

# Računalniške komunikacije

2020/21

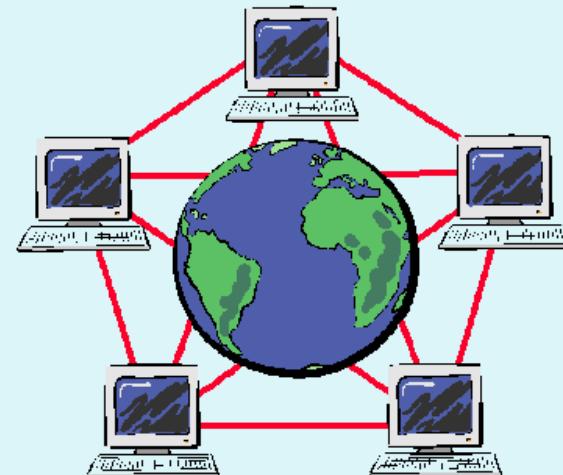
aplikacijska plast  
P2P (BitTorrent, Skype)

kriptografija  
uvod, simetrična, bločna

# P2P storitve

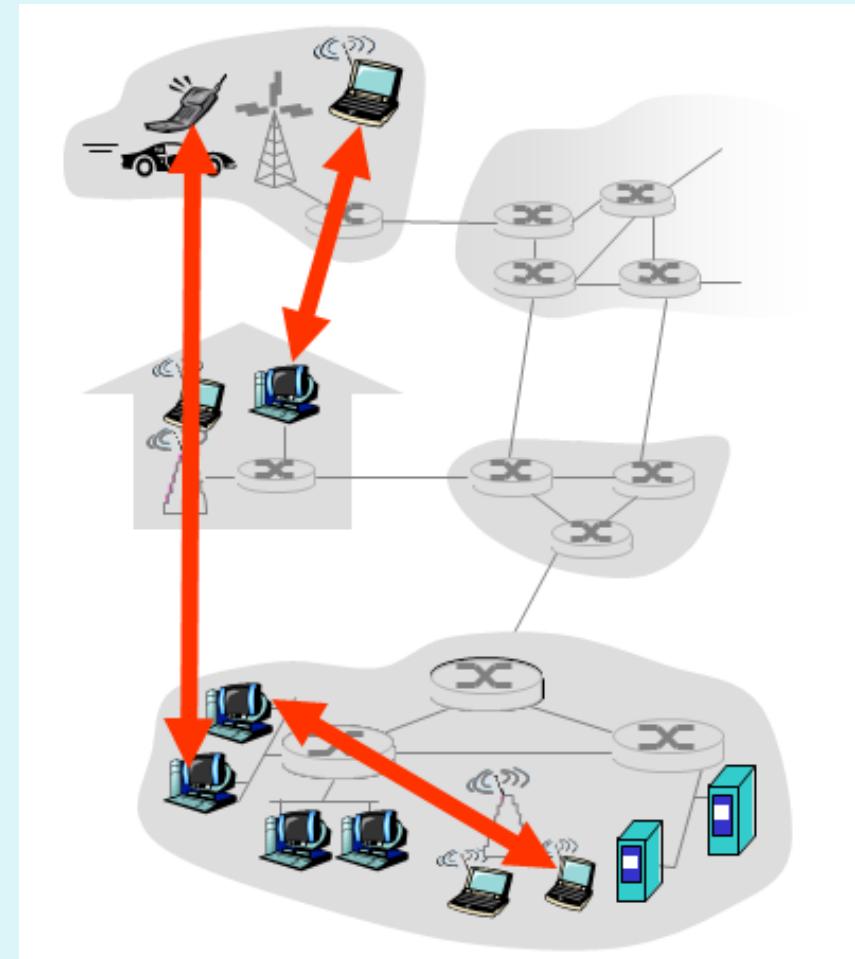
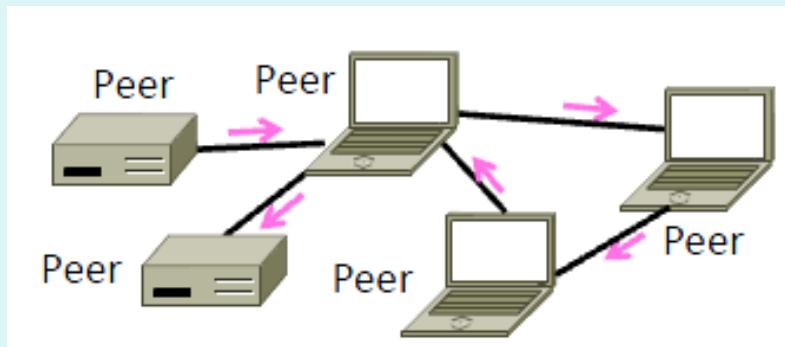
BitTorrent

Skype



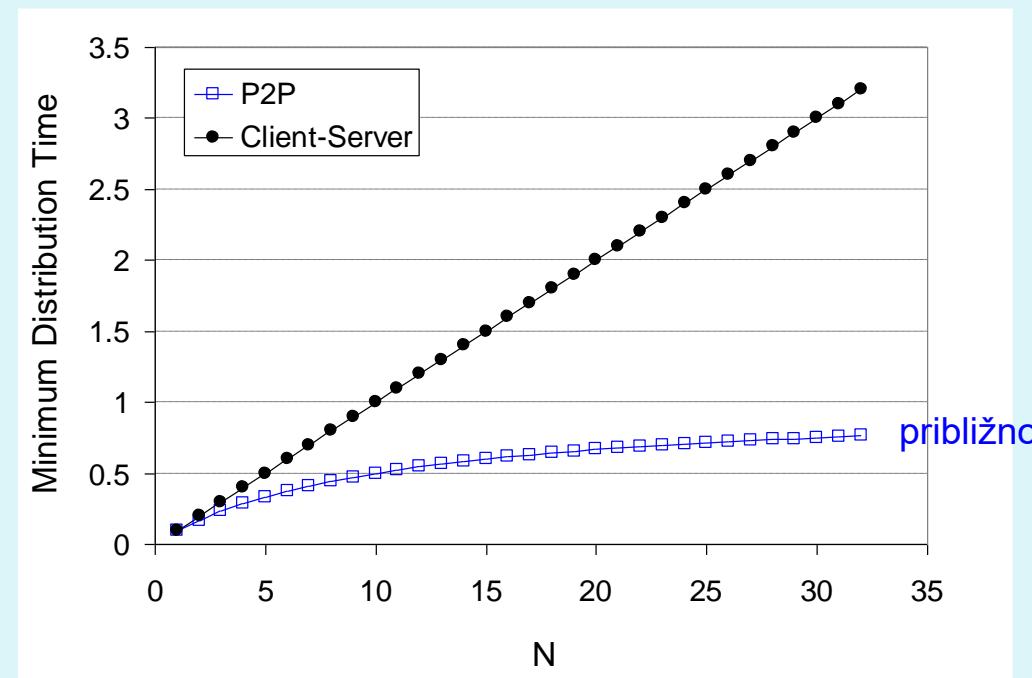
# Arhitektura P2P

- ni strežnika, ki je nenehno prižgan
- izmenjava podatkov med poljubnima končnima sistemoma
- odjemalci sodelujejo po potrebi, pri priklopu menjavajo IP naslov
- izzivi: varnost, iniciativnost pri sodelovanju, NAT



# Skalabilnost sistema P2P

- sistem klient-stežnik: čas prenosa linearno narašča s številom odjemalcev (N krat prenos istega podatka) št. uporabnikov \* velikost prenosa (npr. 1 uporabnik prenese 100 MB datoteko, za 2 je prenos že 200 MB itd.)
- sistem P2P: čas prenosa podatka je krajši, ker izmenjava poteka tudi med klienti (proste prenosne kapacitete) -> prednost: večja skalabilnost sistema

