

# Classical Cryptography

Encodings and digital ciphers

Karst Koymans

Informatics Institute  
University of Amsterdam

(version 19.2, 2020/02/20 14:16:34 UTC)

Thursday, February 20, 2020

- 1 Codes and ciphers
- 2 Public codes
- 3 From public codes to ciphers
- 4 Digital ciphers

# Outline

- 1 Codes and ciphers
- 2 Public codes
- 3 From public codes to ciphers
- 4 Digital ciphers

# Codes and codebooks

- A **code** operates on **semantic** components
  - Words, paragraphs, ...
- A **codebook** is a **lookup table** for codes
- Codes can be very hard to cryptanalyze
- Possible analysis methods
  - Compare many different coded texts
  - Use side channels (other available sources)
  - Try to identify cribs
  - Build up knowledge over time

## An example codebook

all the same for	159	218 3c	274 000	322	387 C	441 R	492 9	5
under go me complete	159 0	218 0/1/1	275 0/1/1	322 0/1/1	387 0/1/1	441 0/1/1	492 0/1/1	5
	160 change	219 0/1/1	276 0/1/1	323	388 0/1/1	442 0/1/1	493 0/1/1	5
	161	221	277	324	389	443	494 0/1/1	5
162 a	162	222	278	325	390	444	495 0/1/1	5
163 ad	163 che	223	279	326	391	445	496 0/1/1	5
164	164	224	280	327	392	446	497 0/1/1	5
165	165	225	281	328	393	447	498 0/1/1	5
166	166 a	226	282	329	394	448	499 0/1/1	5
167	167	227	283	330	395	449	500 0/1/1	5
168	168	228	284	331	396	450	501 0/1/1	5
169	169	229	285	332	397	451	502 0/1/1	5
170	170	230	286	333	398	452	503 0/1/1	5
171	171	231	287	334	399	453	504 0/1/1	5
172	172	232	288	335	400	454	505 0/1/1	5
173	173	233	289	336	401	455	506 0/1/1	5
174	174	234	290	337	402	456	507 0/1/1	5
175	175	235	291	338	403	457	508 0/1/1	5
176	176	236	292	339	404	458	509 0/1/1	5
177	177	237	293	340	405	459	510 0/1/1	5
178	178	238	294	341	406	460	511 0/1/1	5
179	179	239	295	342	407	461	512 0/1/1	5
180	180	240	296	343	408	462	513 0/1/1	5
181	181	241	297	344	409	463	514 0/1/1	5
182	182	242	298	345	410	464	515 0/1/1	5
183	183	243	299	346	411	465	516 0/1/1	5
184	184	244	300	347	412	466	517 0/1/1	5
185	185	245	301	348	413	467	518 0/1/1	5
186	186	246	302	349	414	468	519 0/1/1	5
187	187	247	303	350	415	469	520 0/1/1	5
188	188	248	304	351	416	470	521 0/1/1	5
189	189	249	305	352	417	471	522 0/1/1	5
190	190	250	306	353	418	472	523 0/1/1	5
191	191	251	307	354	419	473	524 0/1/1	5
192	192	252	308	355	420	474	525 0/1/1	5
193	193	253	309	356	421	475	526 0/1/1	5
194	194	254	310	357	422	476	527 0/1/1	5
195	195	255	311	358	423	477	528 0/1/1	5
196	196	256	312	359	424	478	529 0/1/1	5
197	197	257	313	360	425	479	530 0/1/1	5
198	198	258	314	361	426	480	531 0/1/1	5
199	199	259	315	362	427	481	532 0/1/1	5
200	200	260	316	363	428	482	533 0/1/1	5
201	201	261	317	364	429	483	534 0/1/1	5
202	202	262	318	365	430	484	535 0/1/1	5
203	203	263	319	366	431	485	536 0/1/1	5
204	204	264	320	367	432	486	537 0/1/1	5
205	205	265	321	368	433	487	538 0/1/1	5
206	206	266	322	369	434	488	539 0/1/1	5
207	207	267	323	370	435	489	540 0/1/1	5
208	208	268	324	371	436	490	541 0/1/1	5
209	209	269	325	372	437	491	542 0/1/1	5
210	210	270	326	373	438	492	543 0/1/1	5
211	211	271	327	374	439	493	544 0/1/1	5
212	212	272	328	375	440	494	545 0/1/1	5
213	213	273	329	376	441	495	546 0/1/1	5
214	214	274	330	377	442	496	547 0/1/1	5
215	215	275	331	378	443	497	548 0/1/1	5
216	216	276	332	379	444	498	549 0/1/1	5
217	217	277	333	380	445	499	550 0/1/1	5
218	218	278	334	381	446	500	551 0/1/1	5
219	219	279	335	382	447	501	552 0/1/1	5
220	220	280	336	383	448	502	553 0/1/1	5
221	221	281	337	384	449	503	554 0/1/1	5
