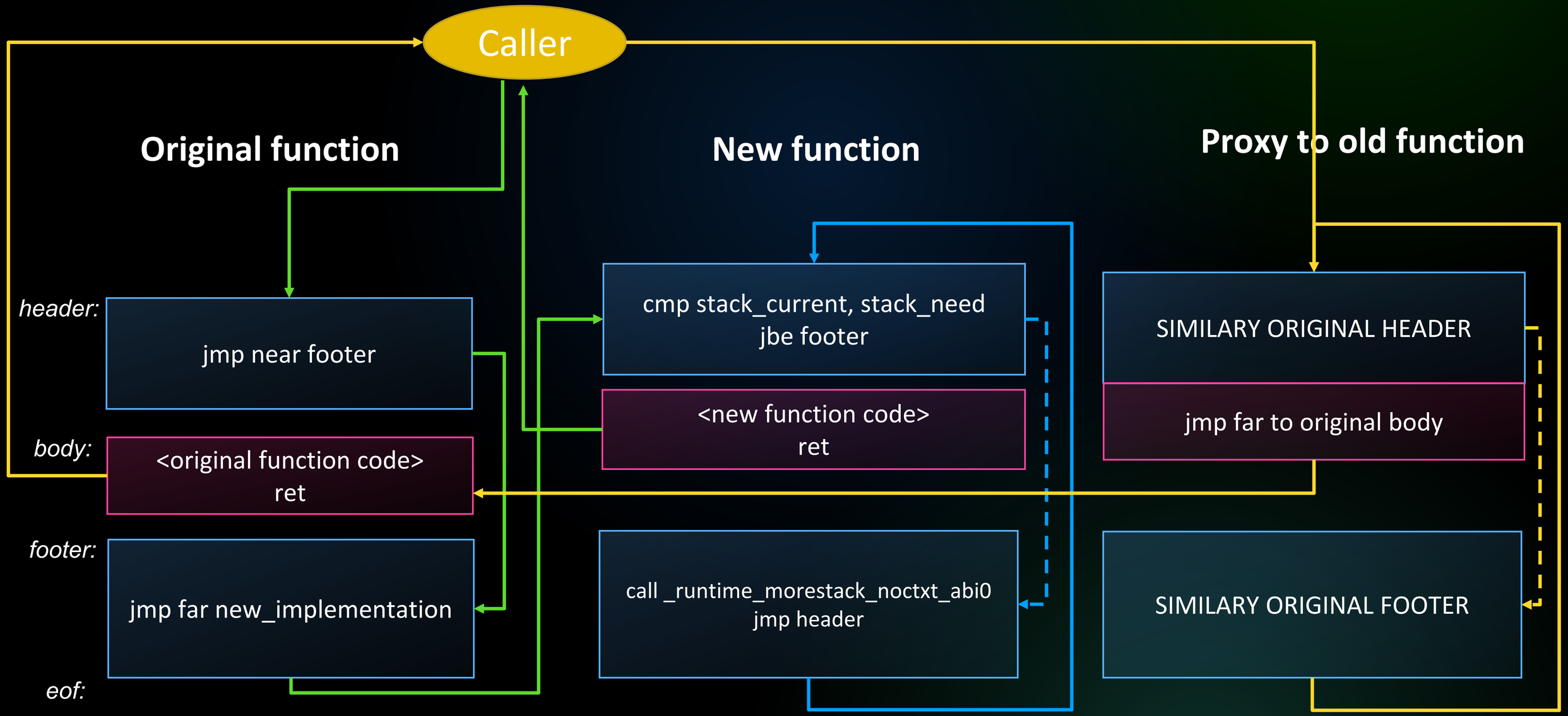
Исследуем устройство Go функции

Ида, подай микроскоп!

```
__text:0000000001086E00 ; void __cdecl main_main()
__text:0000000001086E00 _main_main      proc near               ; CODE XREF: _runt
__text:0000000001086E00                                         ; sub_1086E5A↓j
__text:0000000001086E00                                         ; DATA XREF: ...
__text:0000000001086E00
__text:0000000001086E00 var_18          = xmmword ptr -18h
__text:0000000001086E00
__text:0000000001086E00                 cmp     rsp, [r14+10h]
__text:0000000001086E04                 jbe     short loc_1086E55
__text:0000000001086E06                 push    rbp
__text:0000000001086E07                 mov     rbp, rsp
__text:0000000001086E0A                 sub     rsp, 38h
__text:0000000001086E0E                 movups  [rsp+40h+var_18], xmm15
__text:0000000001086E14                 mov     eax, 499602D2h
__text:0000000001086E19                 call    _runtime_convT64
__text:0000000001086E1E                 lea     rcx, unk_108E560
__text:0000000001086E25                 mov     qword ptr [rsp+40h+var_18], rcx
__text:0000000001086E2A                 mov     qword ptr [rsp+40h+var_18+8], rax
__text:0000000001086E2F                 mov     rbx, cs:_os_Stdout
__text:0000000001086E36                 lea     rax, _go_itab__ptr_os_File_comma_i
__text:0000000001086E3D                 lea     rcx, [rsp+40h+var_18]
__text:0000000001086E42                 mov     edi, 1
__text:0000000001086E47                 mov     rsi, rdi
__text:0000000001086E4A                 call    _fmt_Fprintln
__text:0000000001086E4F                 add     rsp, 38h
__text:0000000001086E53                 pop     rbp
__text:0000000001086E54                 retn
__text:0000000001086E55 ; ---------------------------------------------------------
__text:0000000001086E55
__text:0000000001086E55 loc_1086E55:                            ; CODE XREF: _main_
__text:0000000001086E55                 call    _runtime_morestack_noctxt_abi0
