Kriptografija za CTF natjecanja

izv. prof. dr. sc. Ante Đerek

Ciljevi za danas

- Kratko ponoviti dijelove kriptografije koji mogu biti korisni u CTF natjecanjima.
- Pokriti što više različitih tema kako bi mogli znati gdje sami početi i dalje istraživati.
- Zajedno riješiti par zadataka.
- Nisu nam ciljevi:
  - Iscrpan pregled kriptografije.
  - Iscrpan pregled svega što se može pojaviti na CTF-ovima, a ima veze s kriptografijom.
  - Ulaženje u detalje rada pojedinih kriptografskih algoritama ili pojedinih napada.
  - …

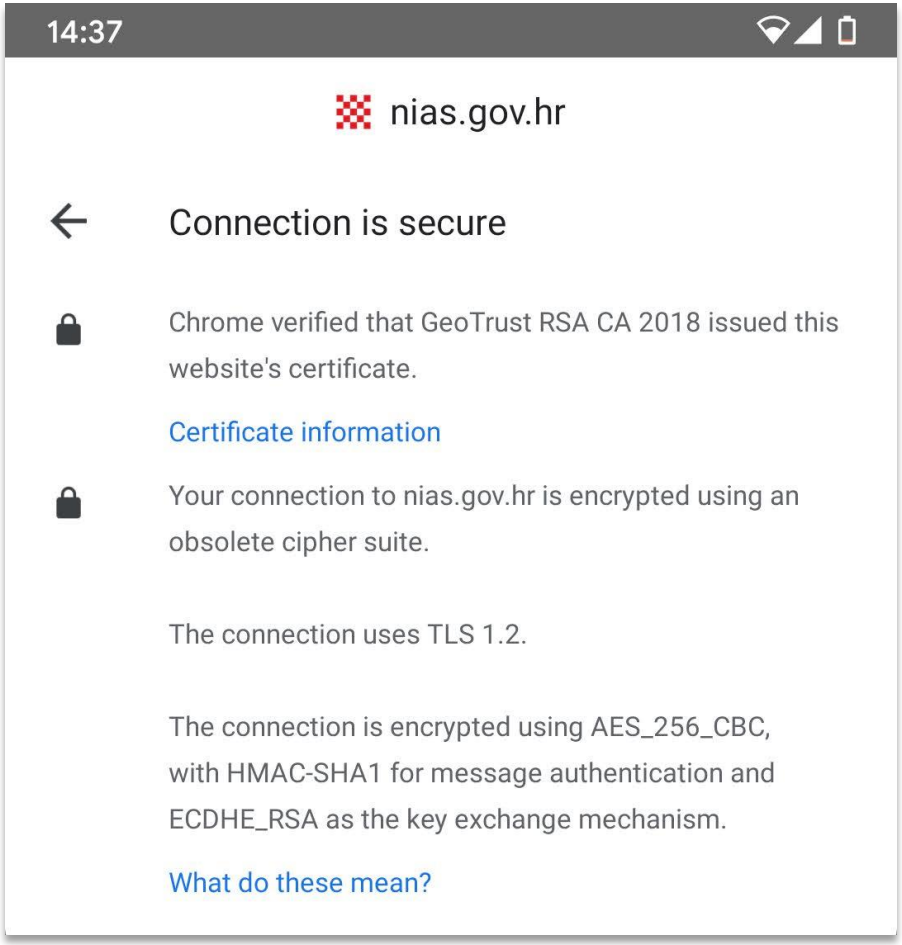
Neka korisna literatura

- FER – SRS video predavanja
  - https://www.youtube.com/playlist?list=PLRFTXn5ZdqaPTikB06X61gyijBWbF9jdG
- Udžbenici
  - Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone (1996.), *Handbook of Applied Cryptography*, CRC Press
  - A Graduate Course in Applied Cryptography, Dan Boneh and Victor Shoup, https://toc.cryptobook.us/
- Kriptografija u CTF natjecanjima
  - https://ctf101.org/cryptography/overview/
  - https://ctf-wiki.mahaloz.re/crypto/introduction/
- Vježbanje
  - https://ctftime.org/

Problem: *Sigurna* komunikacija putem nesigurnog kanala

- *Povjerljivost*: može li napadač saznati sadržaj komunikacija?
- *Integritet*: može li napadač promijeniti sadržaj komunikacije?
- *Autentifikacija*: može li Ana biti sigurna da komunicira baš s Brankom i obrnuto?
- …

Osnovni pojmovi moderne kriptografije



- Simetrične šifre
- Kriptografske *funkcije sažetka*
- Kodovi za integritet poruke

- Asimetrične šifre
- Digitalni potpisi
- Diffie-Hellmanova razmjena ključeva

Kategorija 1: Zadaci koji zapravo nemaju veze s kriptografijom.

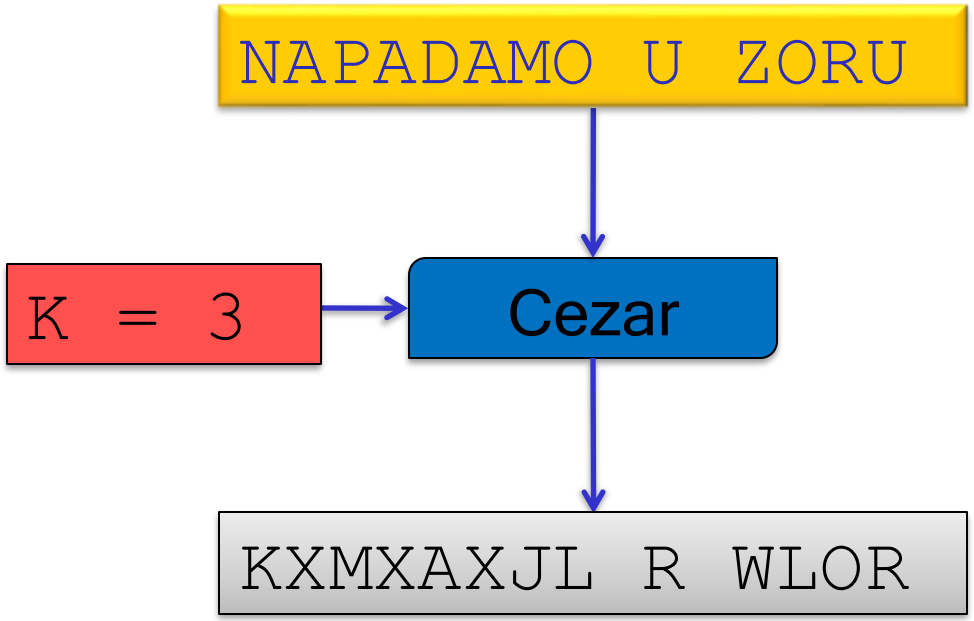
- Tipičan zadatak: zadan je neki tekst ili datoteka koja izgleda nečitljivo.
  - Primjer: https://github.com/infobip/infobip-ctf-2021/tree/master/crypto/mic_check
- Pristupi:
  - Pogodi ili otkrij način na koji su podaci kodirani.
  - Dekodiraj podatke.
- Korisni alati:
  - Google
  - ‘file’ alat na Linux-u
  - https://gchq.github.io/CyberChef/

Osnove kriptografije i kriptoanalize

Klasična kriptografija

Cezarova šifra s *ključem*

- Svako slovo u *izvornom tekstu* zamjeni sa slovom koje dolazi K pozicija ispred



Gruba sila (osnovni algoritam kriptoanalize)

- Isprobaj sve moguće ključeve, dešifriraj poruku i pogledaj ima li rezultat smisla

QRI Z KZ JZEV SILKV	
RSJ A LA KAFW TJMLW	K = 1
STK B MB LBGX UKNMX	K = 2
TUL C NC MCHY VLONY	K = 3
UVM D OD NDIZ WMPOZ	K = 4
VWN E PE OEJA XNQPA	K = 5
WXO F QF PFKB YORQB	K = 6
XYP G RG QGLC ZPSRC	K = 7
YZQ H SH RHMD AQTSD	K = 8
ZAR I TI SINE BRUTE	K = 9

Malo mogućih različitih ključeva znači nesiguran sustav!

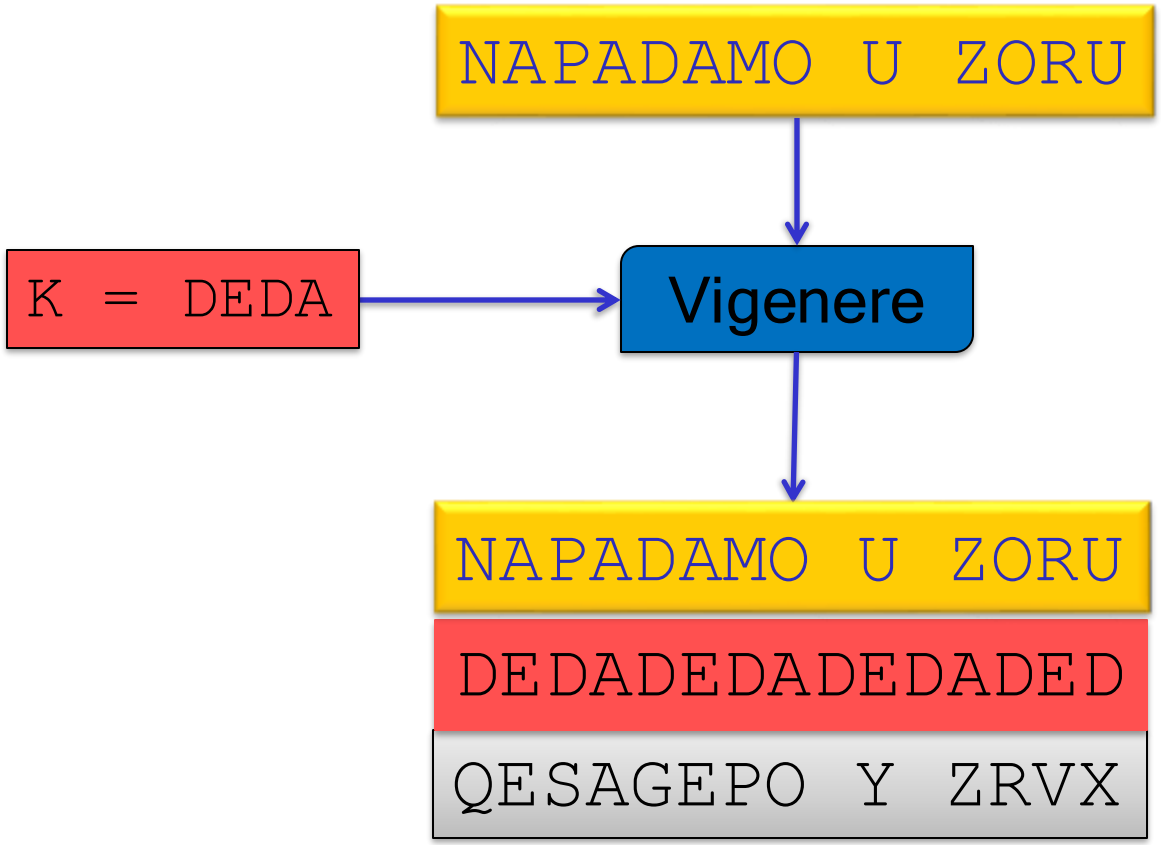
Kategorija 2: Brute-force napadi na kriptosustave.

