ДОКЛАДЫ СЕКЦИИ «ЭЛЕКТРОНИКА»

HTTPS ПРОТОКОЛЫ В GO

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Брынза Д.В.

Сацук С.М. – к.т.н., доцент

В 2015 году компания Google стала помечать web-сайты использующие протокол HTTP протокол для шифрования, как небезопасные. На смену ему пришел HTTPS. В 2009 году компания начала разработку языка Go(Golang), который на данный момент полностью поддерживает новый защищенный протокол шифрования HTTPS.

HTTPS – расширение протокола HTTP для поддержки шифрования в целях повышения информационной безопасности web-сайтов и приложений. Данные в протоколе HTTPS передаются поверх криптографических протоколов SSL или TLS. В отличие от HTTP с TCP-портом 80, для HTTPS по умолчанию используется TCP-порт 443.

С помощью данной функции в Go можно реализовать простой HTTP сервер, который запуститься на 8080 порту:

```
func main() {
   http.HandleFunc("/", handler)
   http.ListenAndServe(":8080", nil)
}
```

Чтобы начать работать по защищенному протоколу HTTPS необходимо вместо
http.ListenAndServe(":8080", nil)
воспользоваться методом
http.ListenAndServeTLS(":8081", "cert.pem", "key.pem", nil), где "cert.pem" - серверный сертификат в PEM формате, "key.pem" – приватный ключ в PEM формате.

Затем необходимо сгенерировать сертификат и приватный ключ ключа с помощью OpenSSL:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout key.pem -out cert.pem
```

На этом настройка HTTPS в Go закончена. Исходный код запуска HTTPS сервера указан в приложении.

```
package main
import (
   "fmt"
   "github.com/kabukky/httpscerts"
   "log"
   "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
   fmt.Fprintf(w, "Привет")
}

func main() {
   // Проверяем, доступен ли cert файл.
   err := httpscerts.Check("cert.pem", "key.pem")
   // Если он недоступен, то генерируем новый.
   if err != nil {
      err = httpscerts.Generate("cert.pem", "key.pem", "127.0.0.1:8081")
      if err != nil {
         log.Fatal("Ошибка: Не можем сгенерировать https сертификат.")
      }
   }
   http.HandleFunc("/", handler)
   http.ListenAndServeTLS(":8081", "cert.pem", "key.pem", nil)
}
```

Таким образом можно сделать вывод, что современный язык программирования Go позволяет всего в пару строк создавать безопасное шифрованное HTTPS подключение к ресурсам сети интернет, в отличии от других языков программирования.

Список использованных источников:

1. The Go Programming Language Brian W. Kernighan [Электронный ресурс]. – Режим доступа : https://golang.org/pkg/net/http/ – Дата доступа: 18.04.2018.