__text:0000000001086E55 _main_main      endp
__text:0000000001086E55
__text:0000000001086E5A
__text:0000000001086E5A ; =============== S U B R O U T I N E =======================
__text:0000000001086E5A
__text:0000000001086E5A
__text:0000000001086E5A sub_1086E5A     proc near
__text:0000000001086E5A                 jmp     short _main_main
__text:0000000001086E5A sub_1086E5A     endp
```



Я кажется придумал...



```go
// Replace replace function to new implementation and returns proxy to old implementation.
func Replace[T any](tg, rp T, oldTo ...*T) *Patch[T] {
    sec, err := inspectFunction(**(**uintptr)(unsafe.Pointer(&tg)))
    if err != nil {
        panic(fmt.Errorf("%w, maybe function already patched", err))
    }

    newFnc := *(*uintptr)(unsafe.Pointer(&rp))
    oldFnc := *(*uintptr)(unsafe.Pointer(&tg))

    beforeJBE := copyBytes(sec.Header, int(sec.JBEAddress-sec.Header))
    beforeCall := copyBytes(sec.Footer, int(sec.CallAddress-sec.Footer))
    afterCall := copyBytes(sec.CallAddress+uintptr(sec.CallInstruction.Len),
        int(sec.JMPAddress-sec.CallAddress-uintptr(sec.CallInstruction.Len)))
    rel, ok := sec.CallInstruction.Args[0].(x86asm.Rel)

    if !ok {
        panic("call of split_stack is not relative")
    }

    newHeader := make([]byte, sec.JBEAddress-sec.Header, sec.Body-sec.Header)
      You, 2 weeks ago • add vendor
    for i := range newHeader {
        newHeader[i] = 0x90
    }

    switch sec.JBEInstruction.Len {
    case 2:
        newHeader = append(newHeader, 0xEB) // JMP SHORT
    case 6:
        newHeader = append(newHeader, 0x90, 0xE9) // JMP NEAR
    default:
        panic(fmt.Sprintf("unsupported JBE instruction %v", sec.JBEInstruction))
    }
```

И даже реализовал

```go
type sections struct {
    Header          uintptr
    Body            uintptr
    Footer          uintptr
    EOF             uintptr
    JBEInstruction  x86asm.Inst
    JBEAddress      uintptr
    CallAddress     uintptr
    CallInstruction x86asm.Inst
    JMPAddress      uintptr
}
```