- Tipičan zadatak: zadan je skriveni tekst i kompliciran algoritam enkripcije koji koristi male ključeve.
  - Primjer: TBTL 2022 Super Encryption (još jedan trik je potreban)
- Pristupi:
  - Isprobaj sve moguće ključeve.
- Korisni alati:
  - Programski jezik po vlastitom izboru ☺
  - Hashcat
  - John the Ripper
  - https://crackstation.net/

Kategorija 3: Povijesna kriptografija

- Tipičan zadatak: zadan je samo tekst koji se sastoji od velikih slova.
  - Primjer: ??
- Pristupi:
  - Pogodi ili zaključi o kojem historijskom kriptosustavu se radi.
  - Dekriptiraj koristeći poznati napad.
- Korisni alati:
  - https://gchq.github.io/CyberChef/
  - https://www.dcode.fr/en

Vigenèreova šifra – *le chiffre indéchiffrable*



Kriptoanaliza Vigenèreove šifre

NIROJBJDLALUKZEUANH**RBZ**NBAUIRZUEEMZELOIOCFN
NYIUISKHOOKUSLIHJRVSSBEOI**QGZ**WOINRWASYKFKQU
ZOARZAWUDRELTQUTFHMOKFRZIUOAQYTWASKIOFNXMB
CHPSLEHQONUMOKBCHPSLESKHOOKUGJJDFATNNBAOUM
GRFZTRTBTHSAJSSXAAIUGNKARZVBRZZIOOUGZOAMPA
LRNFMFDIANBRNJNPPQOZOASGITT**QGZ**JVZTJBRZFVJJ
ZZIHORVOEAQYTWOHPAWNHYELTNXKSOYONPVZIIKESK
DPPQONPSHZIVKTVNPM**QGZ**WOIADSURZVBBHZI**VSS**GNP
VZBITOJOHBKZJENSJOHWRHPEENNYTJIDZIDKHNKSIS
KRJJZSJFSSUKSISOCLOFXAAMHYLKAMPAJPQUPJTHBA
OJZZEKECTALORZITVHNNKEMOHDLZTOWAHHIUIOUKSE
SGCLARTAHAGXVBTRQOHDQASUVZAITPTTJFNIAMJSHP
EGAJALUESGOTLZTJBMNYEOAMGSFTDSEMJMKVSIIKDO
ORZILOIKDBLIK**RBZ**UOJBMNBOEEBGSNOMGCJOMGLOAU
OSPKNYKPLRQAJIRZ**RBZ**HBADKZASUAMUVBSHF**VSS**MOM
OARZAWNHIINAHYTVDDTTJMZI**VSS**SUPPVDFAOARMOTP
NJASSSBONIYBRTNNURHAMOZJRZTAJMDJJVNZXOENNV
RFPNFNBTKPIWA

- Traženje vjerojatnih duljina ključa *Kasiskijevim testom*
- Frekvencijska analiza fragmenata koji odgovaraju istom znaku ključa

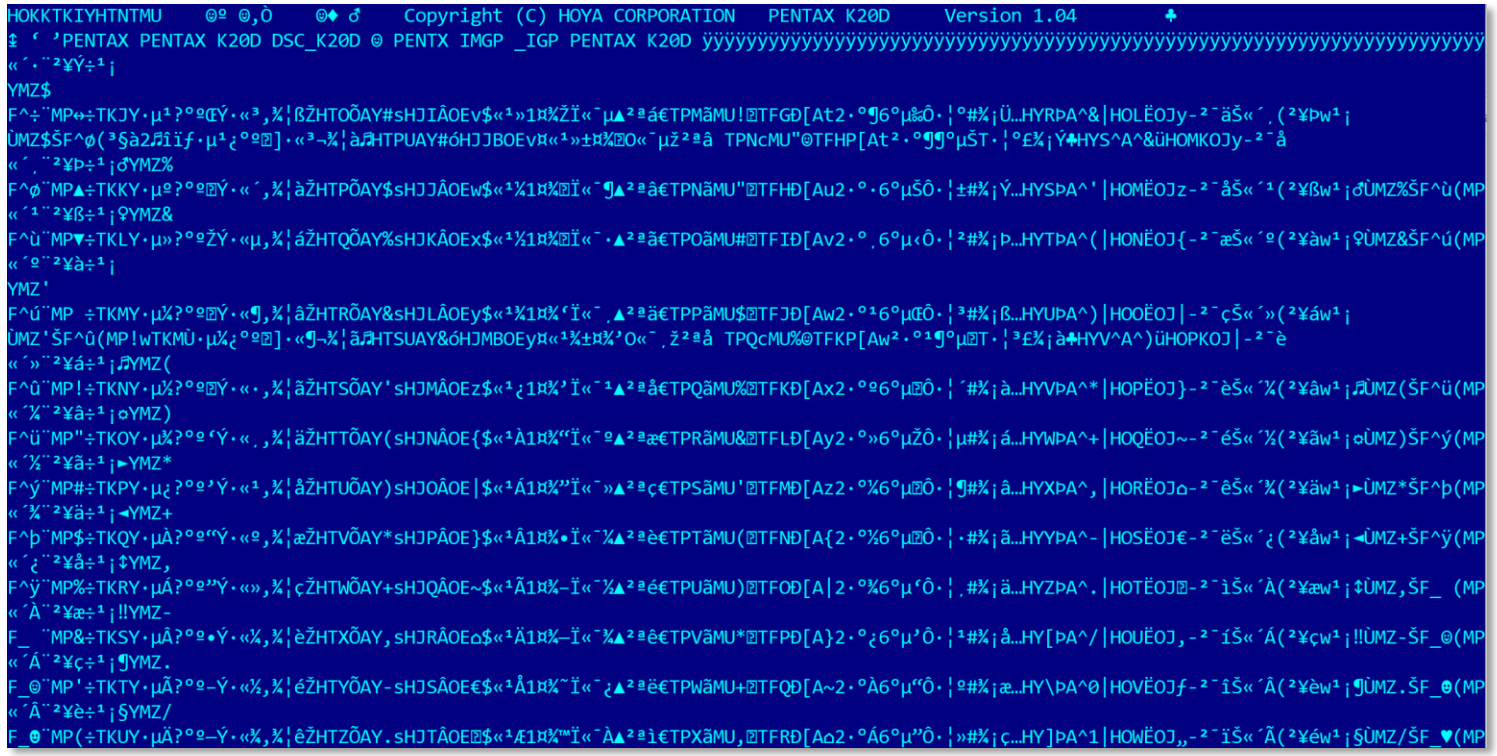
Vigenèreova šifra – napad poznatim tekstom

```
RIYH: RXOE UOMEW <VNFV.YEDVU@FQI.CR>
DJ: JDEEBRX GDFC <EKTZPME.BRAJ@AED.RM>
JEWJQTD: XRLJSU

P BISGOSL KRUAOYLAX UAPRDFA QK PDMS LMSYN.
```

```
RIYH: RXOE UOMEW ...
FROM: ANTE DEREK ...
MRKV??RKVA?RKVAM
```

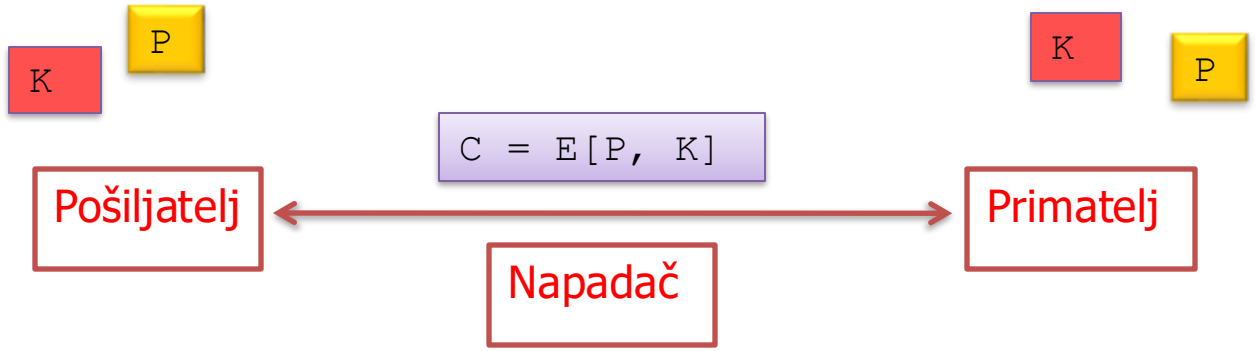
Vigenèreova šifra – primjer iz stvarnog svijeta



Osnove kriptografije i kriptoanalize

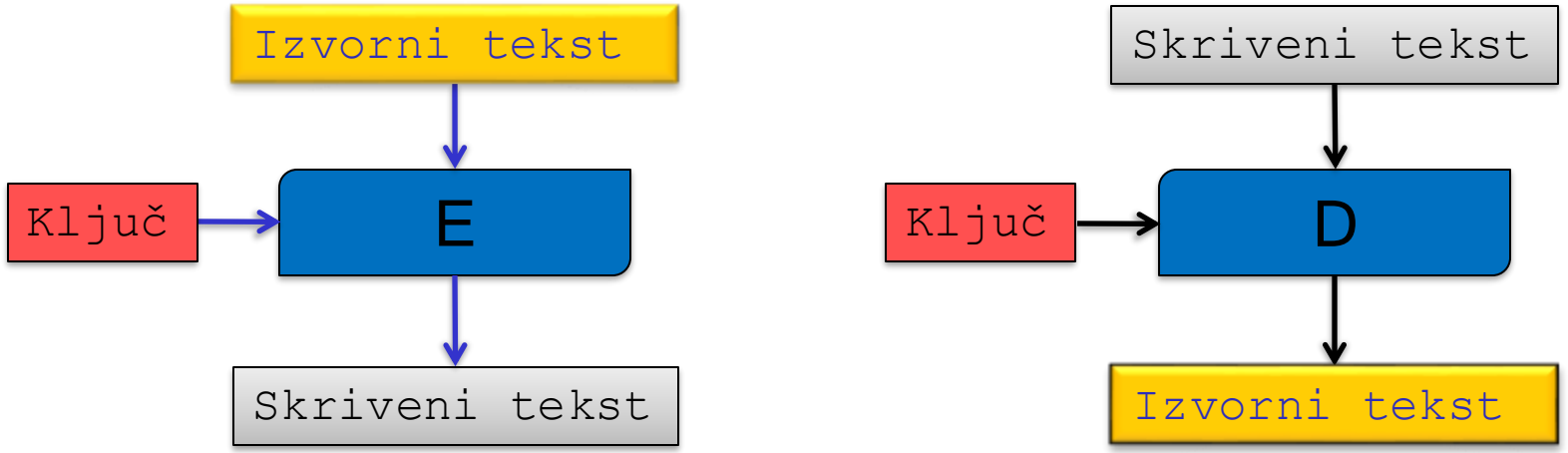
Simetrične šifre

Kako osigurati povjerljivost komunikacije?