222	222	282	338	385	450	504	555 0/1/1	5
223	223	283	339	386	451	505	556 0/1/1	5
224	224	284	340	387	452	506	557 0/1/1	5
225	225	285	341	388	453	507	558 0/1/1	5
226	226	286	342	389	454	508	559 0/1/1	5
227	227	287	343	390	455	509	560 0/1/1	5
228	228	288	344	391	456	510	561 0/1/1	5
229	229	289	345	392	457	511	562 0/1/1	5
230	230	290	346	393	458	512	563 0/1/1	5
231	231	291	347	394	459	513	564 0/1/1	5
232	232	292	348	395	460	514	565 0/1/1	5
233	233	293	349	396	461	515	566 0/1/1	5
234	234	294	350	397	462	516	567 0/1/1	5
235	235	295	351	398	463	517	568 0/1/1	5
236	236	296	352	399	464	518	569 0/1/1	5
237	237	297	353	400	465	519	570 0/1/1	5
238	238	298	354	401	466	520	571 0/1/1	5
239	239	299	355	402	467	521	572 0/1/1	5
240	240	300	356	403	468	522	573 0/1/1	5
241	241	301	357	404	469	523	574 0/1/1	5
242	242	302	358	405	470	524	575 0/1/1	5
243	243	303	359	406	471	525	576 0/1/1	5
244	244	304	360	407	472	526	577 0/1/1	5
245	245	305	361	408	473	527	578 0/1/1	5
246	246	306	362	409	474	528	579 0/1/1	5
247	247	307	363	410	475	529	580 0/1/1	5
248	248	308	364	411	476	530	581 0/1/1	5
249	249	309	365	412	477	531	582 0/1/1	5
250	250	310	366	413	478	532	583 0/1/1	5
251	251	311	367	414	479	533	584 0/1/1	5
252	252	312	368	415	480	534	585 0/1/1	5
253	253	313	369	416	481	535	586 0/1/1	5
254	254	314	370	417	482	536	587 0/1/1	5
255	255	315	371	418	483	537	588 0/1/1	5
256	256	316	372	419	484	538	589 0/1/1	5
257	257	317	373	420	485	539	590 0/1/1	5
258	258	318	374	421	486	540	591 0/1/1	5
259	259	319	375	422	487	541	592 0/1/1	5
260	260	320	376	423	488	542	593 0/1/1	5
261	261	321	377	424	489	543	594 0/1/1	5
262	262	322	378	425	490	544	595 0/1/1	5
263	263	323	379	426	491	545	596 0/1/1	5
264	264	324	380	427	492	546	597 0/1/1	5
265	265	325	381	428	493	547	598 0/1/1	5
266	266	326	382	429	494	548	599 0/1/1	5
267	267	327	383	430	495	549	600 0/1/1	5
268	268	328	384	431	496	550	601 0/1/1	5
269	269	329	385	432	497	551	602 0/1/1	5
270	270	330	386	433	498	552	603 0/1/1	5
271	271	331	387	434	499	553	604 0/1/1	5
272	272	332	388	435	500	554	605 0/1/1	5
273	273	333	389	436	501	555	606 0/1/1	5
274	274	334	390	437	502	556	607 0/1/1	5
275	275	335	391	438	503	557	608 0/1/1	5
276	276	336	392	439	504	558	609 0/1/1	5
277	277	337	393	440	505	559	610 0/1/1	5
278	278	338	394	441	506	560	611 0/1/1	5
279	279	339	395	442	507	561	612 0/1/1	5
280	280	340	396	443	508	562	613 0/1/1	5
281	281	341	397	444	509	563	614 0/1/1	5
282	282	342	398	445	510	564	615 0/1/1	5
283	283	343	399	446	511	565	616 0/1/1	5
284	284	344	400	447	512	566	617 0/1/1	5
285	285	345	401	448	513	567	618 0/1/1	5
286	286	346	402	449	514	568	619 0/1/1	5
287	287	347	403	450	515	569	620 0/1/1	5
288	288	348	404	451	516	570	621 0/1/1	5
289	289	349	405	452	517	571	622 0/1/1	5
290	290	350	406	453	518	572	623 0/1/1	5
291	291	351	407	454	519	573	624 0/1/1	5
292	292	352	408	455	520	574	625 0/1/1	5
293	293	353	409	456	521	575	626 0/1/1	5
294	294	354	410	457	522	576	627 0/1/1	5
295	295	355	411	458	523	577	628 0/1/1	5
296	296	356	412	459	524	578	629 0/1/1	5
297	297	357	413	460	525	579	630 0/1/1	5
298	298	358	414	461	526	580	631 0/1/1	5
299	299	359	415	462	527	581	632 0/1/1	5
300	300	360	416	463	528	582	633 0/1/1	5
301	301	361	417	464	529	583	634 0/1/1	5
302	302	362	418	465	530	584	635 0/1/1	5
303	303	363	419	466	531	585	636 0/1/1	5
304	304	364	420	467	532	586	637 0/1/1	5
305	305	365	421	468	533	587	638 0/1/1	5
306	306	366	422	469	534	588	639 0/1/1	5
307	307	367	423	470	535	589	640 0/1/1	5
308	308	368	424	471	536	590	641 0/	

