LumiNUS Cryptanalysis Challenge on Substitution Cipher

- The substitution table used:

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	_
t	g	a	c	s	o	n	k	_	z	i	v	m	p	d	u	j	y	e	b	l	x	w	f	q	r	h

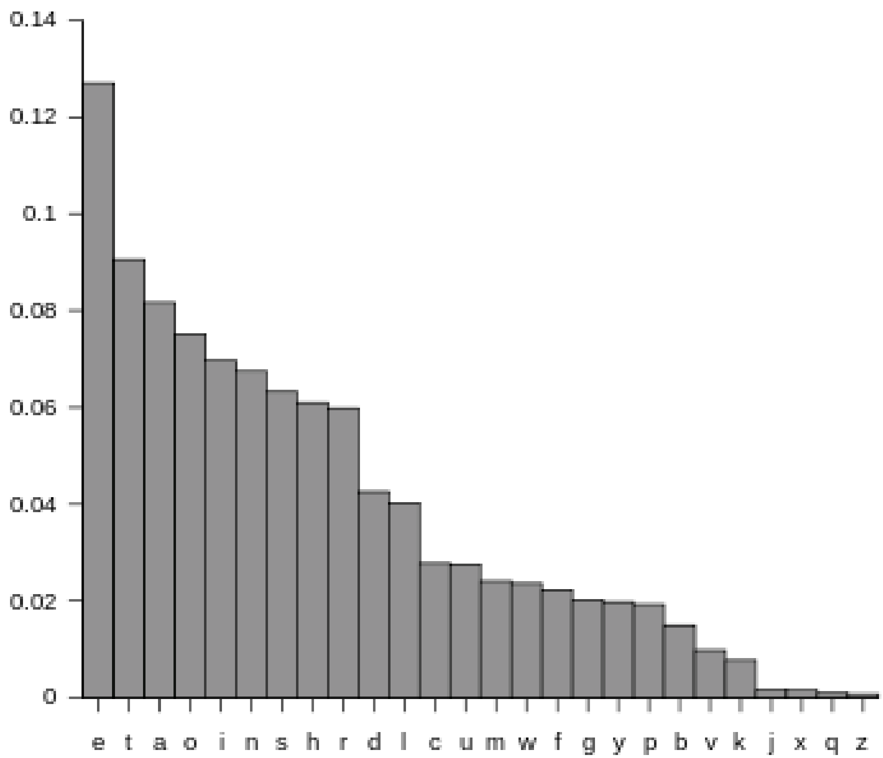
- The plaintext:

cryptanalysis_is_the_study_of_analyzing_information_systems_in_order_to_study_the_hidden_aspects_of_the_systems_given_some_encrypted_data_or_ciphertext_the_goal_of_the_cryptanalyst_is_to_gain_as_much_information_as_possible_about_the_original_unencrypted_data_or_plaintext ...

...they_are_viewed_as_two_sides_of_the_same_coin_secure_cryptography_requires_design_against_possible_cryptanalysis_governments_have_long_recognized_the_potential_benefits_of_cryptanalysis_for_intelligence_both_military_and_diplomatic_and_they_established_dedicated_organizations_devoted_to_breaking_the_codes_and_ciphers_of_other_nations

Some Useful Heuristics

- The most-frequently occurring characters in ciphertext?
  - h (333×) ← _ (space) , s (215×) ← e, b (206x) ← t, t (157×) ← a:
    match the 3 non-space most-frequently occurring characters in English
- Spaces must break up the sentence reasonably well



From http://en.wikipedia.org/wiki/Letter_frequency

Some Useful Heuristics

- Single-letter words:
  - **a**: an indefinite article
  - i: the first (singular) person
- Digraphs:
  - **to**, **is**, **in**, **do**, **on**, **of**, **or**, it, at, an, he, …
- Trigraphs:
  - **the**, **and**, **for**, **has**, **not**, **can**, one, get, man, …
- More and more *guessable words*