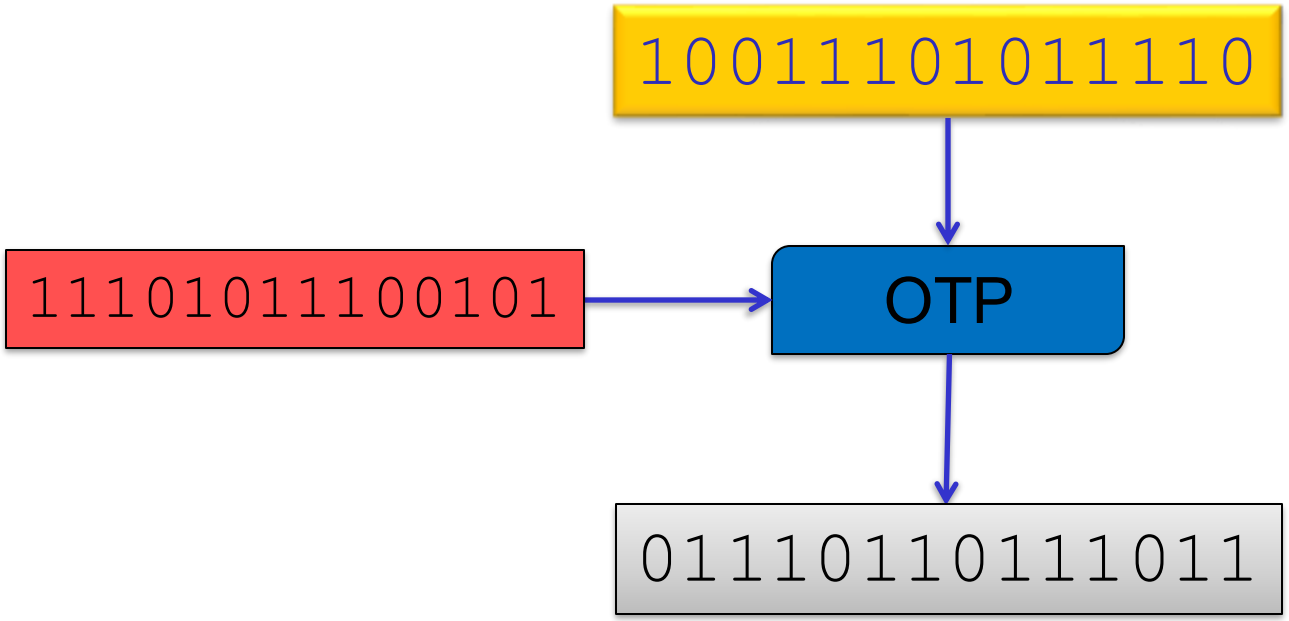
Simetrična šifra

- Poruka ili izvorni tekst ili otvoreni tekst (*plaintext*)
- Šifrat ili skriveni test (*ciphertext*)

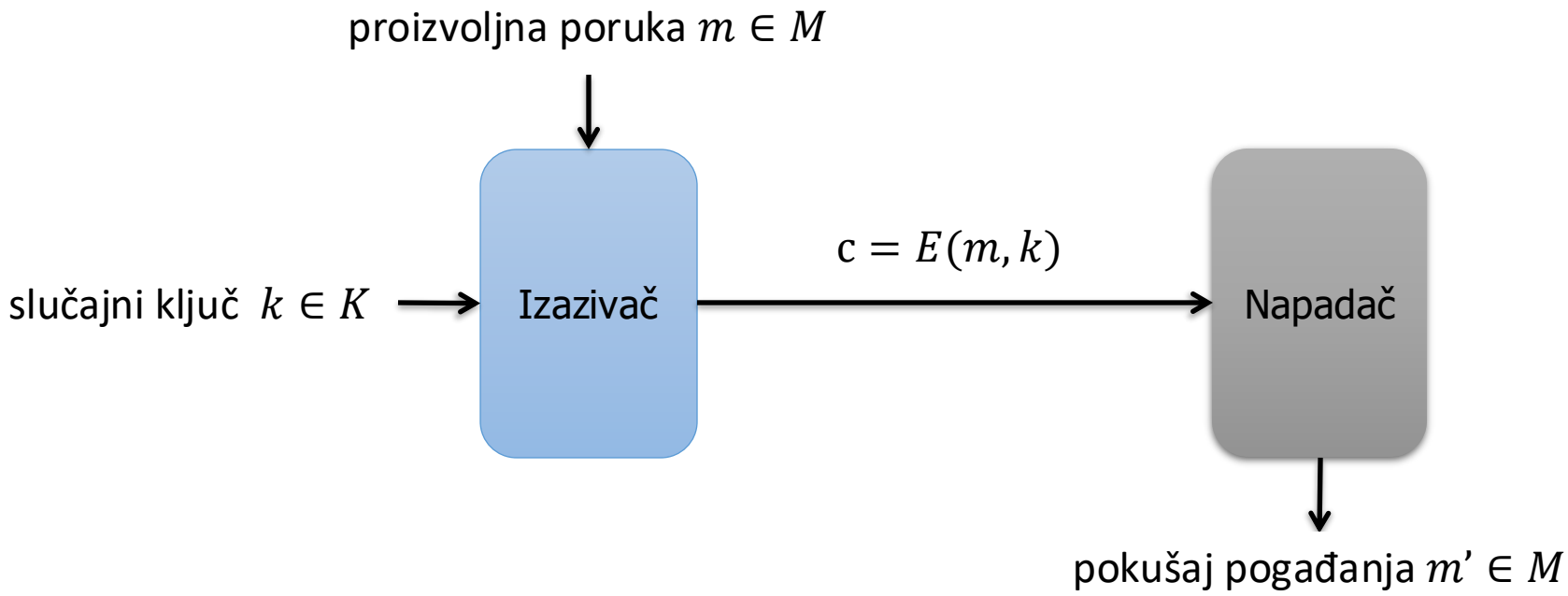


Jednokratna bilježnica (*one-time pad*)

- $M = K = C = \{0,1\}^n$
- $E(m,k) = m \oplus k$
- $D(c,k) = c \oplus k$



Savršena povjerljivost (*perfect secrecy)*, Claude Shannon (1946)



Šifra pruža *savršenu povjerljivost* ako je za **svakog napadača** šansa da pogodi poruku jednaka $1/|M|$ (bez obzira na algoritam napadača, vrijeme izvršavanja, računalne resurse, itd.).

Jednokratna bilježnica – nedostatci

- Ključ mora biti jednako velik kao i poruka!
- Ključ se smije koristiti najviše jednom!
  - $c_1 = m_1 \oplus k$
  - $c_2 = m_2 \oplus k$
  - $c_1 \oplus c_2 = m_1 \oplus m_2$

Jednokratna bilježnica – nedostatci

- Ne štiti integritet poruke (kao niti jedna šifra sama po sebi)!
- Moguće je na predvidiv način izmijeniti poruku (*malleable encryption)*!

$$c_1 = OTP(m_1, k) = m_1 \oplus k$$

$$c_2 = c_1 \oplus m_1 \oplus m_2 = m_1 \oplus k \oplus m_1 \oplus m_2 = m_2 \oplus k = OTP(m_2, k)$$

Kategorija 4: Neispravna upotreba inače razumnog kriptografskog algoritma

- Tipičan zadatak: zadan je skriveni tekst i postupak enkripcije koji, čini se, koristi moderne kriptografske algoritme, ali neki detalj se radi na neispravan ili nesiguran način.
  - Primjer: https://github.com/infobip/infobip-ctf-2021/tree/master/crypto/otp_saas
- Pristupi:
  - Pronađi dio koji nije dobar.
  - Pronađi napad (Google, čitanje udžbenika, čitanje znanstvenih članaka, čitanje writeup-ova).
  - Pronađi ili implementiraj alat koji napada kriptosustav.
- Korisni alati:
  - Google
  - …

Fleksibilnije definicije sigurnosti

Što je cilj napada?

- odrediti tajni ključ k
- odrediti poruku m
- odrediti neki dio poruke m
- odrediti bilo kakvu informaciju o poruci m
- ...

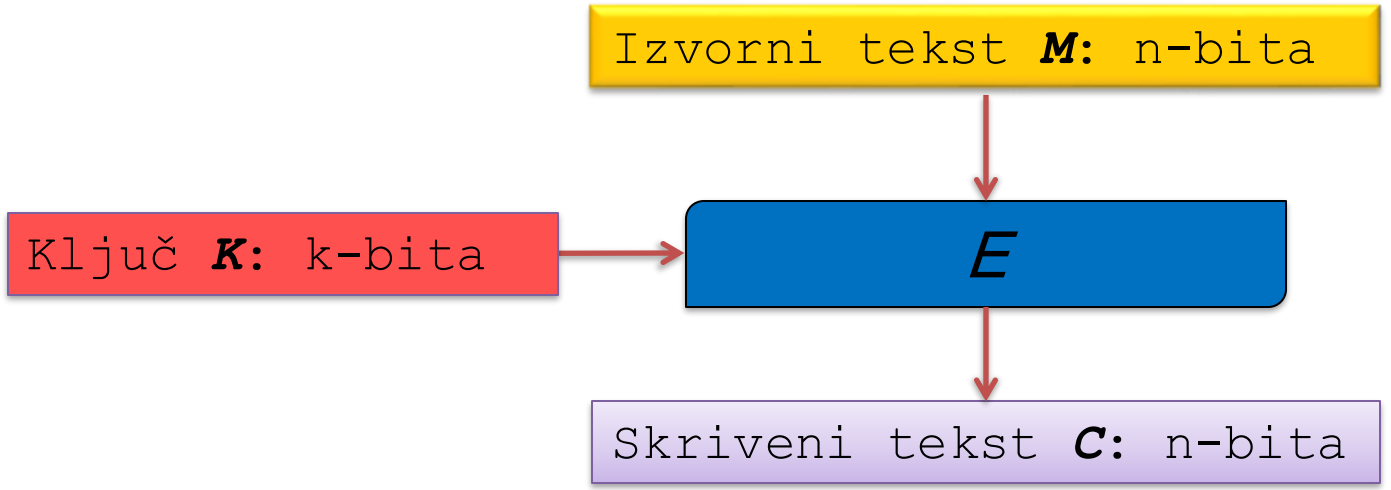
Fleksibilnije definicije sigurnosti

Što napadač ima na raspolaganju?

- samo jedan skriveni tekst
- puno parova (m_i, c_i) gdje je $c_i = E(m_i, k)$
  - Napad poznatim izvornim tekstom / *known plaintext attack*
- mogućnost da dobije $c_i = E(m_i, k)$ za m_i po izboru
  - Napad odabranim izvornim tekstom / *chosen plaintext attack*
- mogućnost da dobije $m_i = D(c_i, k)$ za c_i po izboru
  - Napad odabranim skrivenim tekstom / *chosen ciphertext attack*
- ...