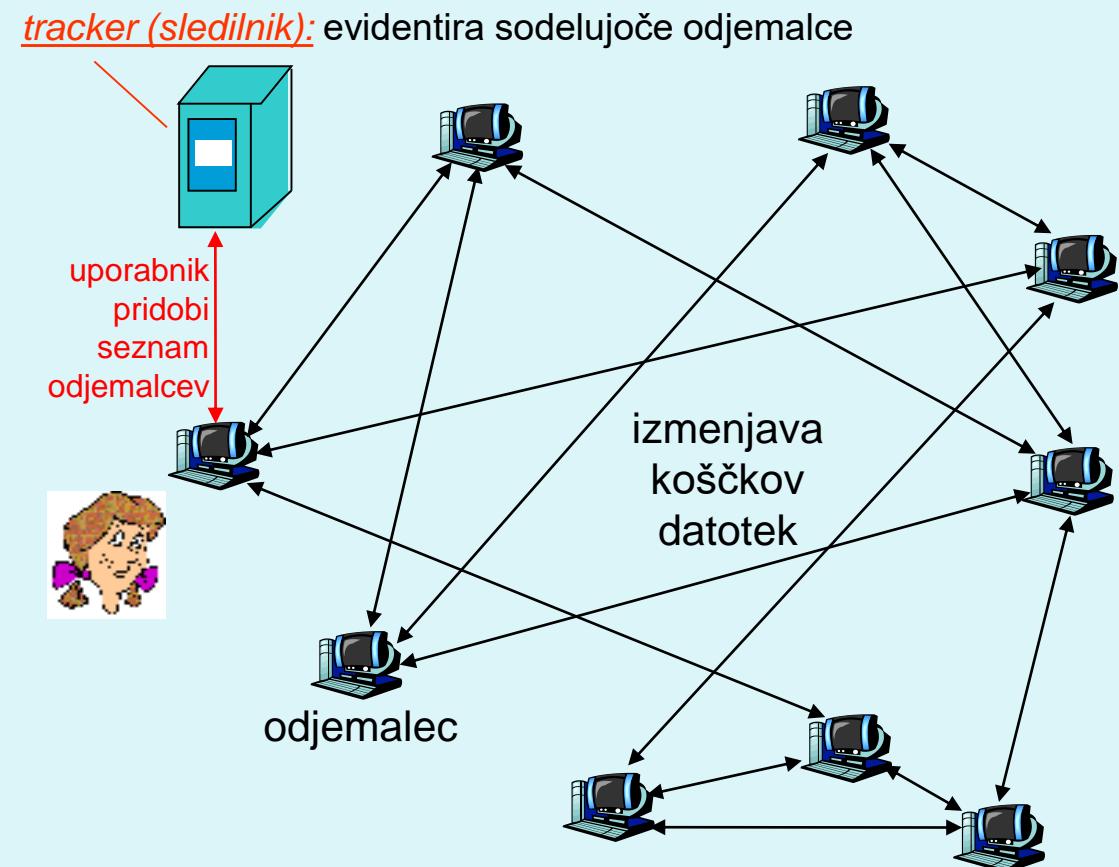


Sistem je skalabilen, če je kvaliteta njegovega delovanja z večanjem števila uporabnikov približno enaka kot pri manjšem številu uporabnikov.

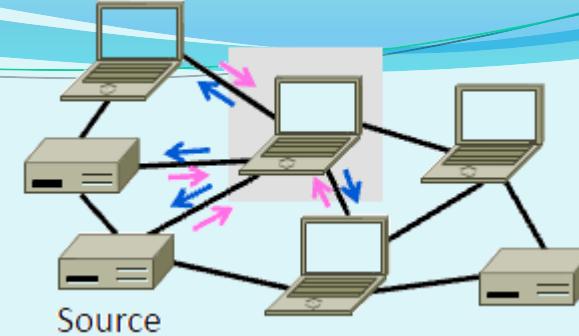
# BitTorrent

- sistem P2P za izmenjavo datotek
- torrent: skupina odjemalcev, ki si delijo kose (chunks) datotek
- odjemalci se lahko pridružijo/odidejo kadarkoli
- pridružitev novega odjemalca:
  - če še nima koščkov datotek, je cilj pridobiti s časom
  - prijavi se sledilnemu strežniku (*tracker*) in od njega pridobi seznam drugih odjemalcev
  - od vseh izjemalcev odjemalec izbere **podmnožico sosedov** (*neighbors*) in izvaja izmenjavo samo z njimi.

komuniciramo lahko z omejenim številom sosedov,  
to je omejeno s št. portov (cca 65k)



# BitTorrent

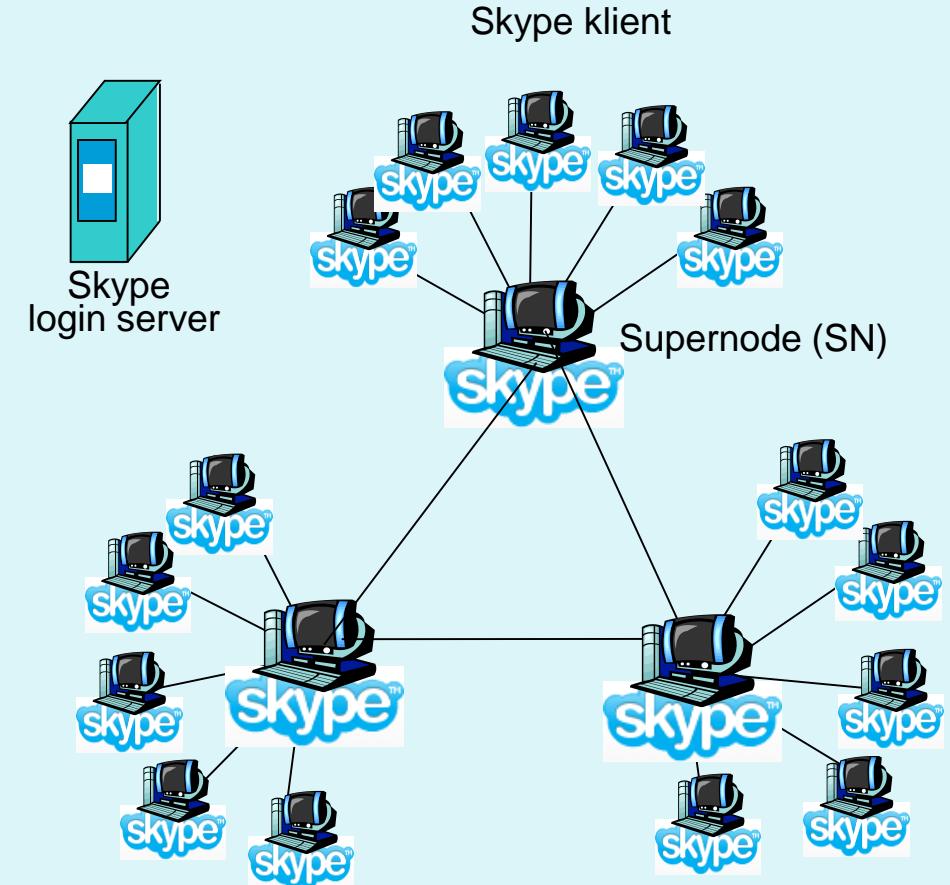


- **strategija komunikacije:**
  - **strategija:** sosede vpraša po razpoložljivih koščkih datotek in zahteva najprej tiste manjkajoče, ki so **najbolj redki med sosedji (*rarest first!*)** ker odjemalci prihajajo/odhajajo zagotovimo, da košček ne izgine
  - med prejemanjem koščkov datotek, pošilja koščke tudi drugim odjemalcem
  - **pravičnost:** opazujemo hitrost prejemanja od sosedov in jim pošiljamo koščke **s sorazmerno visoko hitrostjo**
  - **vzpodbuda za sodelovanje:** odjemalec lahko po prejemu datoteke ostane (radodarno) ali odide iz torrenta (sebično)  
ostanemo in pomagamo pri torrentu
- **aplikacijska sporočila** imajo kontrolna bita za status odjemalca: zamašen (angl. *choked*) in zainteresiran (angl. *interested*)
  - na začetku je stanje vsake povezave pri odjemalcu: *nezainteresiran/zamašen*
  - pretok podatkov se izvaja, kadar je eden zainteresiran, drugi pa ni zamašen
  - namen zastavice za zamašitev: nadzor zasičenja preko različnih TCP povezav

uravnavanje hitrosti pošiljanja za vsakega klienta posebej (na aplikacijski ravni!)