Testing Rounds: After Some Character Mappings

Based on the *top 1+3 most-frequently occurring single characters*, and some common English *digraphs* & *trigraphs*:

```
rio@ubuntu-20-04:~/CS2107$ tr 'hsbtk' ' ETAH' < ciphertext.txt
ayquTApAvqe_e _e THE eTlcq do ApAvqe_pn _podymAT_dp eqeTEme _p dycEy Td eTlcq THE H_ccEp AeuEaTe
do THE eqeTEme n_xEp edmE EpayquTEc cATA dy a_uHEyTEfT THE ndAv do THE ayquTApAvqeT _e Td nA_p Ae
 mlaH _podymAT_dp Ae udee_gvE AgdlT THE dy_n_pAv lpEpayquTEc cATA dy uvA_pTEfT ayquTApAvqe_e _e l
eEc Td gyEAaH ayquTdnyAuH_a eEaly_Tq eqeTEme Apc nA_p AaaEee Td THE adpTEpTe do EpayquTEc mEeeAnE
e ExEp _o THE ayquTdnyAuH_a iEq _e lpipdwp _p Acc_T_dp Td mATHEmAT_aAv ApAvqe_e do ayquTdnyAuH_a
Avndy_THme ayquTApAvqe_e _pavlcEe THE eTlcq do e_cE aHAppEv ATTAaie THAT cd pdT TAynET wEAipEeeEe
 _p THE ayquTdnyAuH_a Avndy_THme THEmeEvxEe glT _peTEAc Efuvd_T wEAipEeeEe _p THE_y _muvEmEpTAT_d
p ATTAaie aAp gE avAee_o_Ec gAeEc dp wHAT TquE do _podymAT_dp THE ATTAaiEy HAe AxA_vAgvE _p a_uHE
yTEfT dpvq THE ayquTApAvqeT HAe AaaEee dpvq Td A advvEaT_dp do a_uHEyTEfTe dy adcETEfTe _p ipdwp
uvA_pTEfT THE ATTAaiEy HAe A eET do a_uHEyTEfTe Td wH_aH THEq ipdw THE adyyEeudpc_pn uvA_pTEfT AT
TAaie aAp Aved gE aHAyAaTEy_eEc gq THE yEedlyaEe THEq yEjl_yE THdeE yEedlyaEe _pavlcE THE odvvdw_
pn T_mE _e THE plmgEy do admulTAT_dp eTEue wH_aH mleT gE uEyodymEc mEmdyq _e THE AmdlpT do eTdyAn
E yEjl_yEc Td uEyodym THE ATTAai cATA _e THE jlApT_Tq Apc TquE do uvA_pTEfTe Apc a_uHEyTEfTe yEjl
_yEc ody A uAyT_alvAy AuuydAaH THE yEelvTe do ayquTApAvqe_e aAp Aved xAyq _p leEolvpEee _p TdTAv
gyEAi THE ATTAaiEy cEclaEe THE eEayET iEq _p nvdgAv cEclaT_dp THE ATTAaiEy c_eadxEye A olpaT_dpAv
vq Ejl_xAvEpT Avndy_THm ody EpayquT_dp Apc cEayquT_dp glT w_THdlT vEAyp_pn THE iEq _p vdaAv cEcla
T_dp THE ATTAaiEy c_eadxEye Acc_T_dpAv uvA_pTEfTe dy a_uHEyTEfTe pdT uyEx_dlevq ipdwp ayquTApAvqe
_e HAe adExdvxEc TdnETHEy w_TH ayquTdnyAuHq Apc THE adpTEeT aAp gE TyAaEc THydlnH THE H_eTdyq do
ayquTdnyAuHq pEw a_uHEye gE_pn cEe_npEc Td yEuvAaE dvc gydiEp cEe_npe Apc pEw ayquTApAvqT_a TEaHp
_jlEe _pxEpTEc Td ayAai THE _muydxEc eaHEmEe _p uyAaT_aE THEq AyE x_EwEc Ae Twd e_cEe do THE eAmE
 ad_p eEalyE ayquTdnyAuHq yEjl_yEe cEe_np AnA_peT udee_gvE ayquTApAvqe_e ndxEypmEpTe HAxE vdpn yE
adnp_rEc THE udTEpT_Av gEpEo_Te do ayquTApAvqe_e ody _pTEvv_nEpaE gdTH m_v_TAyq Apc c_uvdmAT_a Ap
c THEq EeTAgv_eHEc cEc_aATEc dynAp_rAT_dpe cExdTEc Td gyEAi_pn THE adcEe Apc a_uHEye do dTHEy pAT
_dpe
```

Testing Rounds: After Some More Character Mappings

After a few more *guessable characters* in their respective words:

```
rio@ubuntu-20-04:~/CS2107$ tr 'hsbtkm_ec' ' ETAHMISD' < ciphertext.txt
ayquTApAvqSIS IS THE STlDq do ApAvqrIpn IpodyMATIdp SqSTEMS Ip dyDEy Td STlDq THE HIDDEp ASuEaTS
do THE SqSTEMS nIxEp SdME EpayquTED DATA dy aIuHEyTEfT THE ndAv do THE ayquTApAvqST IS Td nAIp AS
 MlaH IpodyMATIdp AS udSSIgvE AgdlT THE dyInIpAv lpEpayquTED DATA dy uvAIpTEfT ayquTApAvqSIS IS l
SED Td gyEAaH ayquTdnyAuHIa SEalyITq SqSTEMS ApD nAIp AaaESS Td THE adpTEpTS do EpayquTED MESSAnE
S ExEp Io THE ayquTdnyAuHIa iEq IS lpipdwp Ip ADDITIdp Td MATHEMATIaAv ApAvqSIS do ayquTdnyAuHIa
AvndyITHMS ayquTApAvqSIS IpavlDES THE STlDq do SIDE aHAppEv ATTAaiS THAT Dd pdT TAynET wEAipESSES
 Ip THE ayquTdnyAuHIa AvndyITHMS THEMSEvxES glT IpSTEAD EfuvdIT wEAipESSES Ip THEIy IMuvEMEpTATId
p ATTAaiS aAp gE avASSIoIED gASED dp wHAT TquE do IpodyMATIdp THE ATTAaiEy HAS AxAIvAgvE Ip aIuHE
yTEfT dpvq THE ayquTApAvqST HAS AaaESS dpvq Td A advvEaTIdp do aIuHEyTEfTS dy adDETEfTS Ip ipdwp
uvAIpTEfT THE ATTAaiEy HAS A SET do aIuHEyTEfTS Td wHIaH THEq ipdw THE adyyESudpDIpn uvAIpTEfT AT
TAaiS aAp AvSd gE aHAyAaTEyISED gq THE yESdlyaES THEq yEjlIyE THdSE yESdlyaES IpavlDE THE odvvdwI
pn TIME IS THE plMgEy do adMulTATIdp STEuS wHIaH MlST gE uEyodyMED MEMdyq IS THE AMdlpT do STdyAn
E yEjlIyED Td uEyodyM THE ATTAai DATA IS THE jlApTITq ApD TquE do uvAIpTEfTS ApD aIuHEyTEfTS yEjl
IyED ody A uAyTIalvAy AuuydAaH THE yESlvTS do ayquTApAvqSIS aAp AvSd xAyq Ip lSEolvpESS Ip TdTAv
gyEAi THE ATTAaiEy DEDlaES THE SEayET iEq Ip nvdgAv DEDlaTIdp THE ATTAaiEy DISadxEyS A olpaTIdpAv
vq EjlIxAvEpT AvndyITHM ody EpayquTIdp ApD DEayquTIdp glT wITHdlT vEAypIpn THE iEq Ip vdaAv DEDla
TIdp THE ATTAaiEy DISadxEyS ADDITIdpAv uvAIpTEfTS dy aIuHEyTEfTS pdT uyExIdlSvq ipdwp ayquTApAvqS
IS HAS adExdvxED TdnETHEy wITH ayquTdnyAuHq ApD THE adpTEST aAp gE TyAaED THydlnH THE HISTdyq do
ayquTdnyAuHq pEw aIuHEyS gEIpn DESInpED Td yEuvAaE dvD gydiEp DESInpS ApD pEw ayquTApAvqTIa TEaHp
IjlES IpxEpTED Td ayAai THE IMuydxED SaHEMES Ip uyAaTIaE THEq AyE xIEwED AS Twd SIDES do THE SAME
 adIp SEalyE ayquTdnyAuHq yEjlIyES DESInp AnAIpST udSSIgvE ayquTApAvqSIS ndxEypMEpTS HAxE vdpn yE
adnpIrED THE udTEpTIAv gEpEoITS do ayquTApAvqSIS ody IpTEvvInEpaE gdTH MIvITAyq ApD DIuvdMATIa Ap
D THEq ESTAgvISHED DEDIaATED dynApIrATIdpS DExdTED Td gyEAiIpn THE adDES ApD aIuHEyS do dTHEy pAT
IdpS
```