# Encodings

- An **encoding** is a transformation of pieces of information into another representation for communication or storage
- An encoding is keyless
- An encoding can be public or secret
- The pieces of information need not have a semantic value and can be single letters or symbols

# Ciphers and algorithms

- A **cipher** operates on meaningless components
  - Individual letters
  - Small groups of letters
  - Bits
  - Bytes
- Ciphers are **syntax** related
- Ciphers use **algorithms**
  - with secret (or public) keys as parameters
- Encryption/decryption is the process of applying/reversing a cipher

# Outline

- 1 Codes and ciphers
- 2 Public codes**
- 3 From public codes to ciphers
- 4 Digital ciphers

# Polygraphic versus polyliteral ciphers/encodings

- Polygraphic ciphers/encodings translate a block of letters into another block of letters, numbers or symbols
  - An example is Porta's digraph system
- Polyliteral ciphers/encodings translate a single letter into a (larger, full) block of letters, numbers or symbols
  - Polyliteral ciphers/encodings are nothing more than a simple substitution into an “unknown”, “bigger”, but also “structured” alphabet which can henceforth be fractionated

# Polybius Square

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Figure 1: A simple polyliteral<sup>1</sup>encoding (**Polybius**)

<sup>1</sup>Because we use digits this is also called a dinome substitution

# A rectangular variant

	0	1	2	3	4	5	6	7	8
0		A	B	C	D	E	F	G	H
1	I	J	K	L	M	N	O	P	Q
2	R	S	T	U	V	W	X	Y	Z

Figure 2: A 0-based rectangular encoding for the full alphabet

But note it still uses the legacy A=01 encoding

# The standard legacy encoding

	0	1	2	3	4	5	6	7	8	9
0		A	B	C	D	E	F	G	H	I
1	J	K	L	M	N	O	P	Q	R	S
2	T	U	V	W	X	Y	Z			

Figure 3: A 0-based encoding for the full alphabet, with open space

- This encoding is just the regular legacy encoding translating A, ..., Z to  $01^2$ , ..., 26
- 00, 27, 28 and 29 are available for more symbols if needed

<sup>2</sup>One may or may not remove leading 0s

# The standard modern encoding

	0	1	2	3	4	5	6	7	8	9
0	A	B	C	D	E	F	G	H	I	J
1	K	L	M	N	O	P	Q	R	S	T
2	U	V	W	X	Y	Z				

Figure 4: A 0-based encoding for the full alphabet, with open space

This encoding is just the regular modern encoding  
 translating A, ..., Z to 00, ..., 25

# A table for every base $b$ numeral system (1)

Let us for instance look at base  $b = 3$

	00	01	02	10	11	12	20	21	22
0	□	A	B	C	D	E	F	G	H
1	I	J	K	L	M	N	O	P	Q
2	R	S	T	U	V	W	X	Y	Z

Figure 5: A ternary encoding for the full alphabet including a space

It would have been so nice<sup>3</sup> to build computers based on the (balanced) ternary system instead of the usual binary one...