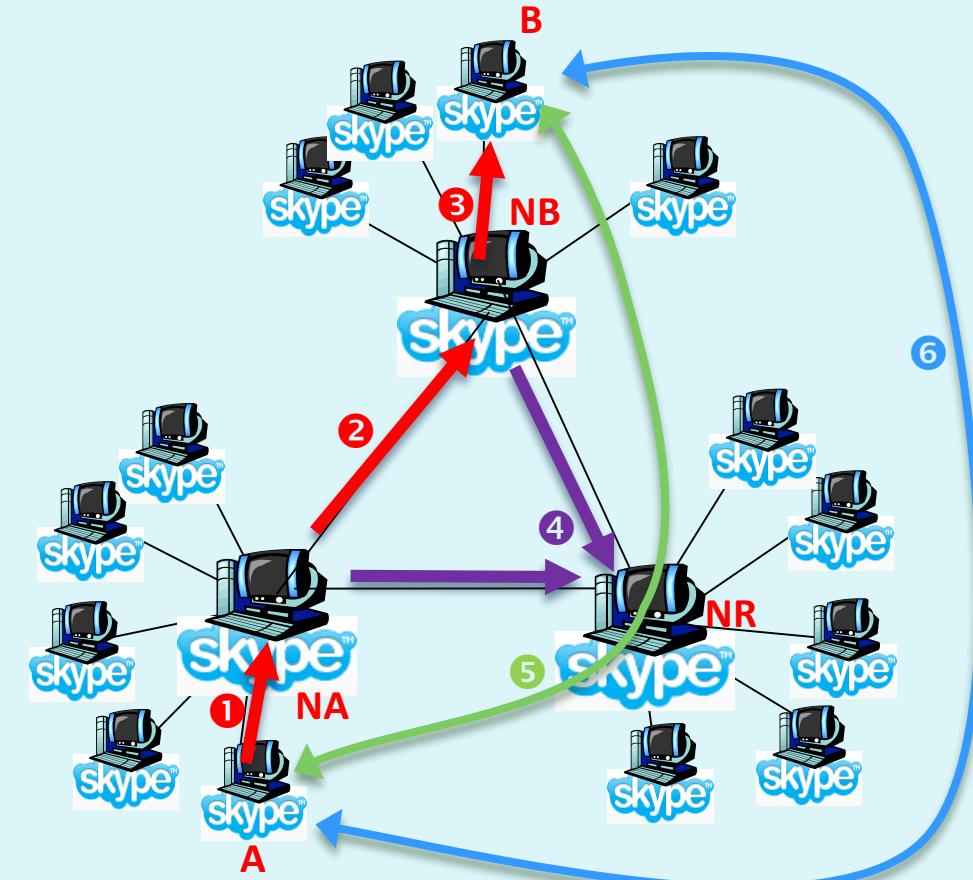
# Skype

- tudi P2P: komunikacija med poljubnimi uporabniki
- **lastniški aplikacijski protokol**, podrobnosti neznane (nekatere lastnosti pridobljene z **obratnim inženiringom**)
- strežnik za prijavo preverja podatke o uporabnikih
- nadzorna vozlišča (**supernodes**):
  - **hranijo preslikave uporabniško ime -> IP naslov**
  - *skrbijo za povezovanje med uporabniki*



# Premostitvena vozlišča (relay)

- NAT povzroča težave v P2P arhitekturah, ker zunanji odjemalci ne morejo direktno kontaktirati odjemalca za prehodom NAT
- rešitev:
  - odjemalce A in B vzpostavi zvezo preko nadzornih vozlišč NA in NB (1-3),
  - pri tem NA in NB izbereta tretje **premostitveno (relay) nadzorno vozlišče (NR)**, s katerim A in B vzpostavita sejo (4-5) (ker sta odjemalca prva vzpostavila zvezo, lahko prejema dohodni promet od premostitvenega vozlišča)
  - premostitveno vozlišče poskuša **zagotoviti neposredno povezavo med odjemalcema (6)**



# **Podporne storitve**

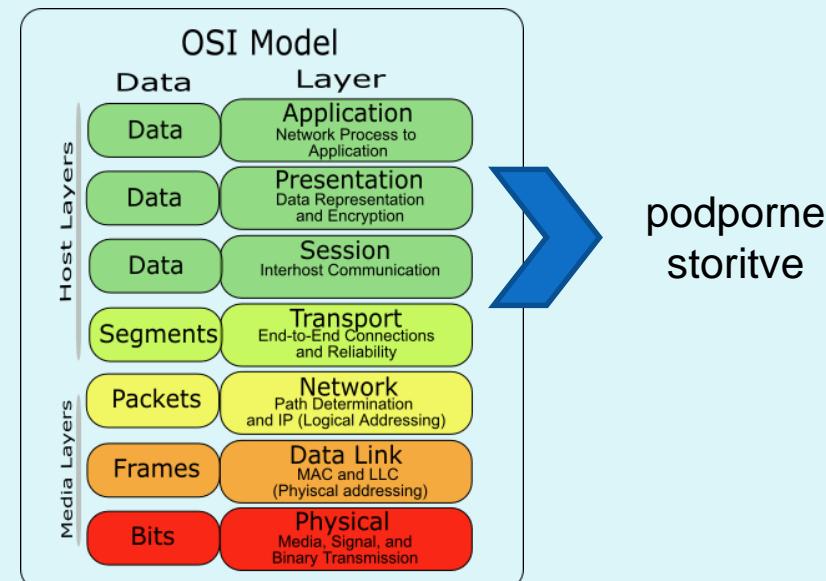
**Predstavitevna plast**

**Sejna plast**



# Aplikacijske storitve

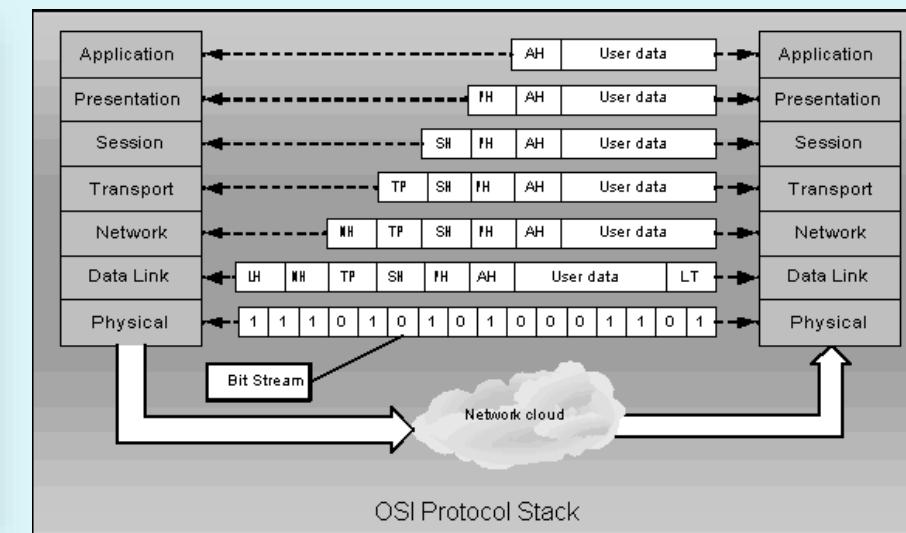
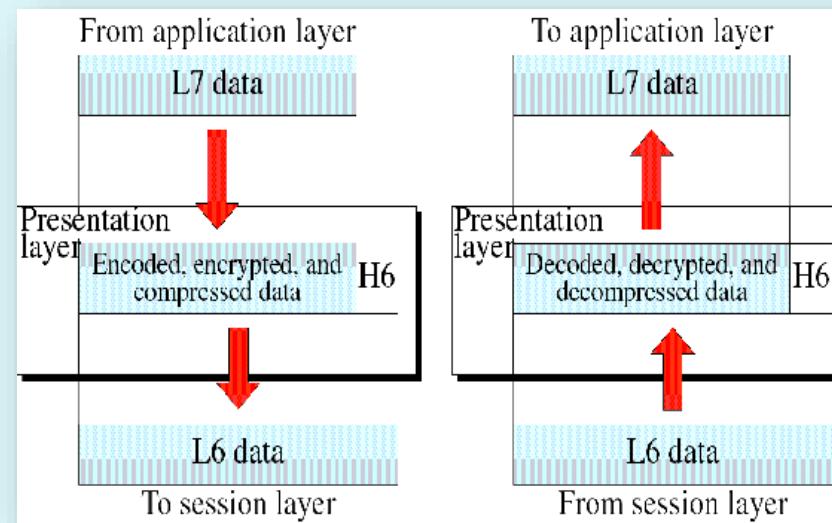
- delovati morajo v skladu s tehnološko infrastrukturo. Da lahko to zagotovimo, aplikacija potrebuje *podporne storitve*, ki v OSI modelu ležijo med aplikacijsko plastjo in omrežjem
- podporne storitve:
  - predstavitevna plast: usklajevanje predstavitev podatkov
  - sejna plast: logično povezovanje med aplikacijskimi procesi



# Storitve predstavitevne plasti

npr. little endian, big endian, različne predstavitve med arhitekturami (x86, ARM ...)