Testing Rounds: After Some More Character Mappings

After a few more *guessable characters* in their respective words:

```
rio@ubuntu-20-04:~/CS2107$ tr 'hsbtkm_ecqpvxna' ' ETAHMISDYNLVGC' < ciphertext.txt
CyYuTANALYSIS IS THE STlDY do ANALYrING INodyMATIdN SYSTEMS IN dyDEy Td STlDY THE HIDDEN ASuECTS
do THE SYSTEMS GIVEN SdME ENCyYuTED DATA dy CIuHEyTEfT THE GdAL do THE CyYuTANALYST IS Td GAIN AS
 MlCH INodyMATIdN AS udSSIgLE AgdlT THE dyIGINAL lNENCyYuTED DATA dy uLAINTEfT CyYuTANALYSIS IS l
SED Td gyEACH CyYuTdGyAuHIC SEClyITY SYSTEMS AND GAIN ACCESS Td THE CdNTENTS do ENCyYuTED MESSAGE
S EVEN Io THE CyYuTdGyAuHIC iEY IS lNiNdwN IN ADDITIdN Td MATHEMATICAL ANALYSIS do CyYuTdGyAuHIC
ALGdyITHMS CyYuTANALYSIS INCLlDES THE STlDY do SIDE CHANNEL ATTACiS THAT Dd NdT TAyGET wEAiNESSES
 IN THE CyYuTdGyAuHIC ALGdyITHMS THEMSELVES glT INSTEAD EfuLdIT wEAiNESSES IN THEIy IMuLEMENTATId
N ATTACiS CAN gE CLASSIoIED gASED dN wHAT TYuE do INodyMATIdN THE ATTACiEy HAS AVAILAgLE IN CIuHE
yTEfT dNLY THE CyYuTANALYST HAS ACCESS dNLY Td A CdLLECTIdN do CIuHEyTEfTS dy CdDETEfTS IN iNdwN
uLAINTEfT THE ATTACiEy HAS A SET do CIuHEyTEfTS Td wHICH THEY iNdw THE CdyyESudNDING uLAINTEfT AT
TACiS CAN ALSd gE CHAyACTEyISED gY THE yESdlyCES THEY yEjlIyE THdSE yESdlyCES INCLlDE THE odLLdwI
NG TIME IS THE NlMgEy do CdMulTATIdN STEuS wHICH MlST gE uEyodyMED MEMdyY IS THE AMdlNT do STdyAG
E yEjlIyED Td uEyodyM THE ATTACi DATA IS THE jlANTITY AND TYuE do uLAINTEfTS AND CIuHEyTEfTS yEjl
IyED ody A uAyTIClLAy AuuydACH THE yESlLTS do CyYuTANALYSIS CAN ALSd VAyY IN lSEolLNESS IN TdTAL
gyEAi THE ATTACiEy DEDlCES THE SECyET iEY IN GLdgAL DEDlCTIdN THE ATTACiEy DISCdVEyS A olNCTIdNAL
LY EjlIVALENT ALGdyITHM ody ENCyYuTIdN AND DECyYuTIdN glT wITHdlT LEAyNING THE iEY IN LdCAL DEDlC
TIdN THE ATTACiEy DISCdVEyS ADDITIdNAL uLAINTEfTS dy CIuHEyTEfTS NdT uyEVIdlSLY iNdwN CyYuTANALYS
IS HAS CdEVdLVED TdGETHEy wITH CyYuTdGyAuHY AND THE CdNTEST CAN gE TyACED THydlGH THE HISTdyY do
CyYuTdGyAuHY NEw CIuHEyS gEING DESIGNED Td yEuLACE dLD gydiEN DESIGNS AND NEw CyYuTANALYTIC TECHN
IjlES INVENTED Td CyACi THE IMuydVED SCHEMES IN uyACTICE THEY AyE VIEwED AS Twd SIDES do THE SAME
 CdIN SEClyE CyYuTdGyAuHY yEjlIyES DESIGN AGAINST udSSIgLE CyYuTANALYSIS GdVEyNMENTS HAVE LdNG yE
CdGNIrED THE udTENTIAL gENEoITS do CyYuTANALYSIS ody INTELLIGENCE gdTH MILITAyY AND DIuLdMATIC AN
D THEY ESTAgLISHED DEDICATED dyGANIrATIdNS DEVdTED Td gyEAiING THE CdDES AND CIuHEyS do dTHEy NAT
IdNS
```

