



# ШИФРОВАНИЕ ДАННЫХ В URL



# Тема: Защита данных в приложениях от несанкционированного доступа

**Цель работы:** познакомиться и изучить методы защиты данных в веб-приложениях при передаче по незащищенному http-соединению

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Теоретические сведения

Одной из задач разработчиков и любого электронного ресурса – защита конфиденциальной информации от хищения и уничтожения.

Практически любая современная веб-система аутентифицирует (опознает) пользователя через ввод логина и пароля. Эти данные в открытом виде передаются через GET или POST запрос на сервер для проверки. Перехватив трафик между браузером и сервером, злоумышленник легко получает эти логин и пароль.

Одним из вариантов решения проблемы является использование протокола HTTPS с шифрованием трафика между браузером и сервером. Однако данный вариант защиты не всегда доступен.

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

Одной из задач разработчиков и любого электронного ресурса – защита конфиденциальной информации от хищения и уничтожения.

Практически любая современная веб-система аутентифицирует (опознает) пользователя через ввод логина и пароля. Эти данные в открытом виде передаются через GET или POST запрос на сервер для проверки. Перехватив трафик между браузером и сервером, злоумышленник легко получает эти логин и пароль.

Одним из вариантов решения проблемы является использование протокола HTTPS с шифрованием трафика между браузером и сервером. Однако данный вариант защиты не всегда доступен.

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

Шифрование параметров URL во многих случаях является доступным и эффективным защитой данных способом при передаче по незащищенному соединению.

Существует два подхода при шифрование данных: двустороннее и одностороннее шифрование. Последнее иногда называется хешированием.

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Двустороннее шифрование

Шифрование данных - это методы защиты любой информации от несанкционированного доступа, просмотра, а также её использования, основанные на преобразовании данных в зашифрованный формат и состоящие из двух взаимобратных процессов: зашифровывания и расшифровывания.

Расшифровать, восстановить зашифрованную информацию или сообщение, обычно можно только при помощи ключа, который применялся при его зашифровывании

Все методы шифрования делятся на два основных класса: шифрование с симметричным ключом и шифрование с асимметричным ключом.

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Шифрование с симметричным ключом

**Ключ шифрования** - это случайная или специальным образом созданная по паролю последовательность бит, являющаяся переменным параметром алгоритма шифрования. Если зашифровать одни и те же данные одним алгоритмом, но разными ключами, результаты получатся тоже разные. При шифровании с симметричным ключом. Алгоритм и ключ выбирается заранее и известен обеим сторонам. Сохранение ключа в секретности является важной задачей для установления и поддержки защищённого канала связи.



Если в симметричных схемах злоумышленник перехватит ключ, то он сможет как «слушать», так и вносить правки в передаваемую информацию

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Шифрование с симметричным ключом

К алгоритмам с симметричным ключом относятся :

- **AES (Rijndael)**. В настоящее время является федеральным стандартом шифрования США
- **ГОСТ 28147-8**. Стандарт Российской Федерации на шифрование и имитозащиту данных.
- **DES** Федеральный стандарт шифрования США в 1977-2001 годах.
- **CAST** В некотором смысле аналог DES.





# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Шифрование с асимметричным ключом

В системах с открытым ключом используются два ключа — открытый и закрытый. Открытый ключ передаётся по открытому (то есть незащищённому, доступному для наблюдения) каналу и используется для шифрования сообщения. Для расшифровки сообщения используется секретный ключ. В асимметричных системах другой стороне передается открытый ключ, который позволяет шифровать, но не расшифровывать информацию. Данная схема решает проблему симметричных схем



# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Шифрование с асимметричным ключом

- **RSA.** Разработан в 1977 году в Массачусетском технологическом институте (США).
- **ElGamal.** Разработан в 1985 году. Назван по фамилии автора - Эль-Гамаль. Используется в стандарте США на цифровую подпись DSS (Digital Signature Standard).



# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Хеширование данных

Хеширование – это одностороннее шифрование. **Хеш-функция** — функция, осуществляющая преобразование массива входных данных произвольной длины в (выходную) битовую строку установленной длины, выполняемое определённым алгоритмом. Преобразование, производимое хеш-функцией, называется **хешированием**. Исходные данные называются входным массивом, «**ключом**» или «**сообщением**». Результат преобразования (выходные данные) называется «**хеш-кодом**», «**хеш-суммой**». К алгоритмам хеширования относятся : **CRC16/32** , **MD2/4/5/6**, **MD5**, **MD6**, алгоритмы линейки **SHA** , **ГОСТ 34.11-94**.



# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Хеширование данных

Хеширование – это одностороннее шифрование. **Хеш-функция** — функция, осуществляющая преобразование массива входных данных произвольной длины в (выходную) битовую строку установленной длины, выполняемое определённым алгоритмом. Преобразование, производимое хеш-функцией, называется **хешированием**. Исходные данные называются входным массивом, «**ключом**» или «**сообщением**». Результат преобразования (выходные данные) называется «**хеш-кодом**», «**хеш-суммой**». К алгоритмам хеширования относятся : **CRC16/32** , **MD2/4/5/6**, **MD5**, **MD6**, алгоритмы линейки **SHA** , **ГОСТ 34.11-94**.



# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Обфускация данных

Маскировка (обфускация) данных — это способ защиты конфиденциальной информации от несанкционированного доступа путём замены исходных данных фиктивными данными или произвольными символами. При этом замаскированная информация выглядит реалистично и непротиворечиво и может использоваться в процессе тестирования программного обеспечения. В большинстве случаев маскировка применяется для защиты персональных данных и конфиденциальных сведений организации.

```
Profile.cfm?userId=911&name=Bob&departmentId=5
```

```
Profile.cfm?vc5c71dd0fbb58b1de4df=911&vfe6f54c33f9064833ee8=Bob&va929c5b94ad8ed832a38=5
```

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Пример

Захешировать параметры, а данные зашифровать в GET запросе и осуществить передачу по протоколу HTTP.

```
<cfset TheDate = "21/10/19">
<cfset TheString = "I'm working in M&M">
<cfset ThePhoneNumber = "+375(29)12-34-56-1">
</cfscript>

SecretKey = GenerateSecretKey("DES")
SendDate = encrypt(TheDate, SecretKey, "DES", "Hex")
SendText = encrypt(TheString, SecretKey, "DES", "Hex")
SendNumber = encrypt(ThePhoneNumber, SecretKey, "DES", "Hex")

</cfscript>
<cfoutput>
  #TheDate# &nbsp;&nbsp;&nbsp;#SendDate# &nbsp;&nbsp;&nbsp;#SecretKey#<br>
  #TheString# &nbsp;&nbsp;&nbsp;#SendText# &nbsp;&nbsp;&nbsp;#SecretKey#<br>
  #ThePhoneNumber# &nbsp;&nbsp;&nbsp;#SendNumber# &nbsp;&nbsp;&nbsp;#SecretKey#<br>
</cfoutput>
<cfoutput>
  <a href="http://127.0.0.1:8500/CryptoHash/DeCryptoHashURL.cfm?v#Hash('GetDate', 'SHA-384', 'UTF-8',
500)##SendDate#&v#Hash('GetText', 'SHA-384', 'UTF-8', 500)##SendText#&v#Hash('GetNumber', 'SHA-384', 'UTF-8',
500)##SendNumber#&v#Hash('ScKey', 'SHA-384', 'UTF-8', 500)##SecretKey#">Передать текст</a>
</cfoutput>
```

CryptoHesh.cfm

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Пример

Захешировать параметры, а данные зашифровать в GET запросе и осуществить передачу по протоколу HTTP.

### DeCryptoHesh.cfm

```
<cfset AccessDate      = decrypt(#url["v"&#Hash('GetDate', 'SHA-384', 'UTF-8', 500)#]#, #url["v"&#Hash('ScKey', 'SHA-384', 'UTF-8', 500)#] #, "DES", "Hex")>
<cfset AccessText      = decrypt(#url["v"&#Hash('GetText', 'SHA-384', 'UTF-8', 500)#]#, #url["v"&#Hash('ScKey', 'SHA-384', 'UTF-8', 500)#] #, "DES", "Hex")>
<cfset AccessNumber    = decrypt(#url["v"&#Hash('GetNumber', 'SHA-384', 'UTF-8', 500)#]#, #url["v"&#Hash('ScKey', 'SHA-384', 'UTF-8', 500)#] #, "DES", "Hex")>
<cfoutput>
    #AccessDate#<br>
    #AccessText#<br>
    #AccessNumber#<br>
</cfoutput>
```

# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Ход работы

1. Передать шифрованные данные в GET-запросе, захешировав при этом параметры URL.

Имя:	Kelly Smith
Должность:	Vice President
Отдел:	M&M
Телефон:	8(0297)11-23-45
Дата:	08/15/2000
Email:	Kelly@example.com
Комментарий:	I get up at 7 o'clock. I wash my hands, my face and clean my teeth. I air my room and make the bed. Then I have my breakfast.



# Тема: Защита данных в веб-приложениях от несанкционированного доступа

## Ход работы

2. Разработать функционал, осуществляющий:

- *регистрацию и аутентификацию пользователей*
- *проверку уникальности учетной записи (логина) при регистрации*
- *хранение пароля в базе данных в зашифрованной форме*
- *передачу данных и параметров в HTTP-запросах осуществлять в зашифрованной форме*