Blok šifra (*block cipher*)

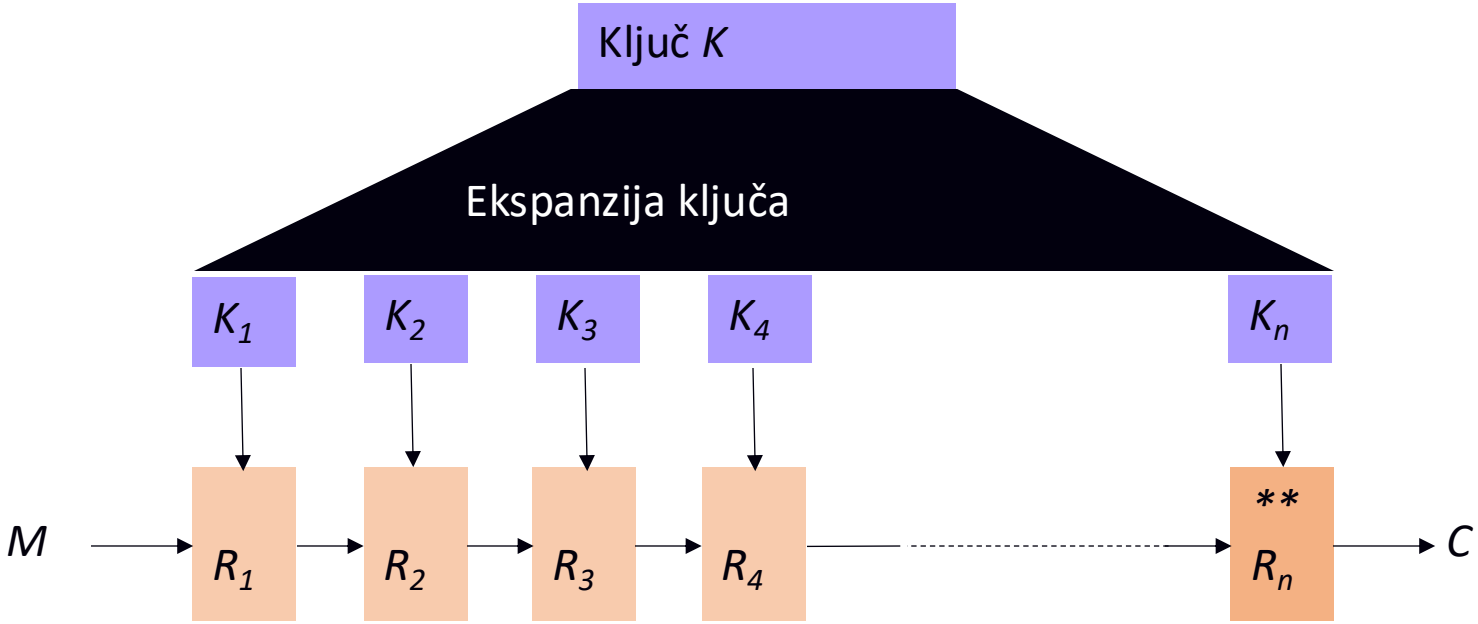
- $M = C = \{0,1\}^n$
- $K = \{0,1\}^k$
- E i D deterministički algoritmi



Primjeri blok šifri

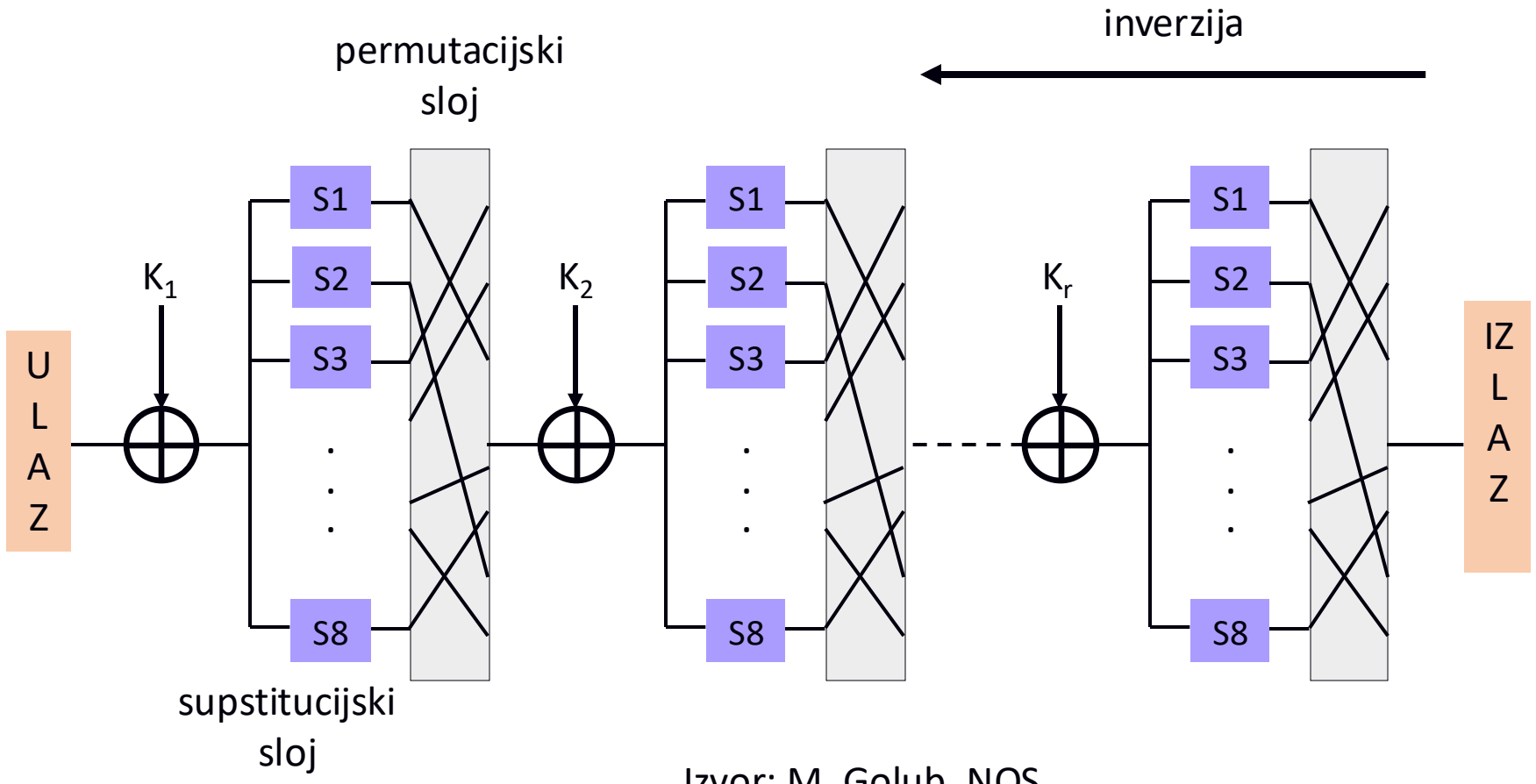
- DES (1970-te)
  - n=64 k=56, dugogodišnji standard, danas potpuno nesiguran zbog malog ključa
- 3DES (1970-te)
  - n=64 k=168, trostruki DES, veća sigurnost s istom šifrom
- IDEA (1991)
  - n=64, k=128
- Blowfish (1993)
  - n=64, k=32–448
- AES (1999)
  - n=128 k=128, 192, 256, standard od 2002., vrlo široko korišten

AES – runde



Izvor: M. Golub, NOS

AES – supstitucijsko-permutacijska mreža

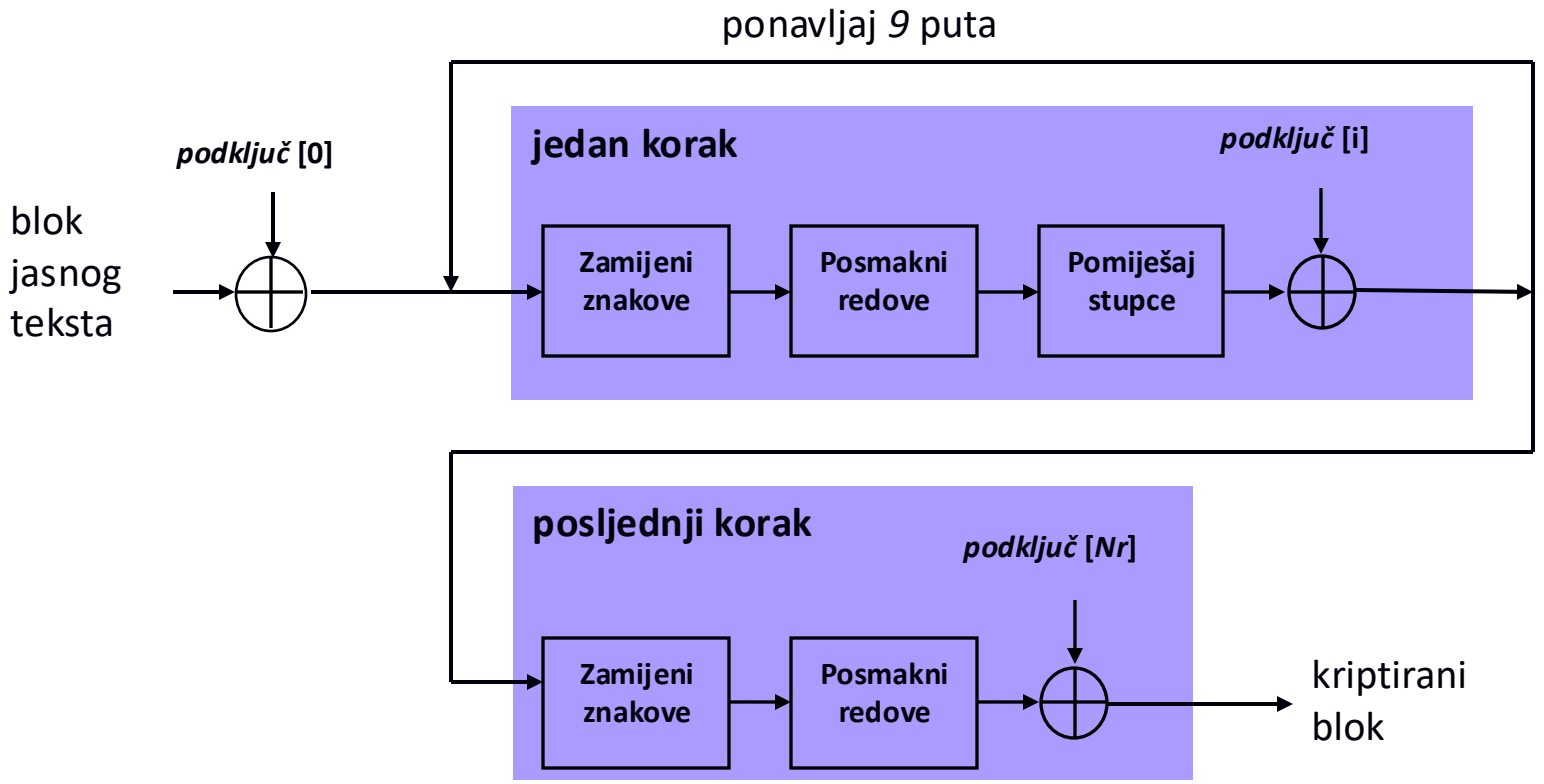


Izvor: M. Golub, NOS

AES128 – blok

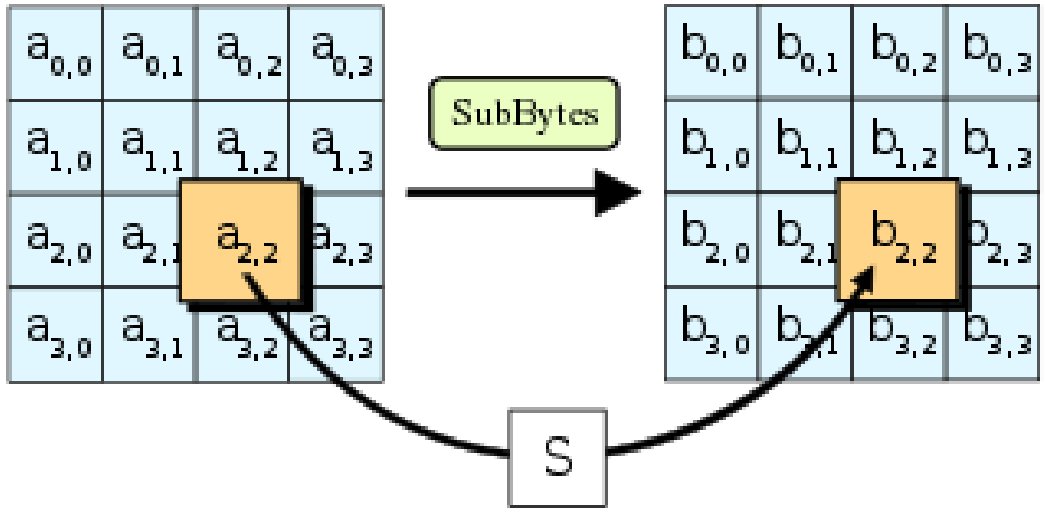
a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