The partial plaintext already looks pretty readable and crackable!

Testing Rounds: After Some More Character Mappings

After a few more *guessable characters* in their respective words:

```
rio@ubuntu-20-04:~/CS2107$ tr 'hsbtkm_ecqpvxnaluyd' ' ETAHMISDYNLVGCUPRO' < ciphertext.txt
CRYPTANALYSIS IS THE STUDY Oo ANALYrING INoORMATION SYSTEMS IN ORDER TO STUDY THE HIDDEN ASPECTS
Oo THE SYSTEMS GIVEN SOME ENCRYPTED DATA OR CIPHERTEfT THE GOAL Oo THE CRYPTANALYST IS TO GAIN AS
 MUCH INoORMATION AS POSSIgLE AgOUT THE ORIGINAL UNENCRYPTED DATA OR PLAINTEfT CRYPTANALYSIS IS U
SED TO gREACH CRYPTOGRAPHIC SECURITY SYSTEMS AND GAIN ACCESS TO THE CONTENTS Oo ENCRYPTED MESSAGE
S EVEN Io THE CRYPTOGRAPHIC iEY IS UNiNOwN IN ADDITION TO MATHEMATICAL ANALYSIS Oo CRYPTOGRAPHIC
ALGORITHMS CRYPTANALYSIS INCLUDES THE STUDY Oo SIDE CHANNEL ATTACiS THAT DO NOT TARGET wEAiNESSES
 IN THE CRYPTOGRAPHIC ALGORITHMS THEMSELVES gUT INSTEAD EfPLOIT wEAiNESSES IN THEIR IMPLEMENTATIO
N ATTACiS CAN gE CLASSIoIED gASED ON wHAT TYPE Oo INoORMATION THE ATTACiER HAS AVAILAgLE IN CIPHE
RTEfT ONLY THE CRYPTANALYST HAS ACCESS ONLY TO A COLLECTION Oo CIPHERTEfTS OR CODETEfTS IN iNOwN
PLAINTEfT THE ATTACiER HAS A SET Oo CIPHERTEfTS TO wHICH THEY iNOw THE CORRESPONDING PLAINTEfT AT
TACiS CAN ALSO gE CHARACTERISED gY THE RESOURCES THEY REjUIRE THOSE RESOURCES INCLUDE THE oOLLOwI
NG TIME IS THE NUMgER Oo COMPUTATION STEPS wHICH MUST gE PERoORMED MEMORY IS THE AMOUNT Oo STORAG
E REjUIRED TO PERoORM THE ATTACi DATA IS THE jUANTITY AND TYPE Oo PLAINTEfTS AND CIPHERTEfTS REjU
IRED oOR A PARTICULAR APPROACH THE RESULTS Oo CRYPTANALYSIS CAN ALSO VARY IN USEoULNESS IN TOTAL
gREAi THE ATTACiER DEDUCES THE SECRET iEY IN GLOgAL DEDUCTION THE ATTACiER DISCOVERS A oUNCTIONAL
LY EjUIVALENT ALGORITHM oOR ENCRYPTION AND DECRYPTION gUT wITHOUT LEARNING THE iEY IN LOCAL DEDUC
TION THE ATTACiER DISCOVERS ADDITIONAL PLAINTEfTS OR CIPHERTEfTS NOT PREVIOUSLY iNOwN CRYPTANALYS
IS HAS COEVOLVED TOGETHER wITH CRYPTOGRAPHY AND THE CONTEST CAN gE TRACED THROUGH THE HISTORY Oo
CRYPTOGRAPHY NEw CIPHERS gEING DESIGNED TO REPLACE OLD gROiEN DESIGNS AND NEw CRYPTANALYTIC TECHN
IjUES INVENTED TO CRACi THE IMPROVED SCHEMES IN PRACTICE THEY ARE VIEwED AS TwO SIDES Oo THE SAME
 COIN SECURE CRYPTOGRAPHY REjUIRES DESIGN AGAINST POSSIgLE CRYPTANALYSIS GOVERNMENTS HAVE LONG RE
COGNIrED THE POTENTIAL gENEoITS Oo CRYPTANALYSIS oOR INTELLIGENCE gOTH MILITARY AND DIPLOMATIC AN
D THEY ESTAgLISHED DEDICATED ORGANIrATIONS DEVOTED TO gREAiING THE CODES AND CIPHERS Oo OTHER NAT
IONS
```

We are almost done!