<sup>3</sup>The Russians tried to do so: <https://en.wikipedia.org/wiki/Setun>

## A table for every base $b$ numeral system (2)

Let us now look at the common binary base  $b = 2$

	000	001	010	011	100	101	110	111
00		A	B	C	D	E	F	G
01	H	I	J	K	L	M	N	O
10	P	Q	R	S	T	U	V	W
11	X	Y	Z					

**Figure 6:** A binary legacy encoding with room for  $2^5 = 32$  symbols

Base32 is a modern variant with added symbols 2, 3, 4, 5, 6, 7

# The Bacon code (steganography)

- Francis Bacon (1561–1626)
- First use a binary code with  $a=0$  and  $b=1$ 
  - In the original we had  $I=J$  and  $U=V$ , coding 24 letters with  $A=aaaaa$ , ...,  $Z=babbb$
  - In modern variants the full alphabet is encoded<sup>4</sup> with  $A=aaaaa$ , ...,  $Z=bbaab$
- SEconDIY HIDE The INDivIDuAL BitS by USiNg GLyPH PROPeRTieS LiKE CoLoR, ITaLIZatIon, SIze, ...

---

<sup>4</sup>Holden's book uses  $A=aaaab$ , ...,  $Z=bbaba$

# The Teletypewriter

- Émile Baudot (1845–1903)
  - Baudot code
  - Paper tape with punched holes
  - 5 positions or bits
- Gilbert Vernam (1890–1960)
  - Secures Baudot code transmission
  - Uses a second (key)tape to be XORed with the plaintext tape
  - Essentially creating a one-time pad

# The wonderfully versatile XOR

- XOR is a binary (bitwise) operation
  - Its nice properties derive from addition modulo 2
  - Modulo 2 subtraction is the same as addition
  - Encryption works by  $c = p \oplus k$
  - and since  $k \oplus k = 0$
  - decryption works by  $p = c \oplus k$
- XOR also has a ternary, quaternary, ... variant
  - Multiple inputs and one output
  - Can be combined in arbitrary trees
    - And with some care even in graphs with loops

# Outline

- 1 Codes and ciphers
- 2 Public codes
- 3 From public codes to ciphers**
- 4 Digital ciphers

# Length tricks

- Nulls
  - Using encoding symbols with no corresponding plaintext
- Straddling (“with a leg on each side”)
  - Use different length encoding strings for different plaintext letters
  - Usually the frequently occurring letters use a smaller length
  - This will result in compression properties

# The straddling checkerboard (1)

	0	1	2	3	4	5	6	7	8	9
				A	B	C	D	E	F	G
1	H	I	J	K	L	M	N	O	P	Q
2	R	S	T	U	V	W	X	Y	Z	

Figure 7: Why are the first three positions blank?

# The straddling checkerboard (2)

	0	1	8	3	4	5	2	9	7	6
				T	R	E	A	S	O	N
0	B	C	D	F	G	H	I	J	K	L
1	M	P	Q	U	V	W	X	Y	Z	.
8	0	1	2	3	4	5	6	7	8	9

Figure 8: A variant that compresses (most occurring letters monome)

Source: slides Hans van der Meer

# The straddling checkerboard (3)

	0	1	8	3	4
	A	E	I	O	U
5	B	C	D	F	G
2	H	K	L	M	N
9	P	Q	R	S	T
7	V	W	X	Y	Z

Figure 9: A variant where the 6 can be used as a null

Source: slides Hans van der Meer

# The straddling checkerboard (4)

	0	1	8	3	4	5
2	A	B	C	D	E	F
9	G	H	I	J	K	L
7	M	N	O	P	Q	R
62	S	T	U	V	W	X
67	Y	Z	0	1	2	3
69	4	5	6	7	8	9

Figure 10: A dinome-trinome variant

Source: slides Hans van der Meer

# The straddling checkerboard (5)

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

Figure 11: Lots of homophones

Source: slides Hans van der Meer

# Cryptanalysis of straddling checkerboards

- Identify dinome coordinates
  - They occur more frequently
  - They have lots of different contacts
  - Look at repetition of four or more identical digits
  - Look at patterns like abab
- Solve the resulting monoalphabetic substitution
- And possibly identify the key used

# Fractionation after polyliteral encoding

- After having encoded letters one may consider subunits of polyliterals
  - In the binary case those subunits could be bits
- More substitutions and especially transpositions can be executed
  - That is what classic and modern block ciphers like DES and AES do
- The resulting new subunits might be assembled again into polyliterals
  - Which can then possibly be translated back to the original alphabet

# Fractionating system example: ADFGVX (1)

	A	D	F	G	V	X
A	b	5	x	q	j	c
D	6	y	r	k	d	7
F	z	s	l	e	8	1
G	t	m	f	9	2	u
V	n	g	0	3	v	o
X	h	a	4	w	p	i