AES128 – postupak (de)šifriranja



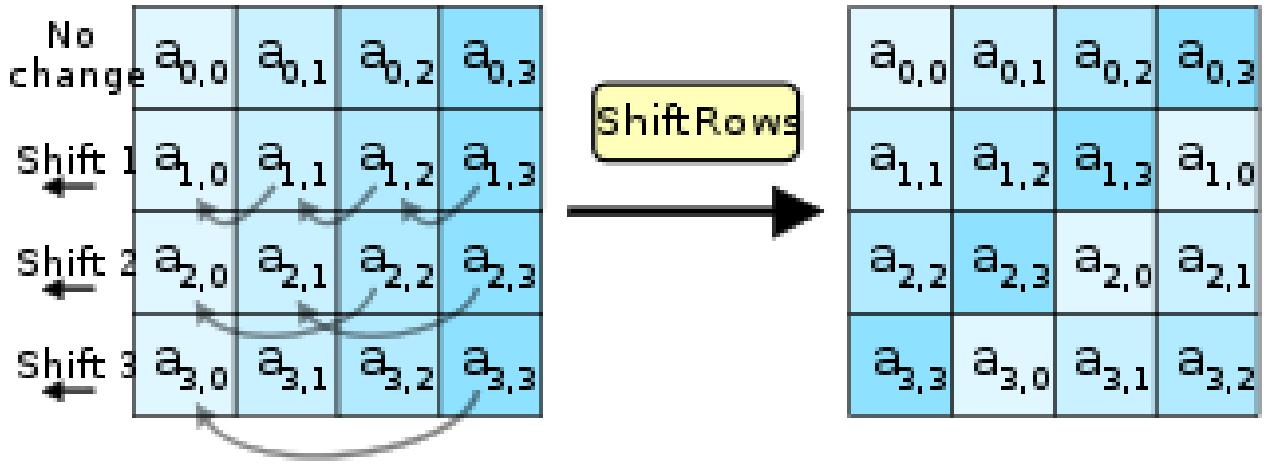
Izvor: M. Golub, NOS

AES128 – Zamijeni znakove



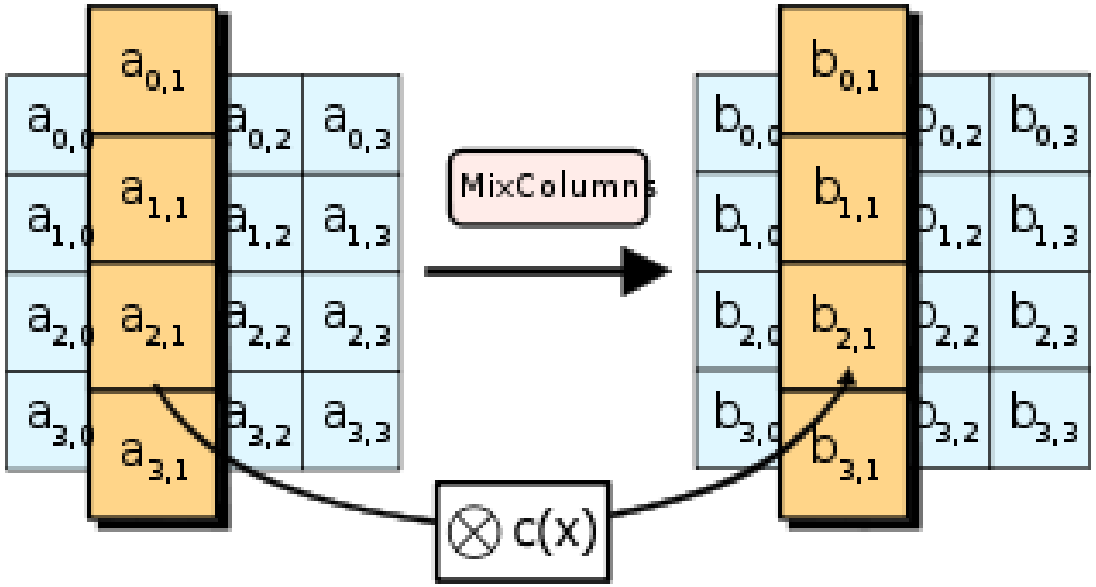
Izvor: wikipedia.org

AES128 – Posmakni redove



Izvor: wikipedia.org

AES128 – Pomiješaj stupce



Izvor: wikipedia.org

AES128 – Pomiješaj stupce

$$\begin{bmatrix} s_{0i}' \\ s_{1i}' \\ s_{2i}' \\ s_{3i}' \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0i} \\ s_{1i} \\ s_{2i} \\ s_{3i} \end{bmatrix}$$

- „zbrajanje” i „množenje” se vrše u polju $GF(2^8)$
- „zbrajanje” i „množenje” u $GF(2^8)$ imaju svojstva potrebna za invertibilnost matričnog množenja

Konačno polje GF(2^8)

- Elementi polja su polinomi oblika:

$$a_7x^7 + a_6x^6 + \cdots + a_1x + a_0, a_i \in \{0, 1\}$$

- Svaki bajt $(a_7a_6a_5a_4a_3a_2a_1a_0)_2$ je predstavljen odgovarajućim polinomom.
- Aritmetičke operacije:
  - zbrajanje: XOR
  - Množenje: binarno množenje polinoma modulo fiksni ireducibilni polinom $g(x) = x^8 + x^4 + x^3 + x + 1$, nekoliko *shift* i XOR operacija

Nije tajni sastojak za sigurnost već za jednostavnost i efikasnost *(citation needed)!*

Zašto ovakav dizajn?

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

The linear mixing layer:	guarantees high diffusion over multiple rounds.
The non-linear layer:	parallel application of S-boxes that have optimum worst-case nonlinearity properties.
The key addition layer:	A simple EXOR of the Round Key to the intermediate State.

Izvor: AES Proposal: Rijndael
Joan Daemen, Vincent Rijmen, 2003.

Domaća zadaća

- Razmatraj AES bez jedne od funkcija.

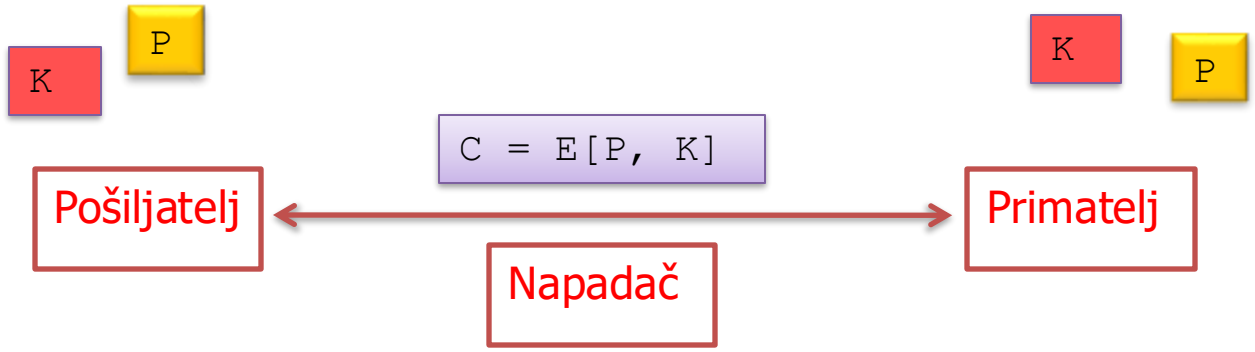
Kategorija 5: Traži se poznavanje pojmova i implementacija

- Tipičan zadatak: zadan je i skriveni tekst i ključ i ime algoritma kojim je tekst šifriran.
  - Primjer: ???
- Pristupi:
  - Pronađi te prilagodi, ili samo upogoni gotovu implementaciju.
- Korisni alati:
  - Google
  - Github

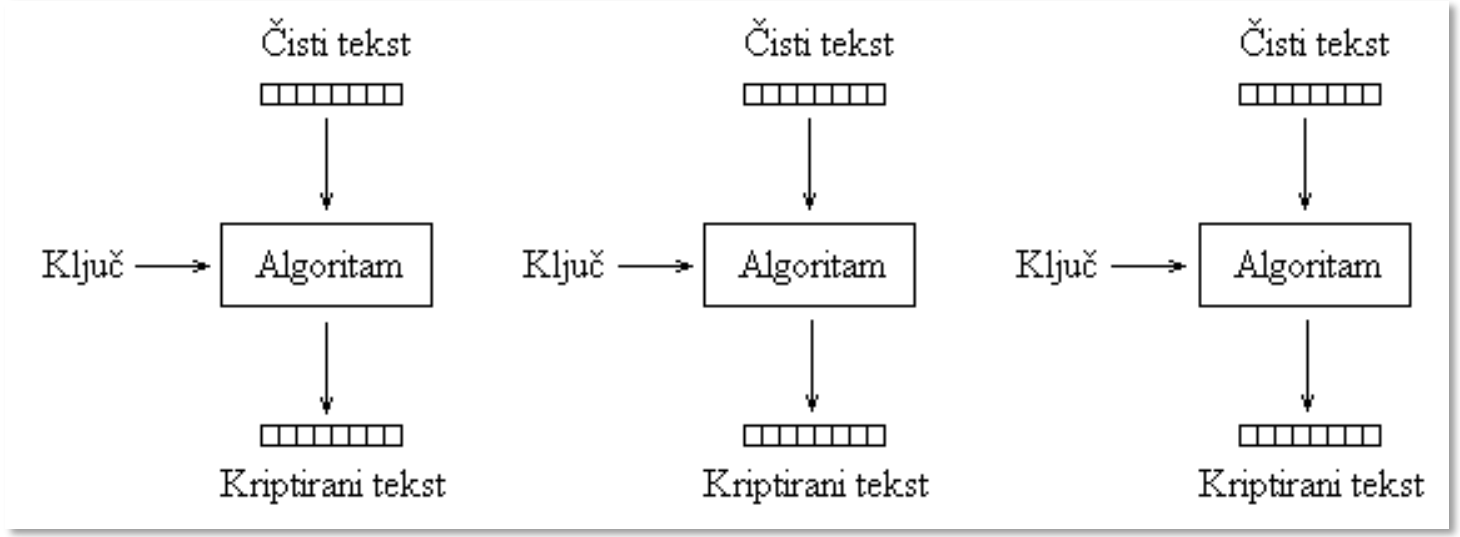
Osnove kriptografije i kriptoanalize

Načini šifriranja
Protočne šifre

Kako šifrirati poruku proizvoljne duljine?