Testing Rounds: Completed Mapping

The complete plaintext:

```
rio@ubuntu-20-04:~/CS2107$ tr 'hsbtkm_ecqpvxnaluydorfgiwj' ' ETAHMISDYNLVGCUPROFZXBKWQ' < ciphert
ext.txt
CRYPTANALYSIS IS THE STUDY OF ANALYZING INFORMATION SYSTEMS IN ORDER TO STUDY THE HIDDEN ASPECTS
OF THE SYSTEMS GIVEN SOME ENCRYPTED DATA OR CIPHERTEXT THE GOAL OF THE CRYPTANALYST IS TO GAIN AS
 MUCH INFORMATION AS POSSIBLE ABOUT THE ORIGINAL UNENCRYPTED DATA OR PLAINTEXT CRYPTANALYSIS IS U
SED TO BREACH CRYPTOGRAPHIC SECURITY SYSTEMS AND GAIN ACCESS TO THE CONTENTS OF ENCRYPTED MESSAGE
S EVEN IF THE CRYPTOGRAPHIC KEY IS UNKNOWN IN ADDITION TO MATHEMATICAL ANALYSIS OF CRYPTOGRAPHIC
ALGORITHMS CRYPTANALYSIS INCLUDES THE STUDY OF SIDE CHANNEL ATTACKS THAT DO NOT TARGET WEAKNESSES
 IN THE CRYPTOGRAPHIC ALGORITHMS THEMSELVES BUT INSTEAD EXPLOIT WEAKNESSES IN THEIR IMPLEMENTATIO
N ATTACKS CAN BE CLASSIFIED BASED ON WHAT TYPE OF INFORMATION THE ATTACKER HAS AVAILABLE IN CIPHE
RTEXT ONLY THE CRYPTANALYST HAS ACCESS ONLY TO A COLLECTION OF CIPHERTEXTS OR CODETEXTS IN KNOWN
PLAINTEXT THE ATTACKER HAS A SET OF CIPHERTEXTS TO WHICH THEY KNOW THE CORRESPONDING PLAINTEXT AT
TACKS CAN ALSO BE CHARACTERISED BY THE RESOURCES THEY REQUIRE THOSE RESOURCES INCLUDE THE FOLLOWI
NG TIME IS THE NUMBER OF COMPUTATION STEPS WHICH MUST BE PERFORMED MEMORY IS THE AMOUNT OF STORAG
E REQUIRED TO PERFORM THE ATTACK DATA IS THE QUANTITY AND TYPE OF PLAINTEXTS AND CIPHERTEXTS REQU
IRED FOR A PARTICULAR APPROACH THE RESULTS OF CRYPTANALYSIS CAN ALSO VARY IN USEFULNESS IN TOTAL
BREAK THE ATTACKER DEDUCES THE SECRET KEY IN GLOBAL DEDUCTION THE ATTACKER DISCOVERS A FUNCTIONAL
LY EQUIVALENT ALGORITHM FOR ENCRYPTION AND DECRYPTION BUT WITHOUT LEARNING THE KEY IN LOCAL DEDUC
TION THE ATTACKER DISCOVERS ADDITIONAL PLAINTEXTS OR CIPHERTEXTS NOT PREVIOUSLY KNOWN CRYPTANALYS
IS HAS COEVOLVED TOGETHER WITH CRYPTOGRAPHY AND THE CONTEST CAN BE TRACED THROUGH THE HISTORY OF
CRYPTOGRAPHY NEW CIPHERS BEING DESIGNED TO REPLACE OLD BROKEN DESIGNS AND NEW CRYPTANALYTIC TECHN
IQUES INVENTED TO CRACK THE IMPROVED SCHEMES IN PRACTICE THEY ARE VIEWED AS TWO SIDES OF THE SAME
 COIN SECURE CRYPTOGRAPHY REQUIRES DESIGN AGAINST POSSIBLE CRYPTANALYSIS GOVERNMENTS HAVE LONG RE
COGNIZED THE POTENTIAL BENEFITS OF CRYPTANALYSIS FOR INTELLIGENCE BOTH MILITARY AND DIPLOMATIC AN
D THEY ESTABLISHED DEDICATED ORGANIZATIONS DEVOTED TO BREAKING THE CODES AND CIPHERS OF OTHER NAT
IONS
```

Substitution Cipher: Review (*Again*)

Some terms:

- The ***key space***: the set of all possible keys
- The ***key space size***: the total number of possible keys
- The ***key size*** or ***key length***: the number of bits required to represent a particular key

- For substitution cipher:
  - The key space?
  - The key space size: 27!
  - The key size: at least $\log_2(27!) \approx 94$ bits

Showing the Lower Bound of the Key Size/Length

- The lower bound of **key size/length**: $\log_2(27!) \approx$ **94 bits**
- A few possible **key representations**:
  - **1 byte** per symbol/character: 27 * 1 byte = 27 bytes = **216** bits
  - **5 bits** per symbol/character: 27 * 5 bits = **135** bits
- How to show that 94 bits is the **lower bound**?
  - Show that using 94 bits is **possible** to represent all keys
  - Show that using <94 bits is **not possible** to represent all keys
- *So, how to show these??*