1. **predstavitev podatkov** (binarni tipi): različni sistemi predstavljajo binarne tipe na različen način; uporaba sintakse ASN.1 (Abstract Syntax Notation 1) zmanjša število vseh možnih preslikav
2. **predstavitev alfanumeričnih znakov** (združljivost kodnih strani): znaki so predstavljeni s številkami po kodnem sistemu, potrebno zagotoviti, da se prenašajo pravi znaki
3. **stiskanje podatkov**: omogoča zmanjšanje velikosti podatkov in s tem pohitritev prenosa (jpeg, mpeg)
4. **zaščita podatkov** (kriptiranje): varnostni mehanizem, ki omogoča vpeljavo zaupnosti v komunikacijo



# Storitve sejne plasti

- **naloge:**
  - vzpostavljanje, rušenje, vzdrževanje sej med aplikacijama (potek dialoga med aplikacijama)
  - odgovornost za obnovitvene točke (checkpoints) seje in obnovo (recovery), če seja ne uspe
  - sinhronizacija podatkov iz različnih tokov in virov
- **protokoli:** H.245 (Call Control Protocol for Multimedia Communication), RTCP (Real-time Transport Control Protocol), SCP (Session Control Protocol), ...
- **možni odnosi** med transportno in sejno plastjo:



Sejna povezava  
Transportna povezava



Več sejnih povezav  
Transportna povezava



Sejna povezava  
Več transportnih povezav

# Računalniške komunikacije

2020/21

kriptografija

(uvod, osnovne metode, simetrična)

# Omrežna varnost



- varnosti izzivi pri elektronski komunikaciji:
  - **zaupnost:** samo pošiljatelj in prejemnik naj bi lahko brala sporočilo
    - preprečevanje **prisluškovanja:** prestrezanje sporočil (pasivni napad)
  - **integriteta:** pošiljatelj in prejemnik želita zagotoviti, da sporočilo med prenosom ni bilo spremenjeno
    - prečevanje dodajanja, brisanja in **spreminjanje** sporočil (aktivni napad)
  - **identifikacija in avtentikacija:** pošiljatelj in prejemnik želita preveriti in potrditi medsebojni identiteti
    - preprečevanje **kraje identitete** (impersonation): napadalec lahko ponaredi izvorni naslov v paketu
    - preprečevanje **ugrabitve seje** (hijacking): napadalec odstrani pošiljatelja in prevzame njegovo vlogo
    - omogočanje **avtorizacije:** katere aktivnosti lahko uporabnik izvaja v sistemu
    - preprečevanje **zanikanja komunikacije** non-repudiation  
*(nekdo nekaj pošlje in potem zanika da je to poslal - to preprečujemo)*
- primeri aplikacij: elektronske transakcije preko spletja (npr. nakupi), elektronsko bančništvo, DNS strežniki, usmerjevalniki, ...

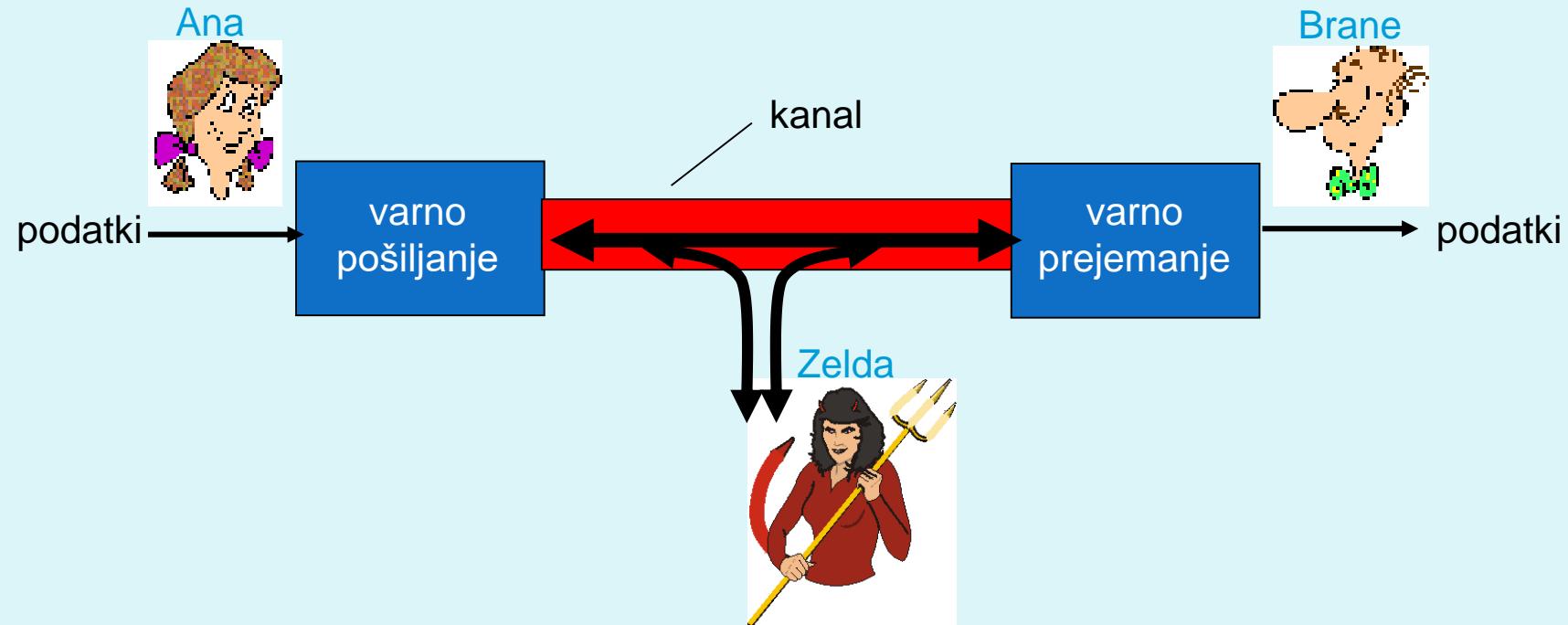
# Omrežna varnost



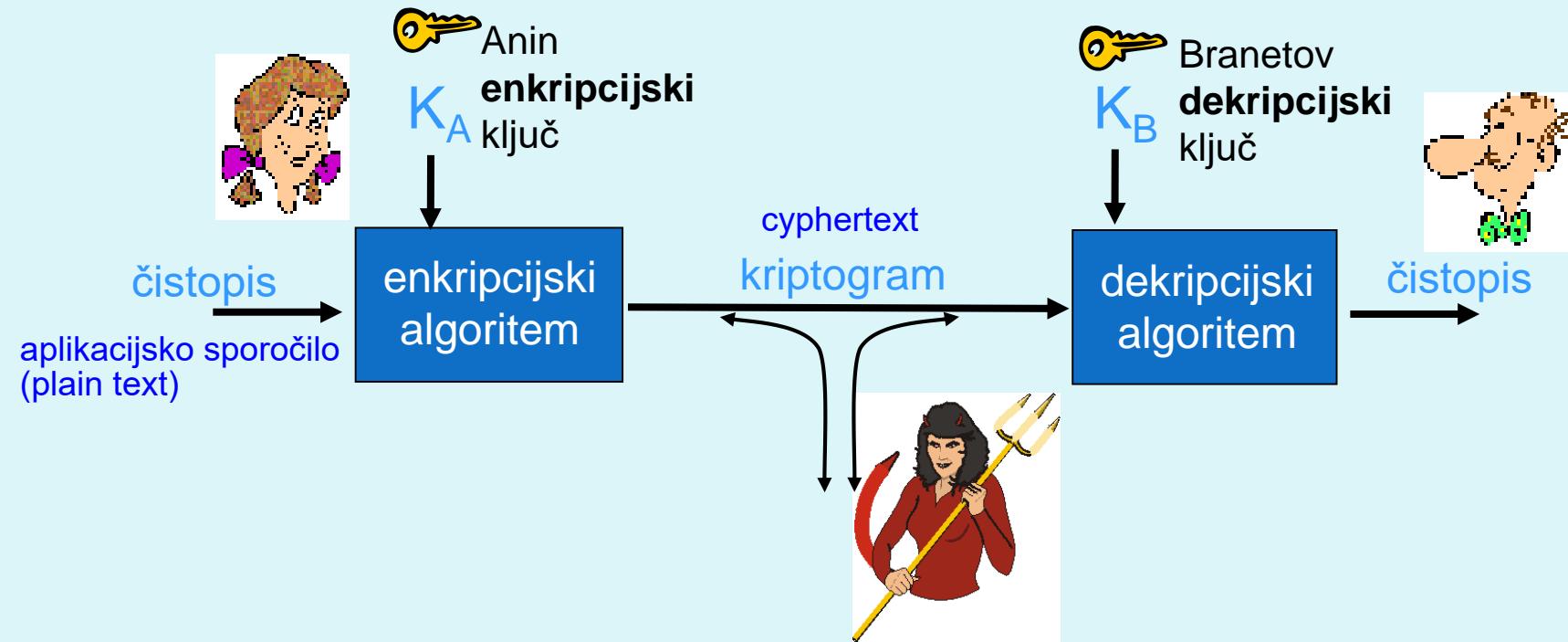
- **varnost v praksi** (operativna varnost):
  - naprave: **požarni zidovi**, sistemi za **zaznavanje/preprečevanje vdorov**,
  - dodaten cilj: preprečevanje **onemogočanja storitev** (denial of service): onemogoči uporabo storitev (npr. s preobremenitvijo virov)
- varnost je **potrebna na vseh plasteh**: na aplikacijski, transportni, omrežni in povezavni plasti
  - **fizična plast**: kriptiranje povezave
  - **omrežna plast**: filtriranje paketov, opazovanje prometa
  - **transportna plast**: kriptiranje povezav med dvema procesoma
  - **aplikacijska plast**: avtentikacija na podlagi preverjanja identitet

# "Prijatelji" in "sovražniki"

- pošiljatelj in prejemnik (Ana in Brane) želita **varno** komunicirati
- za zagotovitev varne komunikacije uporabita postopke, s katerimi zavarujeta sporočilo na kanalu pred sovražniki (Zelda, Sauron, Tracy, ... ?)