Заглянем в тело функции

```
89  func inspectFunction(ptr uintptr) (res sections, err error) {
90      ins, adrr, err := findInstruction(ptr, 16, x86asm.JBE, x86asm.LEA, x86asm.CMP, x86asm.NOP, x86asm.INT)
91      if err != nil {
92          return res, err
93      }
94
95      res.JBEInstruction = *ins
96      res.JBEAddress = adrr
97
98      rel, ok := ins.Args[0].(x86asm.Rel)
99      if !ok {
100         return res, ErrBadJBEArg
101     }
102
103     res.Body = adrr + uintptr(ins.Len)
104     res.Footer = res.Body + uintptr(rel)
105
106     ins, adrr, err = findInstruction(res.Footer, 128, x86asm.CALL, x86asm.MOV, x86asm.CMP, x86asm.NOP, x86asm.INT)
107     if err != nil {
108         return res, err
109     }
110
111     res.CallAddress, res.CallInstruction = adrr, *ins
112     ins, adrr, err = findInstruction(adrr+uintptr(ins.Len), 128, x86asm.JMP, x86asm.MOV, x86asm.CMP, x86asm.NOP, x86asm.INT)
113     if err != nil {
114         return res, err
115     }
116
117     if rel, ok := ins.Args[0].(x86asm.Rel); !ok || adrr+uintptr(rel)+uintptr(ins.Len) != ptr {
118         return res, ErrBackAddressNoMatch
119     }
120
121     res.Header = ptr
122     res.EOF = adrr + uintptr(ins.Len) + 1
123     res.JMPAddress = adrr
124
125     return res, nil
126 }
```

```go
func findInstruction(ptr uintptr, size int, desired x86asm.Op, allowed ...x86asm.Op) (instr *x86asm.I
    code := _asBytes(ptr, size)
    pos := 0
s:
    for {
        if pos > (size - 10) {
            ptr += uintptr(pos)
            code, pos = _asBytes(ptr, size), 0
        }

        ins, err := x86asm.Decode(code[pos:], strconv.IntSize)
        if err != nil {
            return nil, 0, err
        }

        if ins.Op == desired {
            return &ins, ptr + uintptr(pos), nil
        }

        for _, allow := range allowed {
            if ins.Op == allow {
                pos += ins.Len

                continue s
            }
        }

        return nil, 0, fmt.Errorf("%w %v excepted %v", ErrUnexpectedInstructionFound, ins, desired)
    }
}
```

```go
package function

import (
    "encoding/binary"
    "fmt"
    "reflect"
    "strconv"
    "sync/atomic"
    "unsafe"

    "golang.org/x/arch/x86/x86asm"
)
```

Дизассемблер уж есть

Собираем байт-код новой функции

```
183     newFooter := append(moveDXBytes(newFnc), 0xFF, 0x22) // jmp to new implementation
184     oldHeader := copyBytes(sec.Header, len(newHeader))
185     oldFooter := copyBytes(sec.Footer, len(newFooter))
186
187     jmpBytes := append(moveDXBytes(oldFnc), jmpFar(sec.Body)...)
188     callAddr := sec.CallAddress + uintptr(sec.CallInstruction.Len) + uintptr(rel)
189     callBytes := append(moveDXBytes(callAddr), 0xFF, 0xD2) // call morestack_ctx
190     size := len(beforeJBE) + len(beforeCall) + len(callBytes) + len(afterCall) + len(jmpBytes) + 4
191     originTramp := make([]byte, 0, size)
192     originTramp = append(originTramp, beforeJBE...)
193     originTramp = append(originTramp, 0x76, byte(len(jmpBytes))) // JBE SHORT
194     originTramp = append(originTramp, jmpBytes...)               // JMP FAR TO ORIGINAL BODY
195     originTramp = append(originTramp, beforeCall...)
196     originTramp = append(originTramp, callBytes...)
197     originTramp = append(originTramp, afterCall...)
198     originTramp = append(originTramp, 0xEB, byte(-size)) // JMP SHORT
```

```
200     setMemProtect(uintptr(unsafe.Pointer(&originTramp[0])), size, true)
201
202     entrypoint := &originTramp[0]
203     fnc := &entrypoint
204     proxyFnc := *(*T)(unsafe.Pointer(&fnc))
205
206     for _, ot := range oldTo {
207         *ot = proxyFnc
208     }
209
210     doPatch(sec.Header, sec.Footer, sec.EOF, newHeader, newFooter)
211       You, 2 weeks ago • add vendor
212     res := &Patch[T]{
213         CallOld:   proxyFnc,
214         CallNew:   rp,
215         Unpatch:   nil,
216         unpatched: 0,
217     }
218
219     h, f, e := sec.Header, sec.Footer, sec.EOF // copy for nolink to sec
220     res.Unpatch = func() {                     //nolint: mustpanic //never panic in run
221         if old := atomic.SwapInt32(&res.unpatched, 1); old != 0 {
222             panic(fmt.Errorf("function %s %w", New(tg).FullName(), ErrAlreadyPatched))
223         }
224
225         doPatch(h, f, e, oldHeader, oldFooter)
226     }
227
228     return res
229 }
```

```
// Replace replace function to new implementation and
func Replace[T any](tg, rp T, oldTo ...*T) *Patch[T]
```