1.5 Modern Ciphers: Block Ciphers

1.5.1 Block cipher definition

1.5.2 Popular block ciphers

1.5.3 Properties of block ciphers

1.5.4 Block cipher modes-of-operation

1.5.5 Examples of attacks on block ciphers

1.5.1 Block Cipher Definition

Illustration of a Stream Cipher

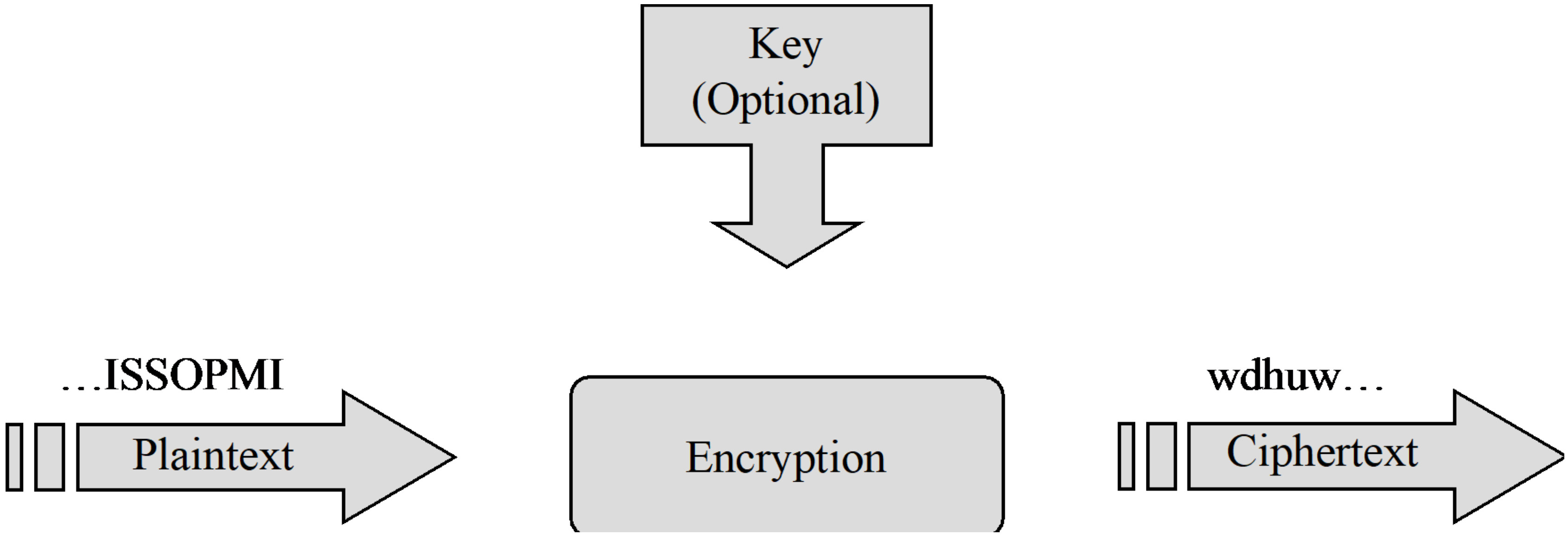
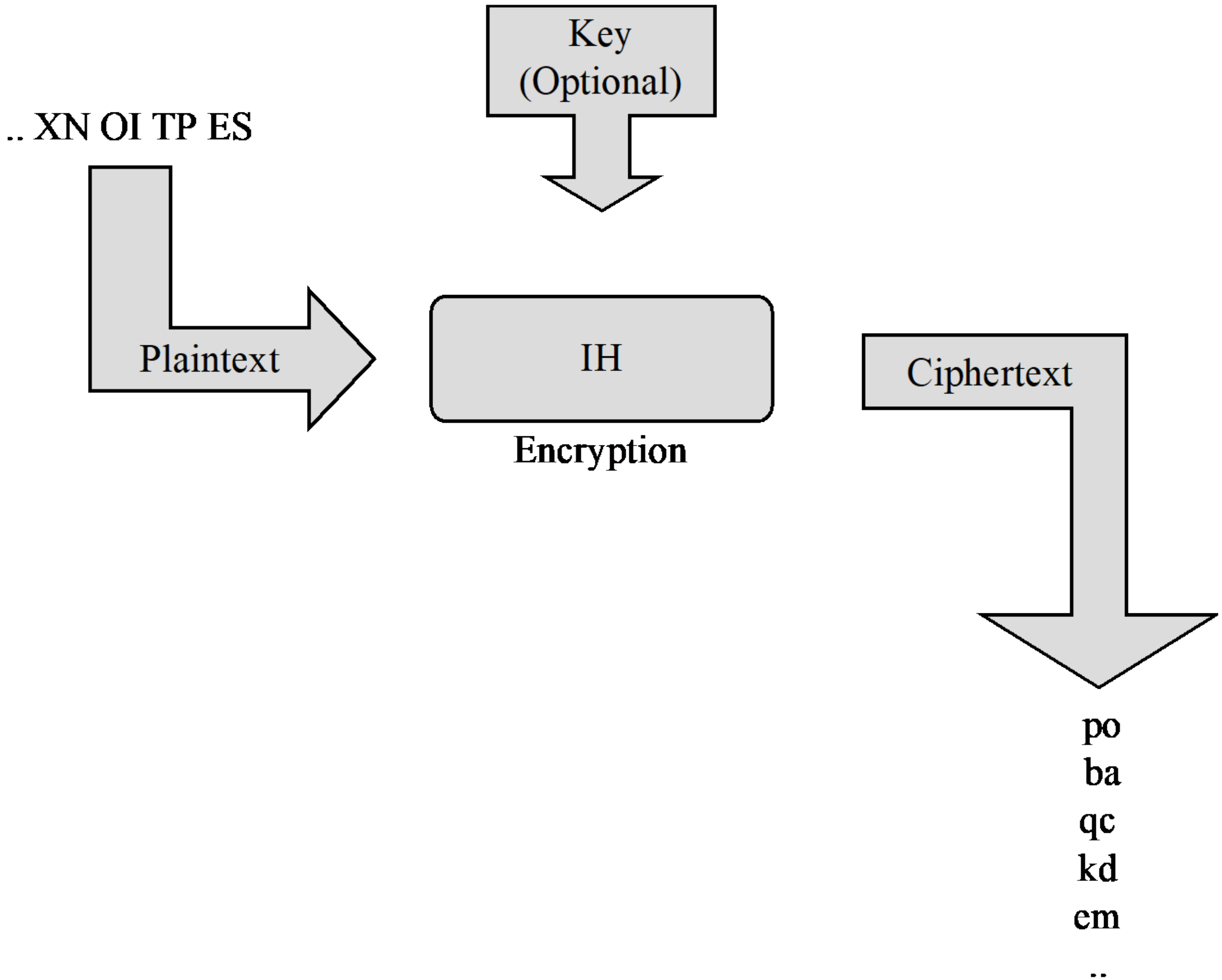
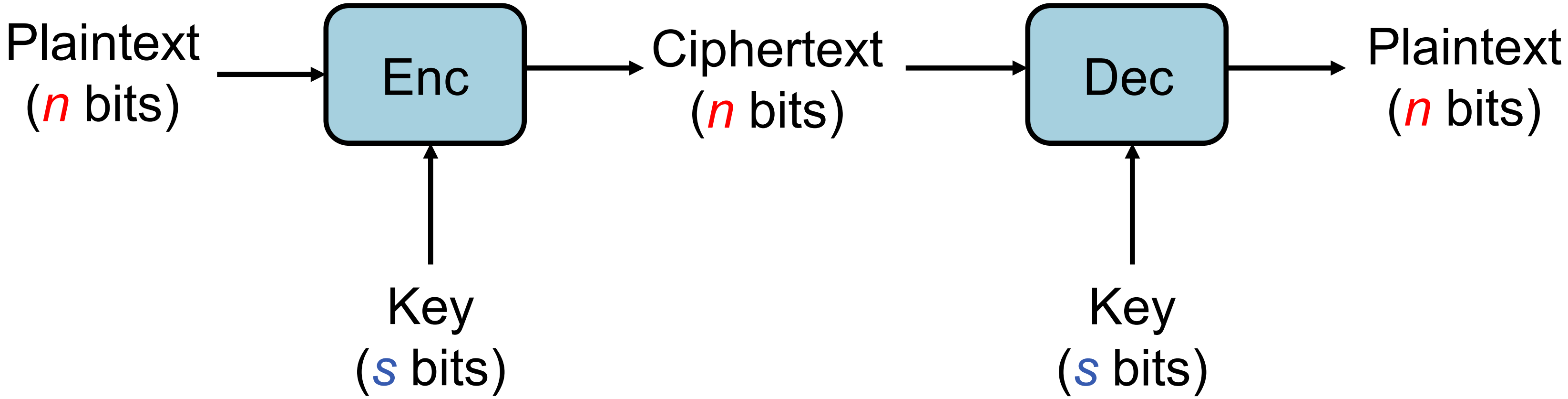