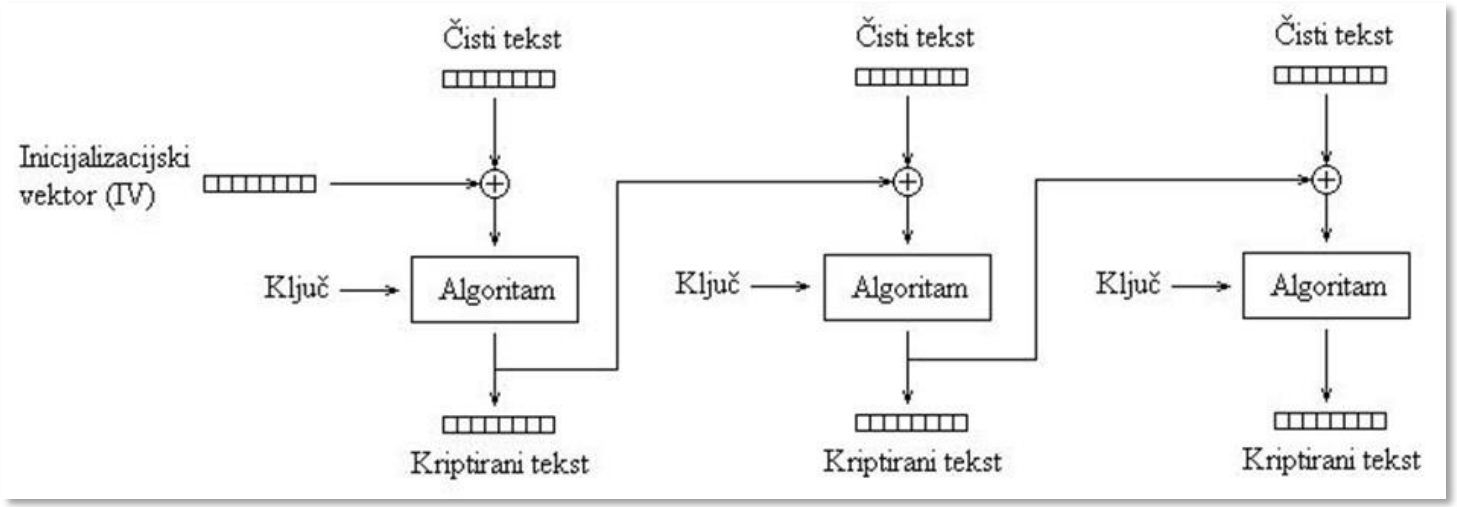
Načini šifriranja ECB – *Electronic Codebook*



Izvor: Budin, Golub, Jakobović, Jelenković, Operacijski sustavi

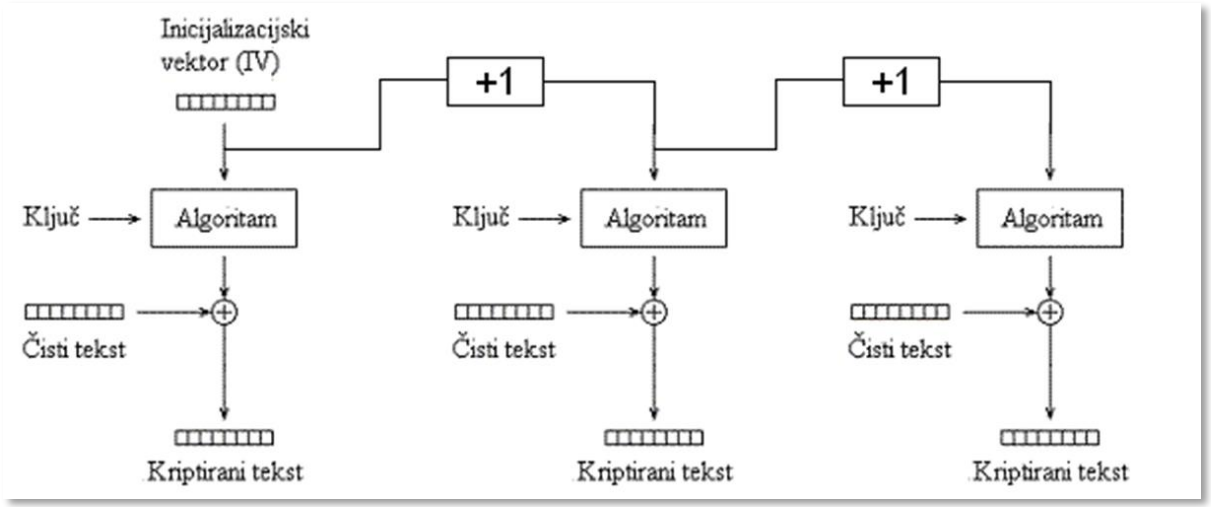
Načini šifriranja ECB – *Electronic Codebook*

Načini šifriranja CBC – *Cipher Block Chaining*



Izvor: Budin, Golub, Jakobović, Jelenković, Operacijski sustavi

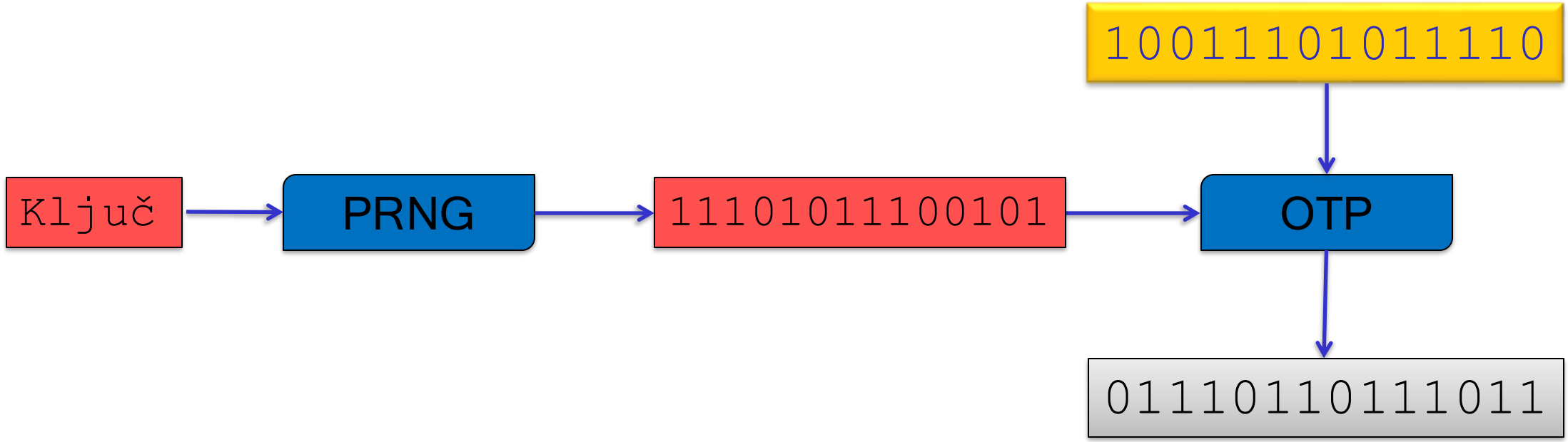
Načini šifriranja CTR – *Counter Mode*



Izvor: Budin, Golub, Jakobović, Jelenković, Operacijski sustavi

Protočna šifra (*stream cipher*)

Generator pseudoslučajnih brojeva na temelju ključa generira niz bitova koji se XOR-a s izvornim tekstom



Primjeri protočnih šifri

- RC4 (1987)
  - ključ veličine 40–2048 bitova
  - vrlo široko korišten, mnoštvo poznatih slabosti
- CSS (1996)
  - 40-bitni ključ
  - zaštita sadržaja na DVD-ovima
  - potpuno razbijen 1999. godine
- Salsa20/ChaCha (2005)
  - ključ 128 ili 256 bitova
  - podržan u TLS-u
  - alternativa AES-u zbog boljih performansi na uređajima gdje sklopovlje ne implementira AES

Kategorija 6: Ad-hoc razbijanje algoritma šifriranja

- Tipičan zadatak: zadan je skriveni tekst i postupak šifriranja koji ne koristi poznate algoritme.
  - Primjeri:
    - https://github.com/maple3142/My-CTF-Challenges/tree/master/TSJ%20CTF%202022/Top%20Secret
- Pristupi:
  - Analiziraj algoritam šifriranja
  - ??
  - Profit ☺
- Korisni alati:
  - Sage (Gaussova eliminacija)
  - Z3 SMT solver

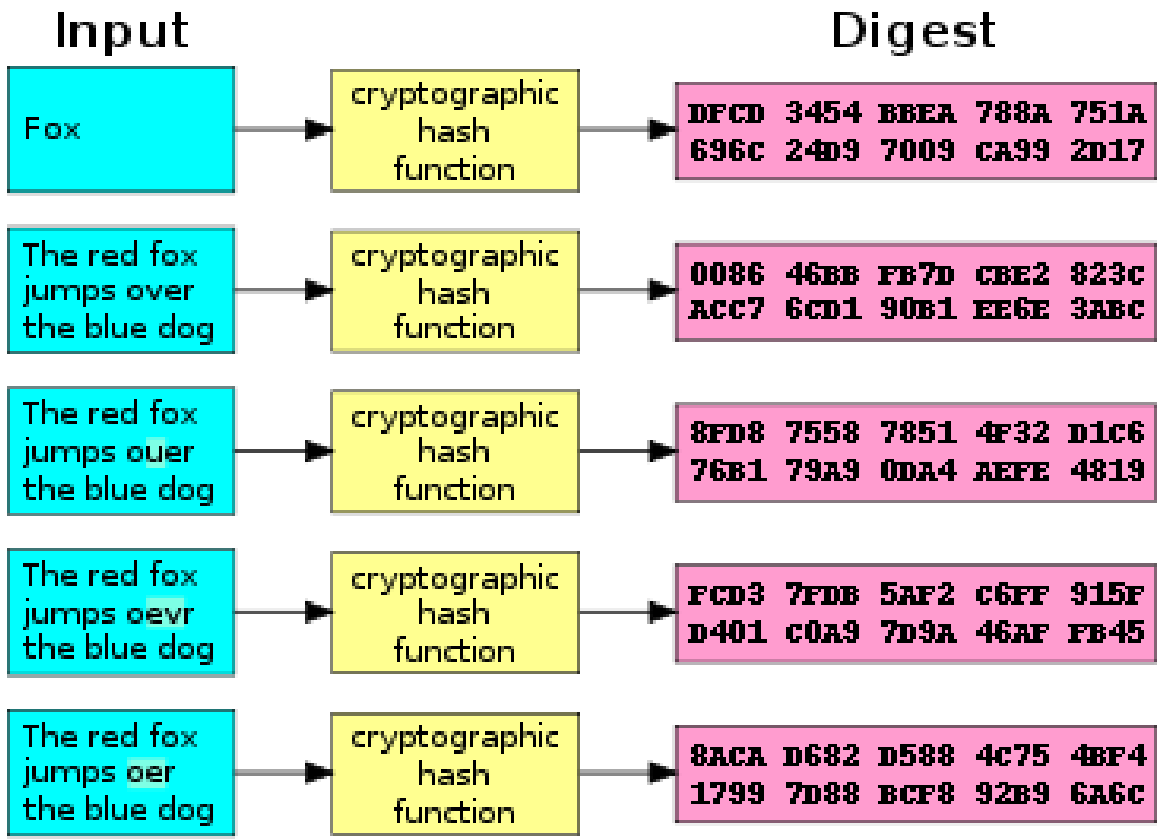
Osnove kriptografije i kriptoanalize

Kriptografske funkcije sažetka

Kriptografska funkcija sažetka (*hash)*

H je deterministički algoritam $H: \{0,1\}^* \rightarrow \{0,1\}^n$ koji proizvoljnoj *poruci* pridružuje *sažetak* fiksne duljine.

```
$ echo -n "fer" | sha1sum
cef48cb4569d34364e0e86067efa14fbe9b4591e  -
$ echo -n "fer" | sha1sum
cef48cb4569d34364e0e86067efa14fbe9b4591e  -
$ echo -n "Fer" | sha1sum
4514751a6511a102351de1f2b6abf0d6633c401f  -
$ sha1sum big.txt
0c496df552232e34beaba1e15046f87e147d14f6  big.txt
$ sha1sum empty.txt
da39a3ee5e6b4b0d3255bfef95601890afd80709  empty.txt
```



Izvor: wikipedia.org

Funkcije sažetka – sigurnost

- Želimo da se ponaša „kao da je potpuno slučajna” te da sažetak dokumenta u praksi jedinstveno određuje originalni dokument.
- Kriptografska funkcija sažetka H je *otporna na kolizije* ako je praktički nemoguće pronaći dvije različite poruke x i y takve da vrijedi $H(x) = H(y)$.

Kolizije uvijek postoje, ali ih je jako teško pronaći!