# Terminologija



$m$  (message)

$K_A(m)$

$m = K_B(K_A(m))$

čistopis

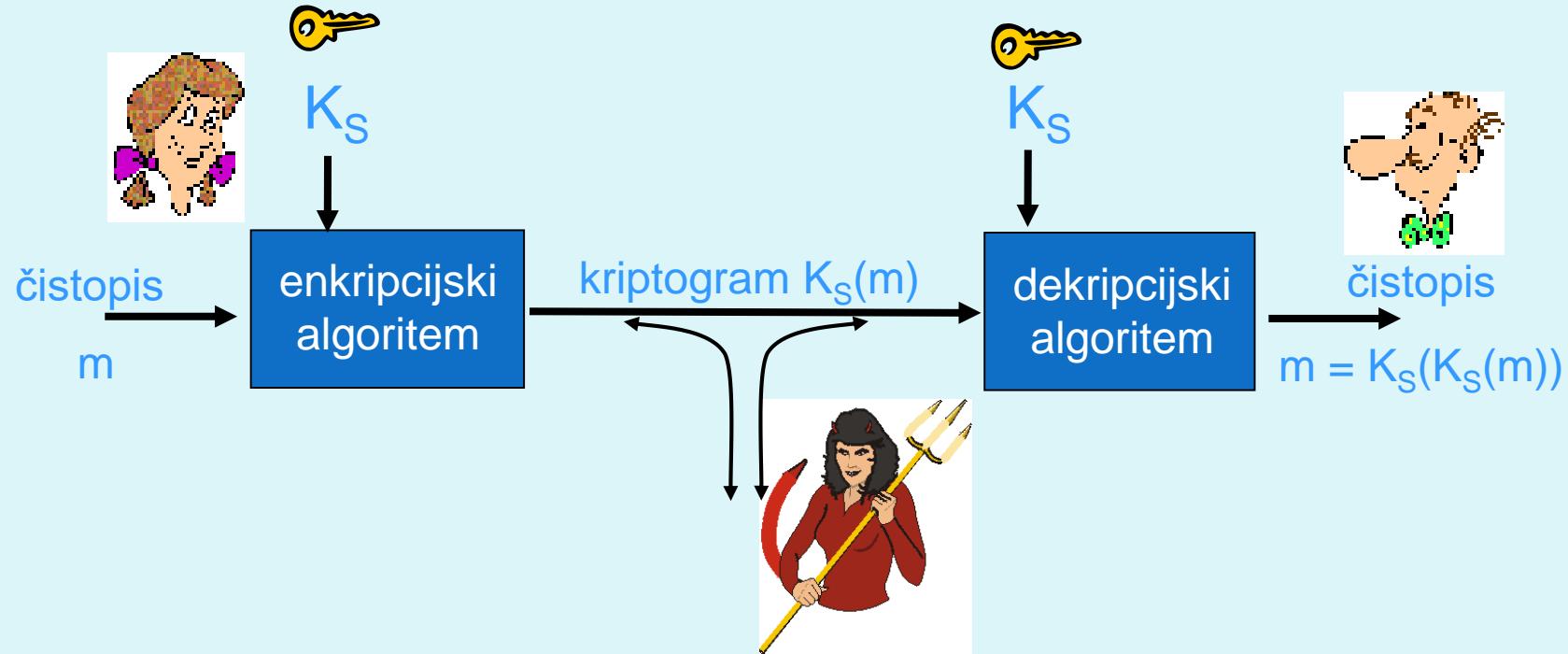
kriptogram, kriptiran s ključem  $K_A$

odkriptiran čistopis

# Kriptografija

- običajno se uporablja
  - algoritmom, ki je **znan** vsem
  - **tajni** so samo ključi
- **vrste kriptografije glede na ključe:**
  - **simetrični ključi:** enkripcijski in dekripcijski ključ sta enaka      (simetrična kriptografija)
  - **javni ključi:** uporaba **dveh ključev, enega javnega in drugega tajnega**      (asimetrična kriptografija)
- zgoščevalne funkcije (*hash functions*)
  - so tudi enkripcijski algoritmom, vendar ne uporablja ključev
  - kdaj je to lahko koristno?

# Kriptografija s simetričnim ključem



- pošiljatelj in prejemnik uporablja isti (simetričen) ključ
- primer ključa: dogovor o zamenjavi znakov v sporočilu (substitucijski vzorec)
- *kako si pošiljatelj in prejemnik varno izmenjata ključ?*

# Metode kriptiranja

- glede na način kriptiranja (algoritem)
  - **substitucija** (zamenjava znakov z drugimi)
    - Cezarjev Kriptogram
    - Vigenèr-jev kriptogram
    - Porterjev kriptogram
  - **transpozicija** (zamenjava vrstnega reda znakov)
  - **sodobne (kombinirane) metode**
- glede na velikost sporočila (podatkov)
  - znakovna
  - bločna (kriptiramo zaporedja znakov)
- glede na uporabljene ključe
  - **simetrična** kriptografija (oba udeleženca uporabljata enak ključ)
    - znakovna (bit po bit)
    - bločna (sporočilo razbijemo na bloke, vsakega kriptiramo neodvisno)
  - **kriptografija z javnimi ključi** (ključa za enkripcijo in dekripcijo sta različna)

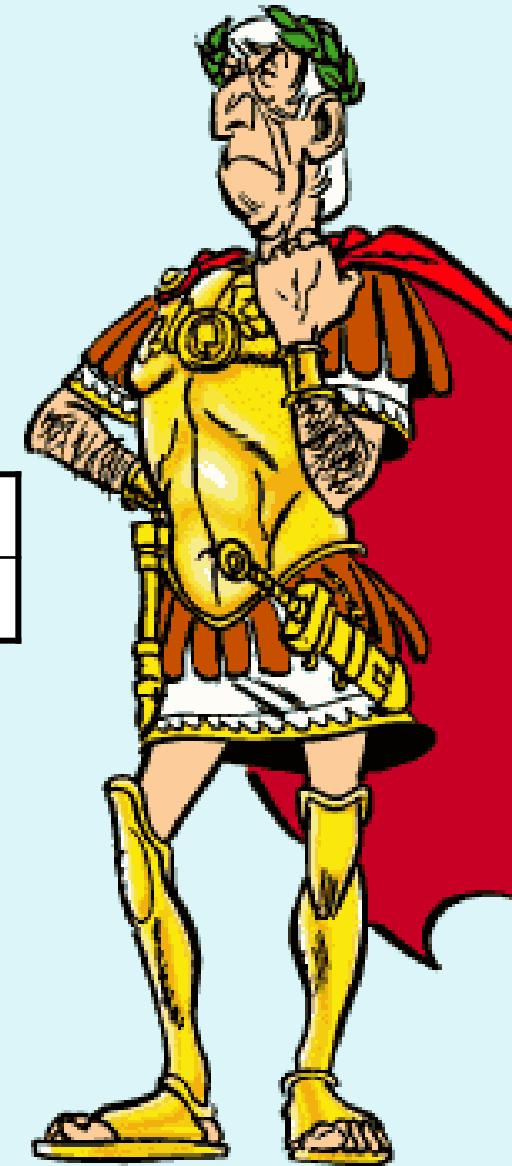


# Klasične metode: Cezar

- Cezarjev kriptogram: substitucija z zamikom za  $k$  črk
  - ključ je  $k$  (velikost zamika)
  - kriptogram JULUA = čistopis ??? ( $k = 21$ )

a	b	c	č	d	e	f	g	h	i	j	k	l	m	n	o	p	r	s	š	t	u	v	z	ž
u	v	z	ž	a	b	c	č	d	e	f	g	h	i	j	k	l	m	n	o	p	r	s	š	t