Figure 12: ADFGVX 6-by-6 square

## Fractionating system example: ADFGVX (2)

- First use the polyliteral ADFGVX square
- Then use a keyed columnar transposition
- Example encryption with keyword GANDHI and square filled as in previous slide
  - AGGAV AXGDA DFGGA FXFFV  
VXXFG XXVGF VAAXX ADAXG  
FFFFVD
- Exercise: decode this message

# Outline

- 1 Codes and ciphers
- 2 Public codes
- 3 From public codes to ciphers
- 4 Digital ciphers**

# Shannon's theory (1)

- **Confusion**

- Each ciphertext bit has complex (nonlinear) relations with the plaintext and key bits
- Mostly done by substitutions

- **Diffusion**

- Each plaintext or key bit affects many bits of the ciphertext
- Mostly done by transpositions

## Shannon's theory (2)

- **Mixing** transformation (function)  $H$ 
  - Non-secret, confusing and diffusing transformation
  - A transposition ( $T$ ), followed by an alternation of linear ( $L$ ) maps and substitutions ( $S$ )
  - $H = L \circ S \circ L \circ S \circ L \circ T$
  - Both  $T$  and  $L$  operate on full blocks of letters
  - $S$  operates componentwise, on each individual letter

# Shannon's theory (3)

- Shannon's cipher construction
  - Uses one, two or even more mixing transformations
    - For two mixings this is  $C = T_k \circ H_2 \circ S_j \circ H_1 \circ R_i$
    - $i, j, k$  is keying material for simple ciphers  $R, S, T$
    - Here secret keys enter the scene by adding more confusion, typically through the simple substitutions  $R, S$  and  $T$

# SP-networks

- **SP-networks** resemble Shannon's construction
  - Works with bits instead of larger alphabets
- Uses **large diffusing transpositions** of bits
- Uses **smaller confusing polygraphic substitutions** of sequences of bits (bytes, nibbles, ...)
- Alternates these in a number of rounds
- Mixes in (parts of) the key at the start of each round
  - Mixing uses simple XORs
  - Also at the end the key is once more mixed in

# Feistel networks (building block)

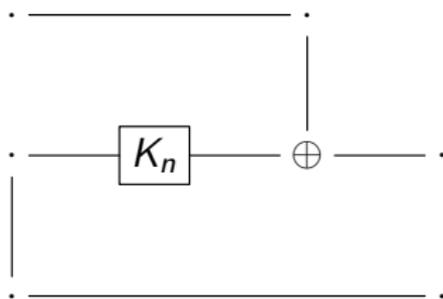


Figure 13: Building block (also used upside down)

## Feistel networks (first few steps)

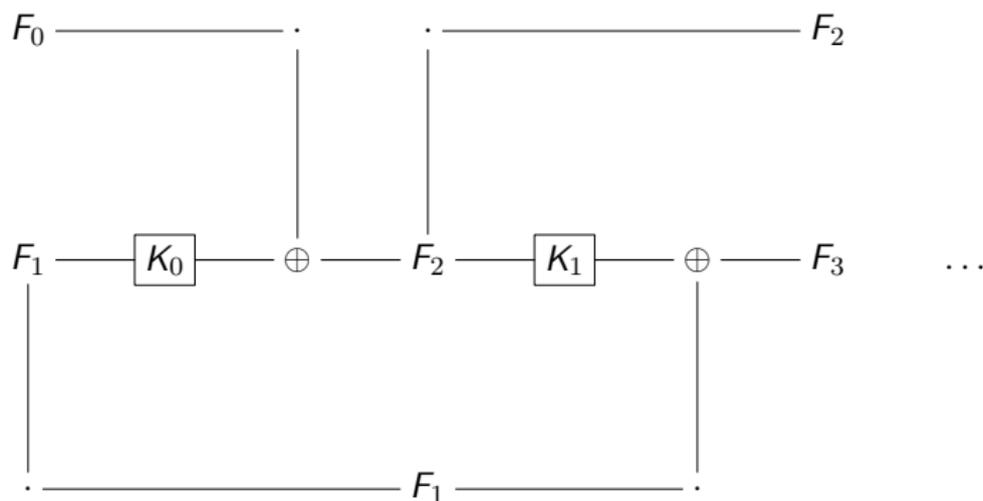


Figure 14:  $F_2 = \mathcal{F}(K_0, F_1) \oplus F_0$ ;  $F_3 = \mathcal{F}(K_1, F_2) \oplus F_1$