Funkcije sažetka – sigurnost

- Nije svaka hash funkcija kriptografska hash funkcija!
- *Checksum* (CRC32, CRC64, …) nije kriptografska hash funkcija!

Funkcije sažetka – primjene

- Integritet datoteka: Spremate važnu datoteku na FER web kako bi je dohvatili s drugog računala. Kako možete biti sigurni da administratori nisu promijenili vašu datoteku?

- Deduplikacija: Odredite koliko različitih datoteka postoji na disku vašeg računala i pronađite sve duplikate.

Funkcije sažetka – primjene u kriptografiji

- Integritet poruka
- Zaštita zaporki
- Deriviranje ključeva iz zaporki
- Generiranje pseudoslučajnih brojeva
- Digitalni potpisi
- *Proof-of-work* kod kriptovaluta
- *…*

Hash funkcije – napad grubom silom

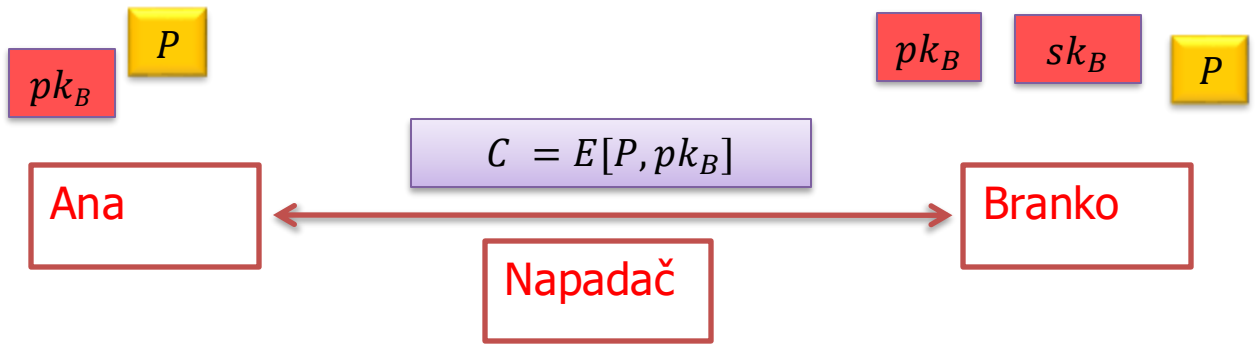
- Algoritam:
  1. Izaberi slučajnu poruku m
  2. Izračunaj $h = H(m)$ i zapamti par (h, m)
  3. Ako smo već vidjeli (h, m') gdje je $m' \neq m$ onda smo gotovi
  4. Skoči na korak 1.
- Iz paradoksa rođendana (*birthday paradox)* slijedi da je, u očekivanju, potrebno oko $1.2 * 2^{\frac{n}{2}}$ iteracija da se pronađe kolizija.

Osnove kriptografije i kriptoanalize

Asimetrične šifre (sustavi kriptiranja javnim ključem)

Javni i tajni ključevi

- Nova ideja: Primatelj ima dva ključa
  - Javni ključ pk_B: Javno poznat (npr. telefonski imenik)
  - Privatni ključ sk_B: Poznat samo Branku
  - Jasni tekst se šifrira s javnim ključem pk_B
  - Skriveni tekst se dešifrira s privatnim ključem sk_B



Teorija brojeva – notacija

- N – prirodni broj
- p, q – prosti brojevi
- $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$ – *prsten* u kojemu se zbraja, oduzima i množi modulo N
- Pišemo $a = b$ u $\mathbb{Z}_N$ umjesto $a \equiv b \pmod{N}$

Aritmetika u $\mathbb{Z}_N$

$$9 + 8 = 5 \text{ u } \mathbb{Z}_{12}$$
$$5 \cdot 7 = 11 \text{ u } \mathbb{Z}_{12}$$
$$7 - 9 = 10 \text{ u } \mathbb{Z}_{12}$$

Propozicija: Za aritmetiku u $\mathbb{Z}_N$ vrijede uobičajena svojstva komutativnosti, asocijativnosti i distributivnost (za sada nema dijeljenja u $\mathbb{Z}_N$).

Prosti brojevi i najveći zajednički djelitelj

- Prirodni broj je *prost* ako je veći od 1 i ako je djeljiv samo s brojem 1 i sa samim sobom.
- $k = nzd(x,\ y)$ – najveći zajednički djelitelj
  - Ako je $nzd(x,\ y) = 1$ onda kažemo da su x i y *relativno prosti.*

Propozicija: Neka su x i y cijeli brojevi i neka je k njihov *najveći zajednički djelitelj*, $k = nzd(x,\ y)$. Postoje cijeli brojevi a i b tako da vrijedi $ax + by = k$.
Brojevi a, b i k se mogu efikasno odrediti *proširenim Euklidovim algoritmom*.

Dijeljenje u $\mathbb{Z}_N$

- *Inverz* elementa $x \in \mathbb{Z}_N$ je element $y \in \mathbb{Z}_N$ takav da vrijedi $x \cdot y = 1$ u $\mathbb{Z}_N$.
- Inverz od x označavamo s x^{-1} (ako postoji)

Inverz od 2 u $\mathbb{Z}_{17}$? 9

Inverz od 4 u $\mathbb{Z}_{10}$? Ne postoji.

Propozicija: Broj x ima inverz u $\mathbb{Z}_N$ ako i samo ako je $nzd(x, N) = 1$.

Eulerova funkcija

- *Eulerova funkcija* $\varphi(N) = |\mathbb{Z}_N^*|$ je broj prirodnih brojeva manjih od N i relativno prostih s N.

$$\varphi(15) = 8$$

Ako je p prost onda $\varphi(p) = p - 1$

Ako su p i q različiti prosti brojevi onda je $\varphi(pq) = (p-1)(q-1)$.

Eulerov teorem

Teorem (Euler): Za svaki prirodni broj N i za svaki $a \in \mathbb{Z}_N^*$ vrijedi $a^{\varphi(N)} = 1$ u $\mathbb{Z}_N$.

Teorem (Fermat): Za svaki prosti broj p i za svaki $a \in \mathbb{Z}_p^*$ vrijedi $a^{p-1} = 1$ u $\mathbb{Z}_p$.

RSA – generiranje ključeva

Algoritam G:

1. Odaberem velike slučajne proste brojeve p i q
2. Izračunam $N = p \cdot q$
3. Izračunam $\varphi(N) = (p-1)(q-1)$
4. Odaberem proizvoljni $e \in \mathbb{Z}^*_{\varphi(N)}$ (u praksi $e = 65537$)
5. Izračunam $d = e^{-1}$ u $\mathbb{Z}^*_{\varphi(N)}$
6. Javni ključ: $pk = (e, N)$
7. Privatni ključ: $sk = (d, N)$

RSA – generiranje ključeva

Algoritam G:

1. Odaberem velike slučajne proste brojeve p i q
2. Izračunam $N = p \cdot q$
3. Izračunam $\varphi(N) = (p-1)(q-1)$
4. Odaberem proizvoljni $e \in \mathbb{Z}^*_{\varphi(N)}$ (u praksi $e = 65537$)
5. Izračunam $d = e^{-1}$ u $\mathbb{Z}^*_{\varphi(N)}$
6. Javni ključ: $pk = (e, N)$
7. Privatni ključ: $sk = (d, N)$

Ako je moguće N efikasno rastaviti na faktore onda je RSA nesiguran
Ako je moguće efikasno izračunati $\varphi(N)$ onda je RSA nesiguran

Obični RSA – enkripcija i dekripcija

Algoritam E:

- $E\big(m,(e,N)\big) = m^e \text{ u } \mathbb{Z}_N$

Algoritam D:

- $D\big(c,(d,N)\big) = c^d \text{ u } \mathbb{Z}_N$

e zovemo *javni eksponent*
d zovemo *privatni eksponent*
N zovemo *modul*
Otvoreni i skriveni tekst su brojevi u $\mathbb{Z}_N$

RSA

- Obični RSA nije siguran sustav kriptiranja javnim ključem ☹
  - Niti od napada poznatim izvornim tekstom.
  - Niti od napada odabranim tekstom.

Primjer 1

- Kriptiramo glasove na izborima
  - sudjeluju dva kandidata označena s 1 i 2
  - izborno povjerenstvo objavi svoj javni ključ pk.
  - glasač A izračuna $c_A = E(g_A, pk)$ gdje je $g_A \in \{1, 2\}$
  - glasač A šalje c_A izbornom povjerenstvu

- $E(1, pk) = 1$
- $E(2, pk) \neq 1$
- Napadač može zaključiti za koga je glasač glasao!

Primjer 2

- Kriptiramo datoteku
  - Datoteka se sastoji se od n bajtova $b_1, b_2, \dots, b_n$
  - kriptiramo svaki bajt zasebno $c_k = E(b_k, pk)$
  - šaljemo $c_1, c_2, \dots, c_n$ Wi-Fi mrežom

- Napadač može za svaki mogući bajt $b = 0, 1, \dots, 255$ izračunati $c = E(b, pk)$
- Kada vidi $c_1, c_2, \dots, c_n$ lagano nalazi $b_1, b_2, \dots, b_n$

Ako je algoritam enkripcije deterministički onda sustav kriptiranja javnim ključem nikako ne može biti siguran!

Primjer 3

- Šaljemo 128-bitni AES ključ K koristeći RSA
  - neka je $e = 3$
  - šaljemo $c = E(K, pk)$
  - Može li napadač odrediti K na temelju javnog ključa i c?

- K^3 ima oko manje od 400 bitova
- Prilikom enkripcije se ne dogodi redukcija modulo N
- Napadač može izračunati $K = \sqrt[3]{c}$

Primjer 4

- Šifriramo 128-bitni AES ključ K
  - Koristimo 2048-bitni RSA ključ u kojem je $e = 3$
  - šaljemo $c = E(K, pk)$
  - Može li napadač odrediti K na temelju javnog ključa i c?

Ponekad će se slučajno dogoditi da je $K = K_1 \cdot K_2$ gdje su K_1 i K_2 32-bitni brojevi

$$c = K^e = K_1^e \cdot K_2^e \text{ u } \mathbb{Z}_N$$

$$c \cdot (K_1^e)^{-1} = K_2^e \text{ u } \mathbb{Z}_N$$

Meet-in-the-middle algoritam nalazi K_1 i K_2 u 2^{32} koraka

RSA – Kombinacija sa simetričnom šifrom

- U praksi se RSA **gotovo nikada** ne koristi za kriptiranje podataka već za kriptiranje ključeva ili materijala za ključeve.
- 1 način: Digitalna omotnica: $E(Pad(k), pk), E_S(m, k)$
- 2 način: Kriptiranje materijala za ključ:

H je hash funkcija, E_s simetrična šifra

Algoritam E:

1. Izaberem slučajni $x \in \mathbb{Z}_N$
2. Izračunam $k = H(x)$
3. Izračunam $c_1 = E(x, pk)$
4. Izračunam $c_2 = E_s(m, k)$
5. Skriveni tekst je (c_1, c_2)

RSA – nadopunjavanje (*Padding)*

- Jasni tekst se uvijek nadopunjuje na zadanu veličinu!
- Postupak nadopunjavanja (padding) igra kritičnu ulogu i pažljivo je osmišljen.
  - PKCS#1 v1.5 (mnoštvo sigurnosnih problema)
  - OAEP

```
EME-PKCS1-v1_5 encoding:

a.  Generate an octet string PS of length k - mLen - 3
    consisting of pseudo-randomly generated nonzero octets.
    The length of PS will be at least eight octets.

b.  Concatenate PS, the message M, and other padding to form
    an encoded message EM of length k octets as

       EM = 0x00 || 0x02 || PS || 0x00 || M.
```