- imamo 25 možnih ključev
  - kriptogram razbijemo v največ 25 poskusih



# Kriptoanaliza substitucijskih kriptogramov

Kriptoanaliza (razbijanje kriptogramov) na osnovi:

- **poznanega besedila** (npr. "please login" ali "HTTP/1.1")
    - zato kriptiramo le vsebino, ne cele komunikacije
  - **statistike jezika** (črke, besede, dvo- ali tročrkovni sklopi – potrebno je daljše besedilo).
  - **poznavanja vsebine** (semantika) olajša razbijanje – iščemo pričakovane korene besed ipd.
- 
- Glej:
    - primer: <http://www.richkni.co.uk/php/crypta/index.php>

# Vigenèr-jev kriptogram – večabecedno kriptiranje

- Viegnerjeva matrika: vse Cesarjeve abecede.

izbira abecede  
glede na ključ

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z			
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z				
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z					
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z						
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z							
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z								
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z									
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z										
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z											
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z												
N	O	P	Q	R	S	T	U	V	W	X	Y	Z													
O	P	Q	R	S	T	U	V	W	X	Y	Z														
P	Q	R	S	T	U	V	W	X	Y	Z															
Q	R	S	T	U	V	W	X	Y	Z																
R	S	T	U	V	W	X	Y	Z																	
S	T	U	V	W	X	Y	Z																		
T	U	V	W	X	Y	Z																			
U	V	W	X	Y	Z																				
V	W	X	Y	Z																					
W	X	Y	Z																						
X	Y	Z																							
Y	Z																								
Z																									

črke sporočila

- ključ = niz D črk, vsaki pripada ena vrstica (enaka 1. črka).
- z abecedo n-te črke gesla kriptiramo  $n$ -to,  $n+D$ -to,  $n+2D$ -to ... črko sporočila.
- prost ključ
- statistika jezika in semantika postaneta nemočni



# Vigenèr-jev kriptogram (primer)

- Geslo: računalniške komunikacije
- Sporočilo: Junija vsi izpiti na žalost odpadejo, razen pri ekonomiki, septembra pa bo spet vse po starem

r	a	č	u	n	a	l	n	i	š	k	e	k	o	m	u	n	i	k	a	c	i	j	e	
j	u	n	i	j	a	v	s	i	i	z	p	i	t	i	n	a	ž	a	l	o	s	t	o	
d	p	a	d	e	j	o	r	a	z	e	n	p	r	i	e	k	o	n	o	m	i	k	i	
S	e	p	t	e	m	b	r	a	p	a	b	o	s	p	e	t	v	s	e	p	o	s	t	
a	r	e	m																					

za ta stolpec uporabimo  
ključ R, za naslednjega  
uporabimo ključ A itd.

- dolžina ključa: D=24
- Prvi stolpec črk (torej 1., 25. (1+24), 49. (1+48) črko) kriptiramo z 18. abecedo (r), itd.

# Porterjev kriptogram

- Kriptiramo po 2 znaka hkrati.
- Simboli so v tabeli – vrstica za en, stolpec za drugi znak.

	a	b	c	č	d	e	f	g	h	i	j	k	l	m
a	⌚	✂️	☎️	⏳	⌚	💣	😊	✡️	🕷️	🏡	😢	🕸️		
b	👓	🔔	☺️	👀	🖱️	🌲	🌐	🎗️	▣					
c	✉️	🕯️	❄️	👋	🍽️	*								
č	▶	⟳	🦋											... itd ...
d														

- npr. KAČA = ⌚️⏳

# Kodiranje

kodiranje  $\neq$  kriptiranje !!!

- cel znak ali besedo nadomestimo z drugo
- ni splošnega pravila za zamenjave
- ključ predstavlja cela kodna tabela



"Bugger! I was just about to crack his code, when he burnt his blanket."

# Metode kriptiranja

- glede na način kriptiranja (algoritem)
  - **substitucija** (zamenjava znakov z drugimi)
    - Cezarjev Kriptogram
    - Vigenèr-jev kriptogram
    - Porterjev kriptogram
  - **transpozicija** (zamenjava vrstnega reda znakov)
  - **sodobne (kombinirane) metode**
- glede na velikost sporočila (podatkov)
  - znakovna
  - bločna (kriptiramo zaporedja znakov)
- glede na uporabljene ključe
  - **simetrična** kriptografija (oba udeleženca uporabljata enak ključ)
    - znakovna (bit po bit)
    - bločna (sporočilo razbijemo na bloke, vsakega kriptiramo neodvisno)
  - **kriptografija z javnimi ključi** (ključa za enkripcijo in dekripcijo sta različna)



# Transpozicijski kriptogram

- spremenimo zaporedje znakov ali delov besedila
- uporabimo ključ, oštevilčimo črke po abecedi
- zapišemo stolpce glede na oštevilčenje črk

k	o	p	r	i	v	a
3	4	5	6	2	7	1
J	u	n	i	j	a	n
a	ž	a	l	o	s	t
v	s	i	i	z	p	i
t	i	o	d	p	a	d
e	j	o	b	l	a	b

kriptogram bi torej bil:  
ntidb jozpl Javte užsij naioo ilidb aspaa

# Sodobna simetrična kriptografija

- **bločna kriptografija:** sporočilo  $m$  kriptiramo tako, da obdelamo posamezne bloke  $k$  bitov (npr. bloke po 64 bitov)
- preslikava med bloki sporočila in kriptograma je bijektivna (enolična)
- primer, če  $k=3$

vhod	izhod
000	110
001	111
010	101
011	100
100	011
101	010
110	000
111	001

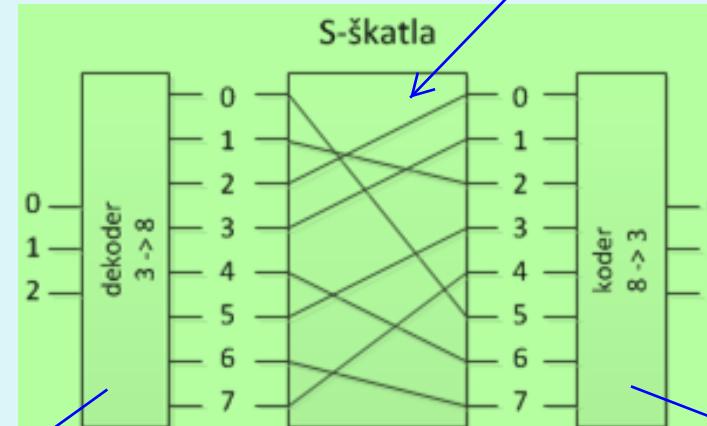
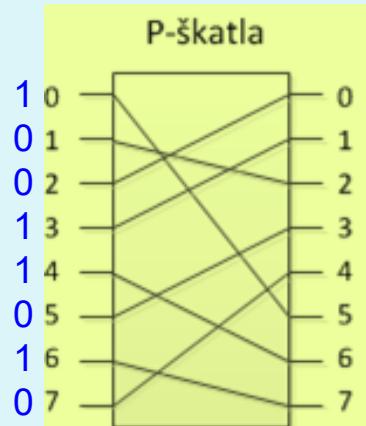
Sodobne metode kriptografije ne kriptirajo vsakega znaka (črke) posebej, ampak kriptirajo po blokih

Kakšen je kriptogram čistopisa 010110001111?

# Bločna kriptografija

PAZI! To pomeni, da je ničti bit na vhodu drugi bit na izhodu, prvi bit je tretji bit na izhodu, drugi bit je prvi bit na izhodu itd.