Illustration of a Block Cipher



A Block Cipher: *Block Size* and *Key Size*

- Block cipher is an important **crypto primitive**:
  used in several schemes/protocols for various purposes
- Recall again:
  E (Enc): $\boldsymbol{K} \times \boldsymbol{M} \rightarrow \boldsymbol{C}$ and **D (Dec)**: $\boldsymbol{K} \times \boldsymbol{C} \rightarrow \boldsymbol{M}$
- $\boldsymbol{M} = \boldsymbol{C} = \{0,1\}^n$, with $\boldsymbol{n}$ **= block size**
- $\boldsymbol{K}$ (key space) $= \{0,1\}^s$, with $\boldsymbol{s}$ **= key size/length**



A Block Cipher: *Block Size* and *Key Size*

- Some popular block ciphers with their **block & key sizes**:
  - **DES** : n = 64 bits, s= 56 bits
  - **3DES** : n = 64 bits, s = (up to) 168 bits
    (but the effective security is lower, *see later slides*)
  - **AES** : n = 128 bits, s = 128, 192, 256 bits
- **The longer the key** is:
  - The more secure the scheme is
  - The slower it is
- Question: Can the **block size** be **too small** (i.e. <64 bits)?
  See Tutorial 2 for a possible attack

A Block Cipher: A Mathematical Model (Formalism)

Optional

- Recall again the 3 algorithms of a cipher: **G, E, D**
- G (key-generation algorithm): just generates $k \in \boldsymbol{K}$
- Any other requirements for E: $\boldsymbol{K} \times \boldsymbol{M} \rightarrow \boldsymbol{C}$ and D: $\boldsymbol{K} \times \boldsymbol{C} \rightarrow \boldsymbol{M}$?