Izvor: ietf.org

RSA – Sigurnost

- Ako se RSA ispravno koristi smatramo ga sigurnim
  - Puno implementacijskih napada!
- Najbolji poznati općeniti napad
  - Faktorizacija modula
  - Na primjer, algoritmom GNFS (General Number Field Sieve)
  - U 2021. najveći faktorizirani modul je veličine 829 bitova

RSA – faktorizacija modula

paul zimmermann Paul.Zimmermann at inria.fr
Fri Feb 28 16:48:03 CET 2020

- Previous message: [Cado-nfs-discuss] move to gitlab
- **Messages sorted by:** [date] [thread] [subject] [author]

```
Date: February 28, 2020

For the past three months, ever since the DLP-240 record announced in
December 2019 [1], we have been in a historically unique state of
affairs: the discrete logarithm record (in a prime field) has been
larger than the integer factorization record. We are pleased to
rectify this situation with the factorization of RSA-250 from the RSA
challenge list:

RSA-250 =
2140324650240744961264423072839333563008614715144755017797754920881418023447140136643
        =
6413528947707158027879019017057738908482501474294344720811685963202453234463023862359
        *
3337202759497815655622601060535511422794076034476755466678452098702384172921003708025
```

```
This computation was performed with the Number Field Sieve algorithm,
using the open-source CADO-NFS software [2].

The total computation time was roughly 2700 core-years, using Intel Xeon
Gold 6130 CPUs as a reference (2.1GHz):

    RSA-250 sieving:  2450 physical core-years
    RSA-250 matrix:    250 physical core-years
```

Izvor: Arhiva mailing liste cado-nfs-discuss

Kategorija 7: Razni napadi na RSA

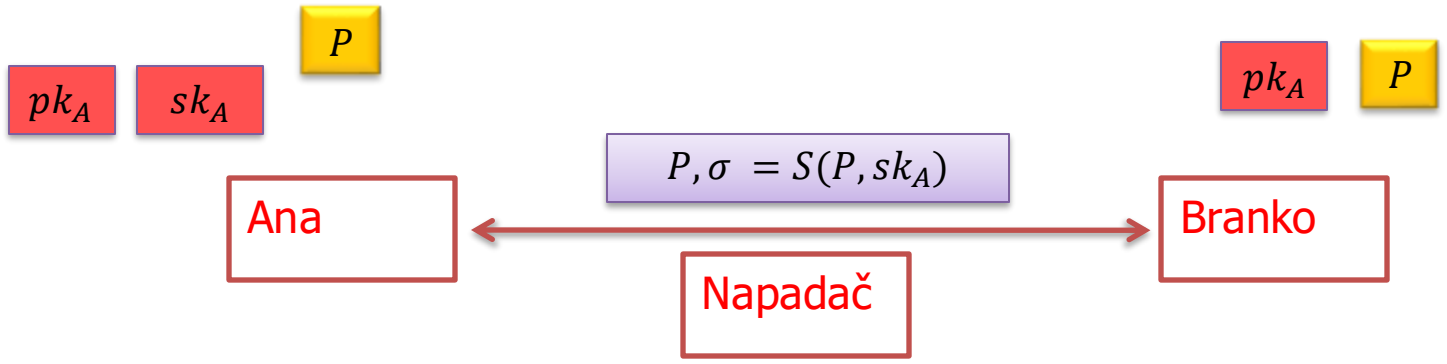
- Tipičan zadatak: zadan je i skriveni tekst i ključ i postupak šifriranja baziran na RSA algoritmu.
  - Primjer: https://github.com/infobip/infobip-ctf-2021/tree/master/crypto/rsa_lol/attachments
  - https://gitlab.com/CapTaaha/foobarctf-22/-/tree/main/crypto/new-intern/dist?ref_type=heads
- Pristupi:
  - Ručno pronađi matematički način da se dođe do privatnog ključa ili poruke.
  - Pretraživanjem literatura pronađi točan napad.
- Korisna literatura:
  - https://crypto.stanford.edu/~dabo/papers/RSA-survey.pdf
- Korisni alati:
  - https://github.com/Ganapati/RsaCtfTool
  - Sage

Osnove kriptografije i kriptoanalize

Digitalni potpisi

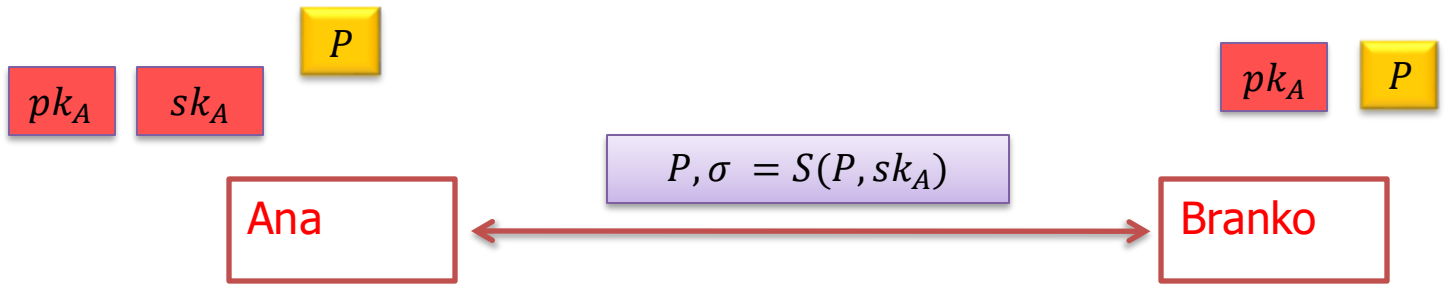
Javni i tajni ključevi

- Stara ideja: Svatko ima dva ključa
  - Javni ključ pk_A: Javno poznat (npr. telefonski imenik)
  - Privatni ključ sk_A: Poznat samo Ani
  - Ana *generira* potpis svojim privatnim ključem sk_A
  - Branko *provjerava* potpis Aninim javnim ključem pk_A



Sustav digitalnog potpisa

- Trojka *efikasnih* algoritama G, S i V
  - G – algoritam koji generira par ključeva pk, sk
  - $S(m, sk)$ – algoritam potpisivanja
  - $V(m, \sigma, pk)$ – algoritam verifikacije
- Za svaki par ključeva (pk, sk) generiranih algoritmom G i za svaki jasni tekst p vrijedi $V(p, S(p, sk), pk) = 1$



Sustav digitalnog potpisa – sigurnost

- SDP je siguran ako je teško odrediti bilo koju poruku p i bilo koji potpis (niz bitova) σ takav da
  - $V(p, \sigma, pk) = 1$
  - p nikad nije potpisan s privatnim ključem sk
- ... čak i ako napadač ima na raspolaganju:
  - Javni ključ pk
  - Mogućnost da dobije potpis $S(p, sk)$ za proizvoljnu poruku p (*chosen message attack)*

“Obični RSA” digitalni potpis (nesiguran)

Algoritam S:

- $S\big(m,(d,N)\big) = m^d \text{ u } \mathbb{Z}_N$

Algoritam V:

- $V\big(m,\sigma,(e,N)\big) = (\sigma^e == m \text{ u } \mathbb{Z}_N)\ ?\ 1 : 0$

Primjer 1

- Može li napadač na temelju javnog ključa (e, N) pronaći *bilo koju* poruku i njen ispravan potpis?

- Odaberem proizvoljni $x \in \mathbb{Z}_N$
- Izračunam $\mathrm{y} = x^e$ u $\mathbb{Z}_N$
- x je ispravan potpis za poruku y.

Primjer 2

- Pretpostavimo da napadač ima dvije poruke i njihove ispravne potpise, može li ih kombinirati tako da dobije ispravan potpis za neku novu poruku?

- $m_1, \sigma_1 = m_1^d$ u $\mathbb{Z}_N$
- $m_2, \sigma_2 = m_2^d$ u $\mathbb{Z}_N$
- $\sigma_1 \cdot \sigma_2 = m_1^d \cdot m_2^d = (m_1 \cdot m_2)^d$ u $\mathbb{Z}_N$
- $\sigma_1 \cdot \sigma_2$ je ispravan potpis od $m_1 \cdot m_2$

Primjer 3

- Napadač ima mogućnost dobiti potpis za točno jednu poruku koja izgleda slučajno. Želi iskoristiti tu mogućnost kako bi dobio potpis konkretne poruke m po njegovom izboru.

RSA digitalni potpis

H – kriptografska funkcija sažetka
Pad – funkcija nadopunjavanja

Algoritam S:

- $S\big(m,(d,N)\big) = Pad(H(m))^d \text{ u } \mathbb{Z}_N$

Algoritam V:

- $V\big(m,\sigma,(e,N)\big) = (Unpad(\sigma^e \text{ u } \mathbb{Z}_N) == H(m))\ ?\ 1 : 0$

Osnove kriptografije i kriptoanalize

Generiranje slučajnih brojeva

Ako nema slučajnih brojeva nema ni sigurnosti!

- Sigurnost svih sustava ovisi u pretpostavci da je moguće na slučajan način generirati:
  - Ključeve
  - Inicijalizacijske vektore
  - *Nonce* vrijednosti u prototokolima
  - …
- Ako napadač može predvidjeti ključeve, nikakva sigurnost nije garantirana!

Primjer: Netscape 1.1 (1995)

```
global variable seed;

RNG_CreateContext()
    (seconds, microseconds) = time of day; /* Time elapsed since 1970 */
    pid = process ID;  ppid = parent process ID;
    a = mklcpr(microseconds);
    b = mklcpr(pid + seconds + (ppid << 12));
     seed = MD5(a, b);

mklcpr(x) /* not cryptographically significant; shown for completeness */
    return ((0xDEECE66D * x + 0x2BBB62DC) >> 1);

MD5() /* a very good standard mixing function, source omitted */
```

Figure 2: *The Netscape 1.1 seeding process: pseudocode.*

Napad: Za sve moguće *seed* parametre

1. Generiraj ključ koristeći isti postupak
2. Provjeri je li moguće dešifrirati komunikaciju s dobivenim ključem

```
RNG_GenerateRandomBytes()
    x = MD5(seed);
    seed = seed + 1;
    return x;

global variable challenge, secret_key;

create_key()
    RNG_CreateContext();
    tmp = RNG_GenerateRandomBytes();
    tmp = RNG_GenerateRandomBytes();
    challenge = RNG_GenerateRandomBytes();
    secret_key = RNG_GenerateRandomBytes();
```

Figure 3: *The Netscape v1.1 key-generation process: pseudocode.*

Izvor: Goldberg, Wagner, „Randomness and the Netscape Browser”, 1996.

Generiranje slučajnih brojeva

- Dva sastojka:
  - Generator stvarno slučajnih vrijednosti s dovoljno *entropije*
  - Kriptografski generator pseudoslučajnih brojeva (*Pseudorandom number generator*)
- Upozorenje: „obični” generatori slučajnih brojeva (npr, `srand()` i `rand()` u jeziku C) su predvidivi i ne smiju se koristiti za kriptografiju!

Kategorija 8: Napad na generator slučajnih brojeva

- Tipičan zadatak: zadan je skriveni tekst i složeni postupak šifriranja, a ključ je generiran „običnim” generatorom slučajnih brojeva.
  - Primjer: https://github.com/infobip/infobip-ctf-2021/tree/master/crypto/otp/attachments
- Pristupi:
  - Odredi sjeme generatora ili na neki drugi način predvidi izlaz generatora na temelju dostupnih informacija.
  - Odredi ključ koji je korišten tijekom enkripcije.
- Korisni alati:
  - ??

Hvala!