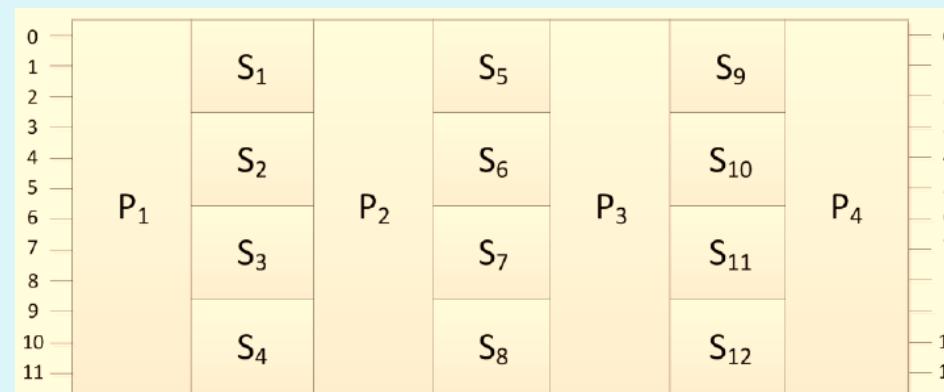
- permutacijska škatla (s ključem 23157046) menja zaporedje posameznih bitov v vhodnem zapisu
- substitucijska škatla (dekoder-permutacija-koder)



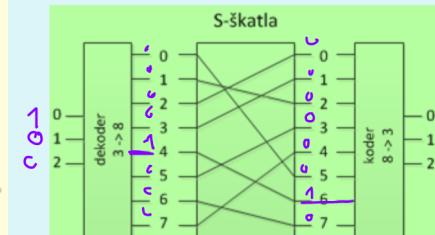
čistopis	kriptogram
000	101
001	010
010	000
011	001
100	110
101	011
110	111
111	100

- možno je kombiniranje škatel v preslikovalno kaskado za poenostavitev logike

na vhod prejme 3 bite,  
na izhodu prižge EN IZHOD



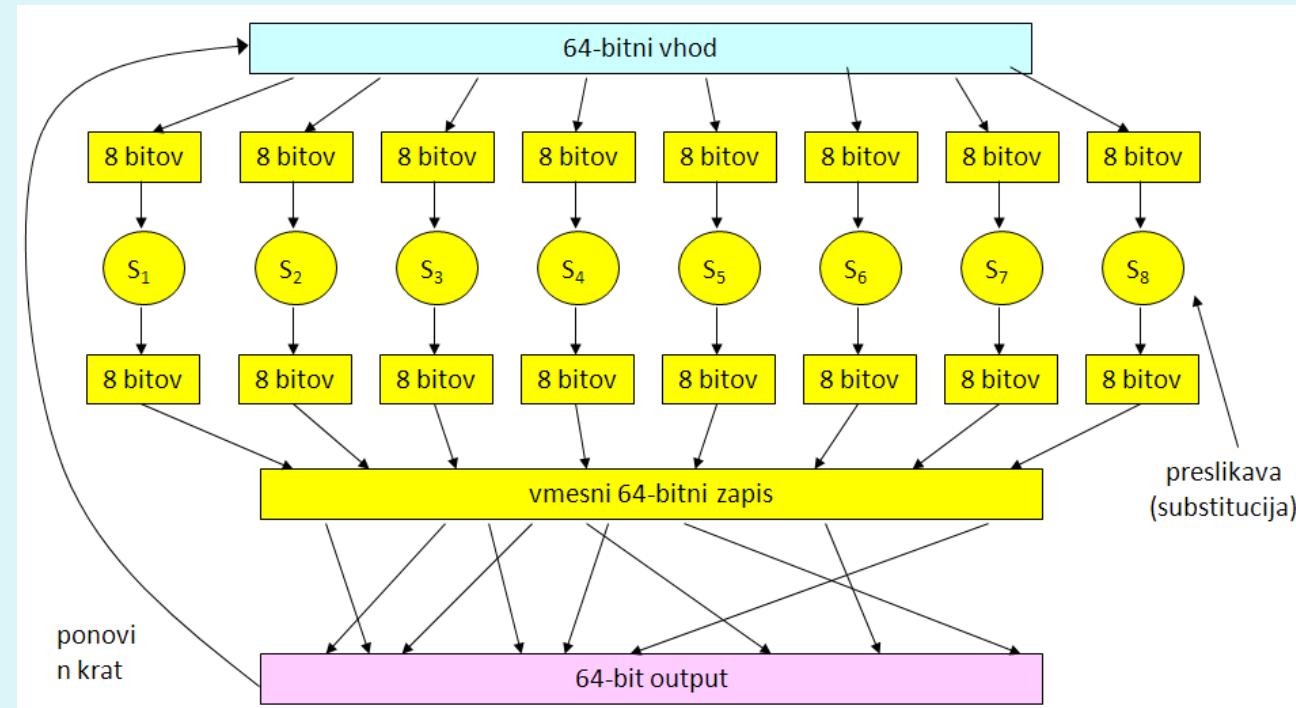
koder dela obratno kot dekoder  
(na vhod prejme nek vhod,  
na izhodu prižge bite na izhodu)



čistopis	kriptogram
000	101
001	010
010	000
011	001
100	110
101	011
110	111
111	100

# Bločna kriptografija

- **problem:**
  - če  $k=3$ , imamo  $40320 (=2^k!=8!)$  permutacij vseh možnih vhodov (možno razvozlati kodo na domačem PC računalniku)
  - če  $k=64$ , je permutacij ogromno ( $=2^{64}!$ ) in je težko hraniti tabelo permutacij (težka tudi izmenjava ključev)
- **rešitev:** uporabimo preprosto funkcijo za kriptiranje, ki pa simulira veliko tabelo

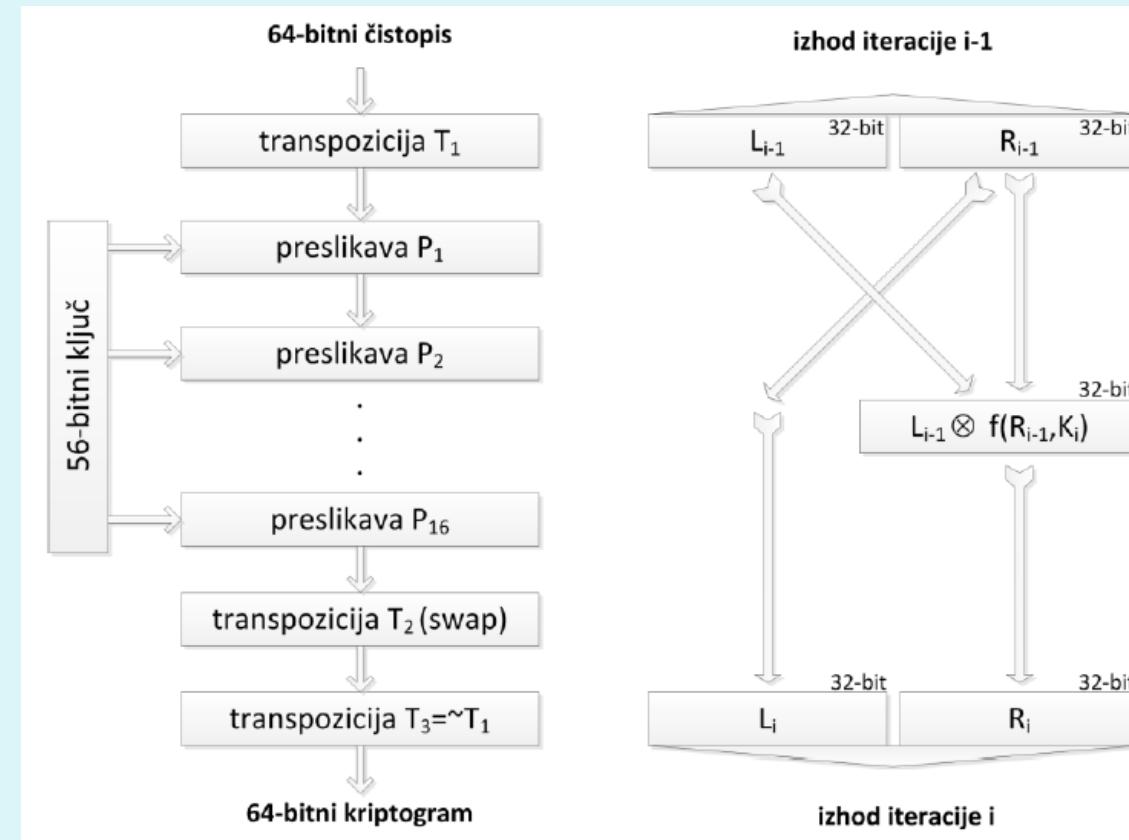


# Bločna kriptografija

- **primeri algoritmov** za bločno kriptografijo
  - DES (*Data Encryption Standard*)
  - 3DES (3-kratni DES s 3 različnimi ključi)
  - AES (*Advanced Encryption Standard*)
- zgornji algoritmi uporabljajo KLJUČE
  - DES 64-bitne bloke in 56-bitni ključ;
  - AES 128-bitne bloke in 128/192/256-bitne ključe