## Feistel network encryption sequence

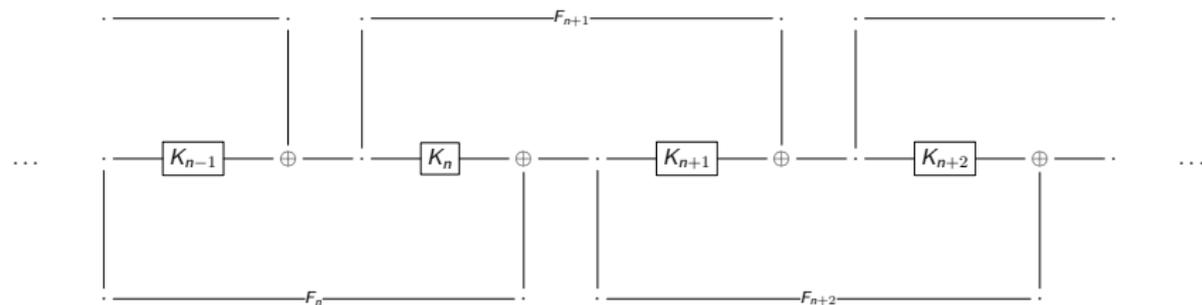


Figure 15:  $F_{n+2} = \mathcal{F}(K_n, F_{n+1}) \oplus F_n$

# Simpler Feistel network building block

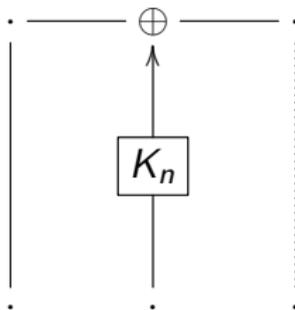


Figure 16: Building block (also used upside down)

## Simpler Feistel network first steps

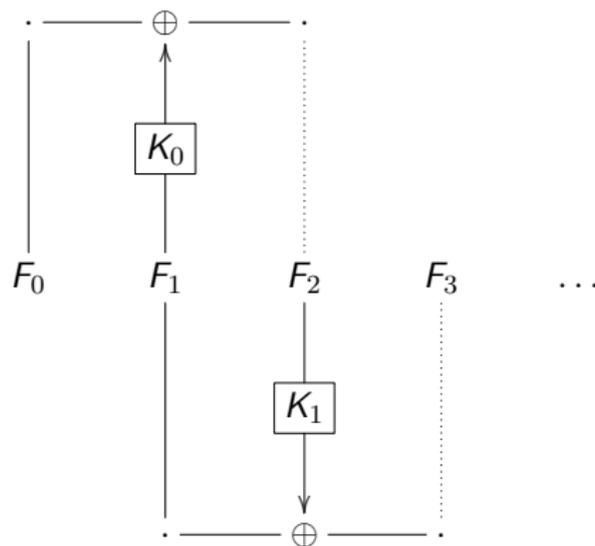


Figure 17:  $F_2 = \mathcal{F}(K_0, F_1) \oplus F_0$ ;  $F_3 = \mathcal{F}(K_1, F_2) \oplus F_1$

## Simpler Feistel network encryption sequence

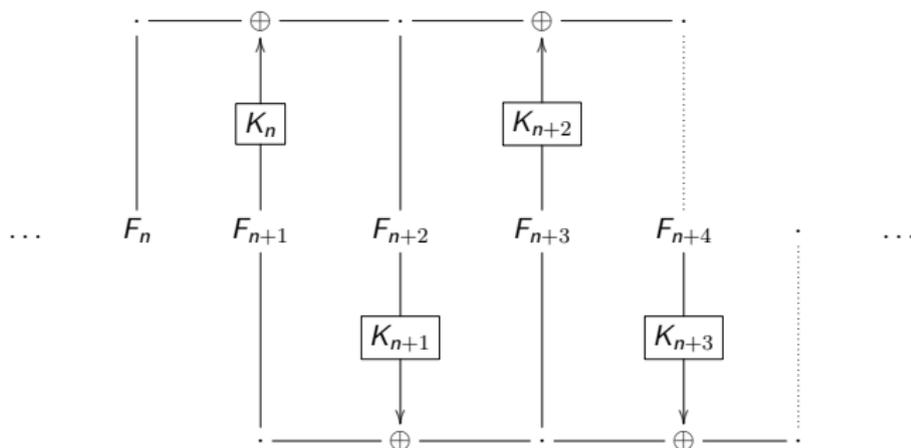


Figure 18:  $F_{n+2} = \mathcal{F}(K_n, F_{n+1}) \oplus F_n$