И скажем компилятору, что это функция!

```go
func Example_logs() {
    log.SetOutput(os.Stdout)
    log.Println("one")

    time.Sleep(time.Second)

    if runtime.GOMAXPROCS(0) > 1 {
        log.Println("two")
        time.Sleep(time.Second)
    }

    log.Println("three")
    // output:
    // 2023/10/03 16:19:48 one
    // 2023/10/03 16:19:48 two
    // 2023/10/03 16:19:48 three
}
```

```
--- FAIL: Example_logs (2.00s)
got:
2023/10/03 16:20:50 one
2023/10/03 16:20:51 two
2023/10/03 16:20:52 three
want:
2023/10/03 16:19:48 one
2023/10/03 16:19:48 two
2023/10/03 16:19:48 three
FAIL
Process 85906 has exited with status 1
```

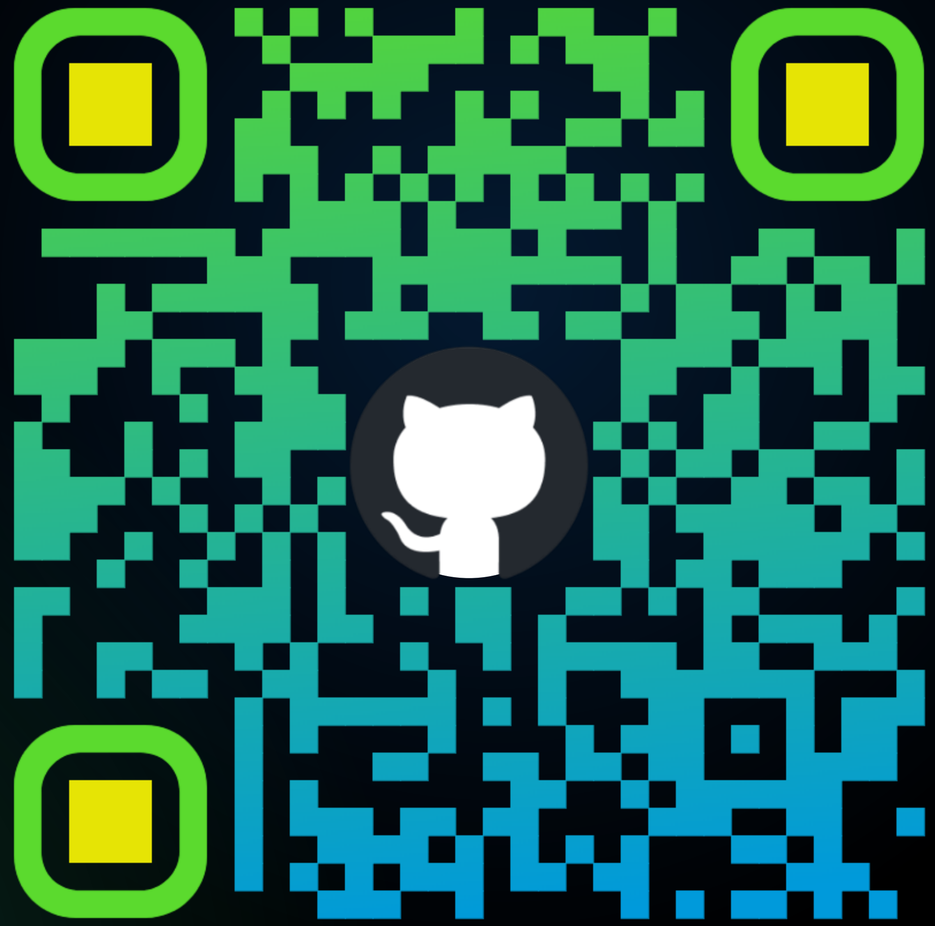
Остановим время!

```go
func Example_logs() {
    log.SetOutput(os.Stdout)
    log.Println("one")

    time.Sleep(time.Second)

    if runtime.GOMAXPROCS(0) > 1 {
        log.Println("two")
        time.Sleep(time.Second)
    }

    log.Println("three")
    // output:
    // 2023/10/03 16:19:48 one
    // 2023/10/03 16:19:48 two
    // 2023/10/03 16:19:48 three
}
func init() {
    function.Replace(time.Now, func() time.Time {
        return time.Date(2023, 10, 03, 16, 19, 48, 0, time.Local)
    })
}
```

```
PASS
Process 88466 has exited with status 0
```

Подведем итоги

- Остановили время в тестах
- Обошли все ограничения ООП
- Изменили работу драйвера БД
  без правки его исходников
- Изменили скомпилированный код
  без исходников
- Выполнили массив байт как машинный код
- Осознали важность средств защиты
  от дизассемблирования и отладки нашего кода



Есть вопросы?

Егор Лазаренков

@elllzr


Спасибо