APP ddb1c9

Ao interromper a execução do programa com "control + |" foi gerado um core e pude verificar algumas chamadas:

net.runtime_pollWait(0x7fe603b90328, 0x72, 0x0)
/usr/lib/go/src/pkg/runtime/netpoll.goc:116 +0x6a
net.(*pollDesc).Wait(0xc2100490d0, 0x72, 0x7fe603b8f0e8, 0xb)
/usr/lib/go/src/pkg/net/fd_poll_runtime.go:81 +0x34
net.(*pollDesc).WaitRead(0xc2100490d0, 0xb, 0x7fe603b8f0e8)
/usr/lib/go/src/pkg/net/fd_poll_runtime.go:86 +0x30
net.(*netFD).accept(0xc210049070, 0x6a5c18, 0x0, 0x7fe603b8f0e8, 0xb)
/usr/lib/go/src/pkg/net/fd_unix.go:382 +0x2c2
net.(*TCPListener).AcceptTCP(0xc2100001c8, 0x18, 0xc210038010, 0x449743)
/usr/lib/go/src/pkg/net/tcpsock_posix.go:233 +0x47
net.(*TCPListener).Accept(0xc2100001c8, 0x7fe603b8f330, 0xc2100369c0, 0x0, 0x0)
/usr/lib/go/src/pkg/net/tcpsock_posix.go:243 +0x27
net/http.(*Server).Serve(0xc21001d410, 0x7fe603b8f388, 0xc2100001c8, 0x0, 0x0)
/usr/lib/go/src/pkg/net/http/server.go:1622 +0x91
net/http.(*Server).ListenAndServe(0xc21001d410, 0xc21001d410, 0x6a3c28)
/usr/lib/go/src/pkg/net/http/server.go:1612 +0xa0
net/http.ListenAndServe(0x6333a0, 0x5, 0x0, 0x0, 0x4183b0, ...)
/usr/lib/go/src/pkg/net/http/**server.go**:1677 +0x6d
main.main()
/home/gallois/Development/go/src/github.com/gallois/http_challenge/http.go:14 +0x65

Então rodei o **NMAP** e verifiquei que a **porta 8011** estava aberta por um serviço desconhecido.

RESULTADO:
Servidor WEB rodando na porta 8011, que imprime apenas "!"

APP cc9621

Utilizando o GDB e o **disassemble main** puder observar:

Existe chamada para **getenv** (aparentemente para pegar o $PATH do usuário que executa)
Chamada para a função **strcat**(para concatenação de strings)
chamada para o **scanf** (leitura de entrada do teclado)
chamadas para **fopen, fputs e fclose** (abertura, gravação e fechamento de arquivo, respectivamente).

RESULTADO:
Aparentemente a aplicação deveria ler o path e concatenar com a entrada do teclado.

mas o que acontece é que a leitura do teclado vai para **/tmp/$USER**

e por fim a uma chamada para **__stack_chk_fail**, para controle de estouro de memória em tempo de execução.

APP d3ea79

Dessa vez foi mais fácil e já fui direto no **/tmp/$USER** e tinha um link para wikipedia, com a mensagem: "Welcome to the Game of Life." e vários padrões desenhados, como se fossem vários frames separados por ----------.
RESULTADO:
Trata-se de um jogo no estilo zero-player que contém sua descrição em:
https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

Via GDB verifiquei que existem apenas 4 funções executadas, são elas: clear, welcome, play e exit.

APP da87fa

RESULTADO:
Trata-se de um **signal_handler**

Ao executá-lo e enviar alguns sinais para término da execução, com o comando kill -n da87fa e para n sendo um inteiro de 1 a 31 no padrão UNIX, o programa termina com uma mensagem que depende do valor de n.

Para n=1, a mensagem de término do programa é: "Desconexão"
Para n=5, "Trace/breakpoint trap (imagem do núcleo gravada)"
Para n=10, "Sinal 1 definido pelo usuário"