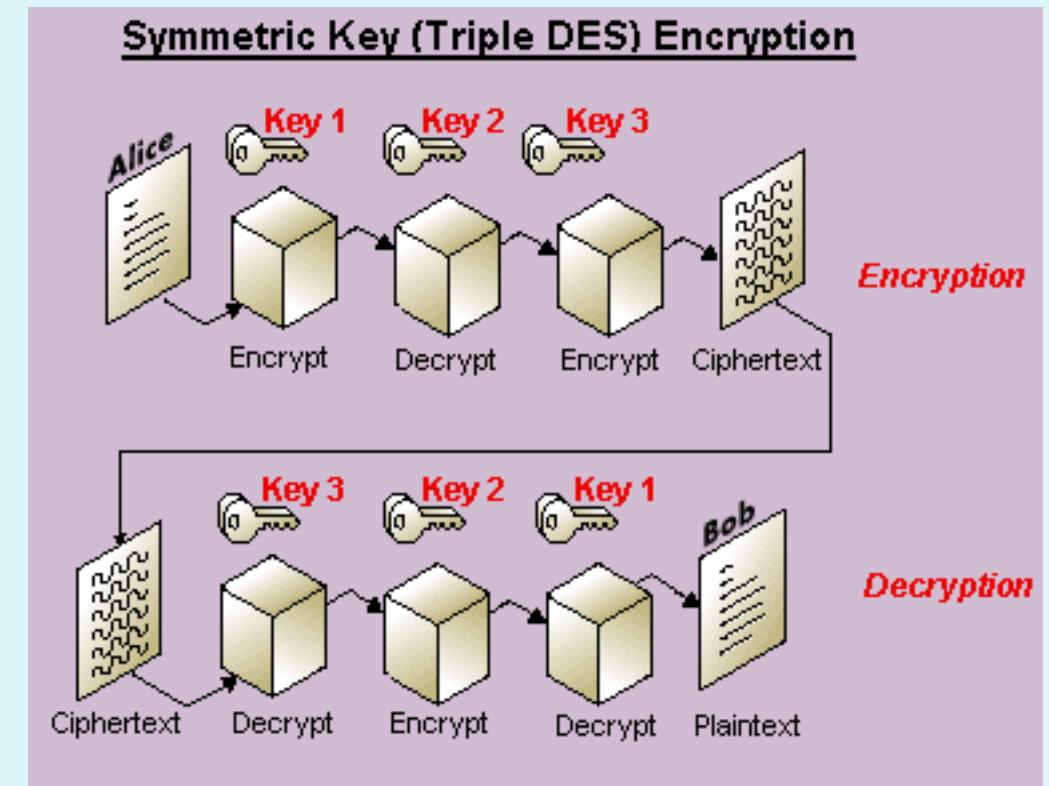
# DES (Data Encryption Standard)

- DES je kombinacija transpozicijskih in substitucijskih metod
- transpozicija -> 16 preslikav (ključ!) -> SWAP -> transpozicija



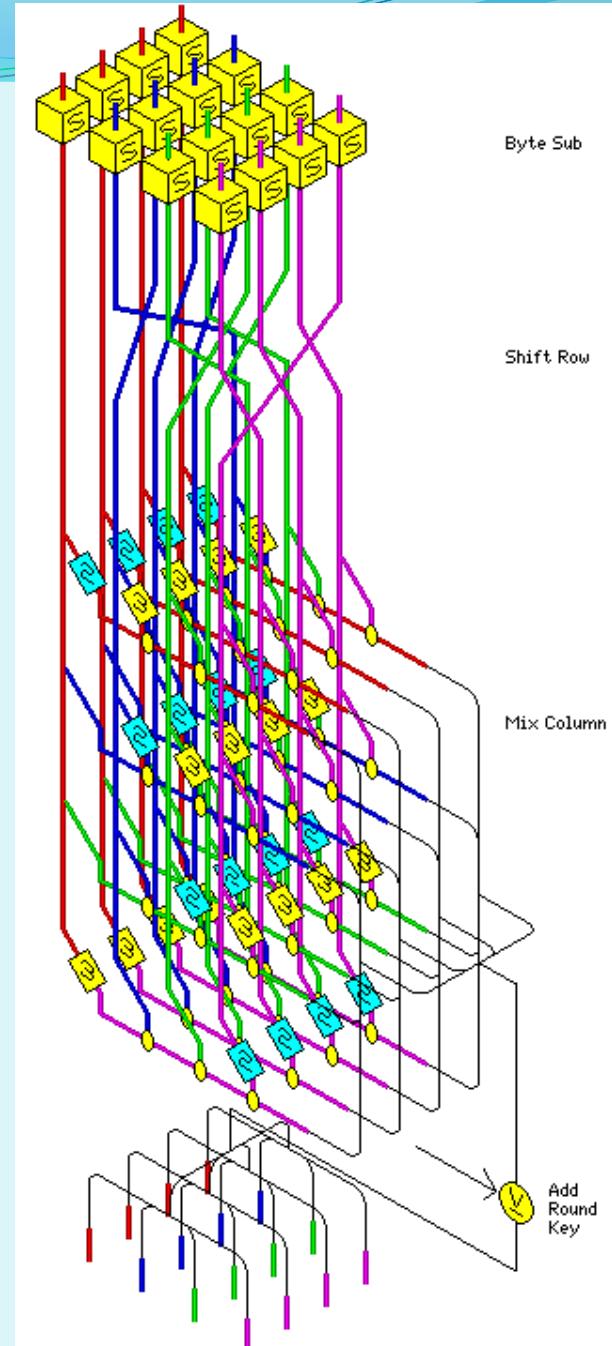
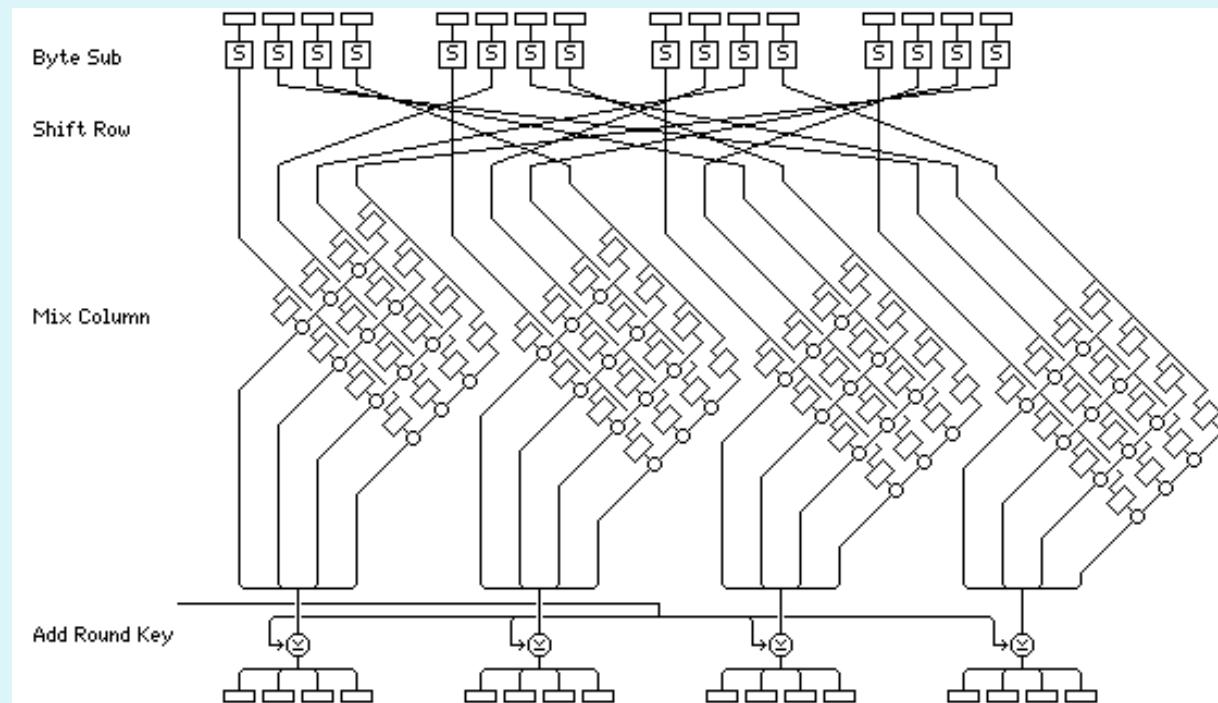
# 3DES (trojni DES)

- 3 x kriptiranje (kriptiranje, dekriptiranje, kriptiranje)
- 3 x počasnejši
- $2^{2 \cdot 56}$  krat varnejši za napad z grobo silo, ker uporablja 2 dodatni kriptiranji s 56-bitnim ključem
- združljivost z DES (če  $K_2 == K_3$ )



# AES

- algoritem Rijndael, najbolj učinkovita in varna metoda
- bloki 128 bitov, ključ 128/192/256 bitov
- dober računalnik bi potreboval  $10^{10}$  let za razbijanje
- različne AES operacije: zamikanje vrstic, zamenjave stolpcev, izpeljave ključev



# Naslednjič gremo naprej!

- asimetrična kriptografija
  - kriptografski principi