- Need to abstract **what a block cipher really does**: a *mathematical model* of a block cipher
- **(Keyed) pseudorandom permutation (PRP)**: E: K × X → X , s.t:
  - [There exists an **efficient deterministic** algorithm to evaluate E(k,x)]
  - The output "**looks random**": indistinguishable from a random function
  - The function E is **bijective (1-to-1)**, and thus is **length preserving**
  - There exists an **efficient** inversion algorithm D(*k*,*y*), which thus satisfies the **correctness requirement**: for all $m \in \boldsymbol{M}$ and $k \in \boldsymbol{K}$, $D_k(E_k(m)) = m$

A Block Cipher: Pseudorandom Permutation

Optional

- Don't confuse **pseudorandom permutation (PRP)** with both:
  - Pseudorandom generator (**PRG**):
    takes a short random seed and outputs a long pseudorandom sequence
  - **Permutation cipher**: a cipher using letter-index permutation operation

- In general, ***permutation*** of a set: a rearrangement of its elements
- A "***permutation function***" (see also https://en.wikipedia.org/wiki/Permutation):
  - Performs a **rearrangement of a set**: a bijection from a set onto itself
  - An example:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 5 & 4 & 3 & 1 \end{pmatrix}$$

A Block Cipher: Pseudorandom Permutation

Optional

- Block cipher as a "**permutation function**":
  - For a **fixed key**, it is a function that maps **2^n plaintexts** to **2^n ciphertexts** (with a unique inverse for each ciphertext)
  - In other words: ***C*** is a rearrangement of ***M*** (or itself, since ***M*** = ***C***)
- To be a block cipher, we need a **keyed pseudorandom/secure permutation**, so that:
  - The permutation should be determined by the **key**
  - Different keys must result into **different permutations**
  - The permutation should "**look random**": ***indistinguishability***

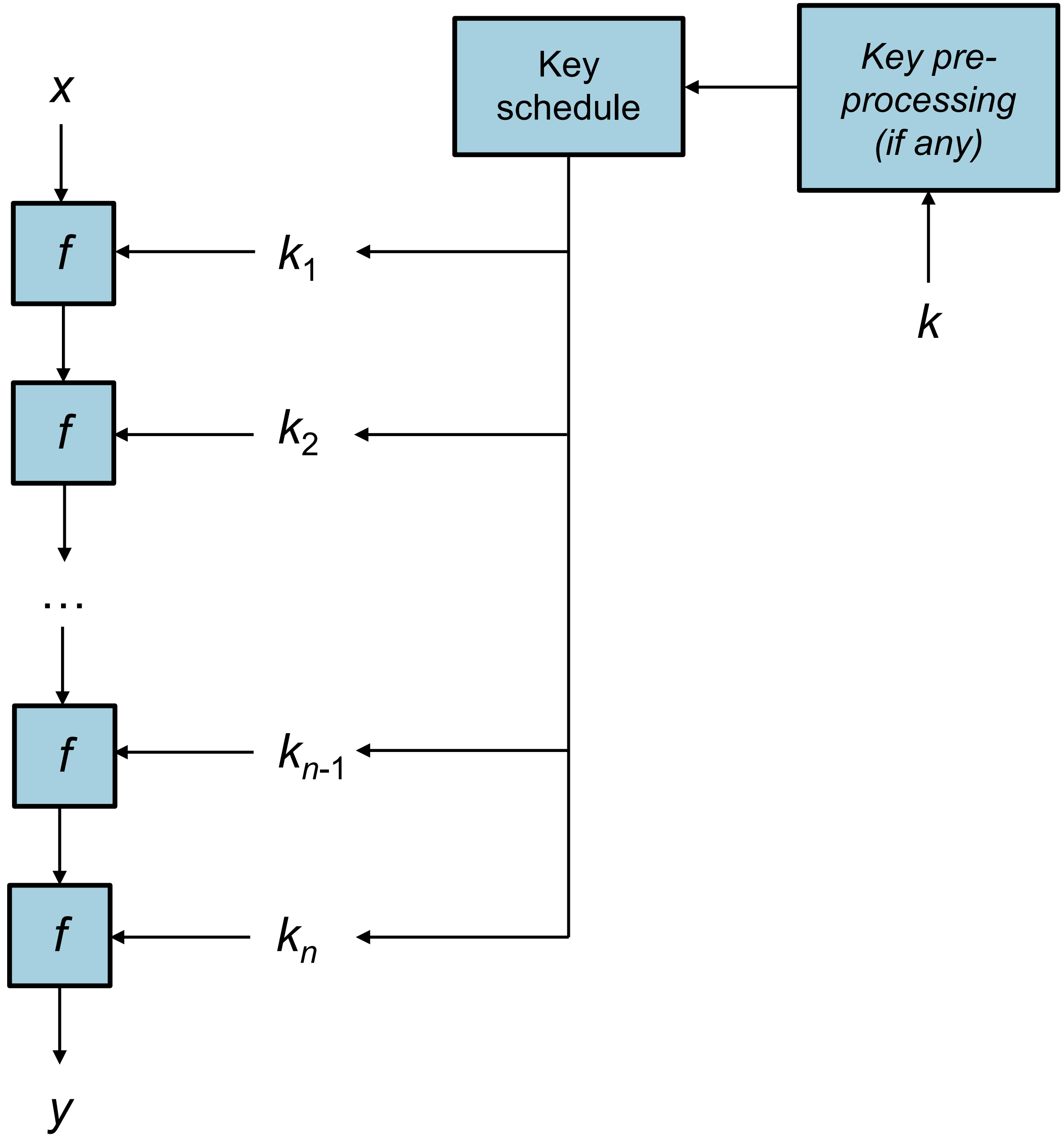
→ ***Keyed random mappings*** *between plaintexts and ciphertexts*

Note: Some people and books do not really like the explanation/abstraction of block ciphers by means of the permutation notion. The PRP, however, is the usual mathematical abstraction used for block ciphers, and can still improve our understanding about block ciphers and their requirements.

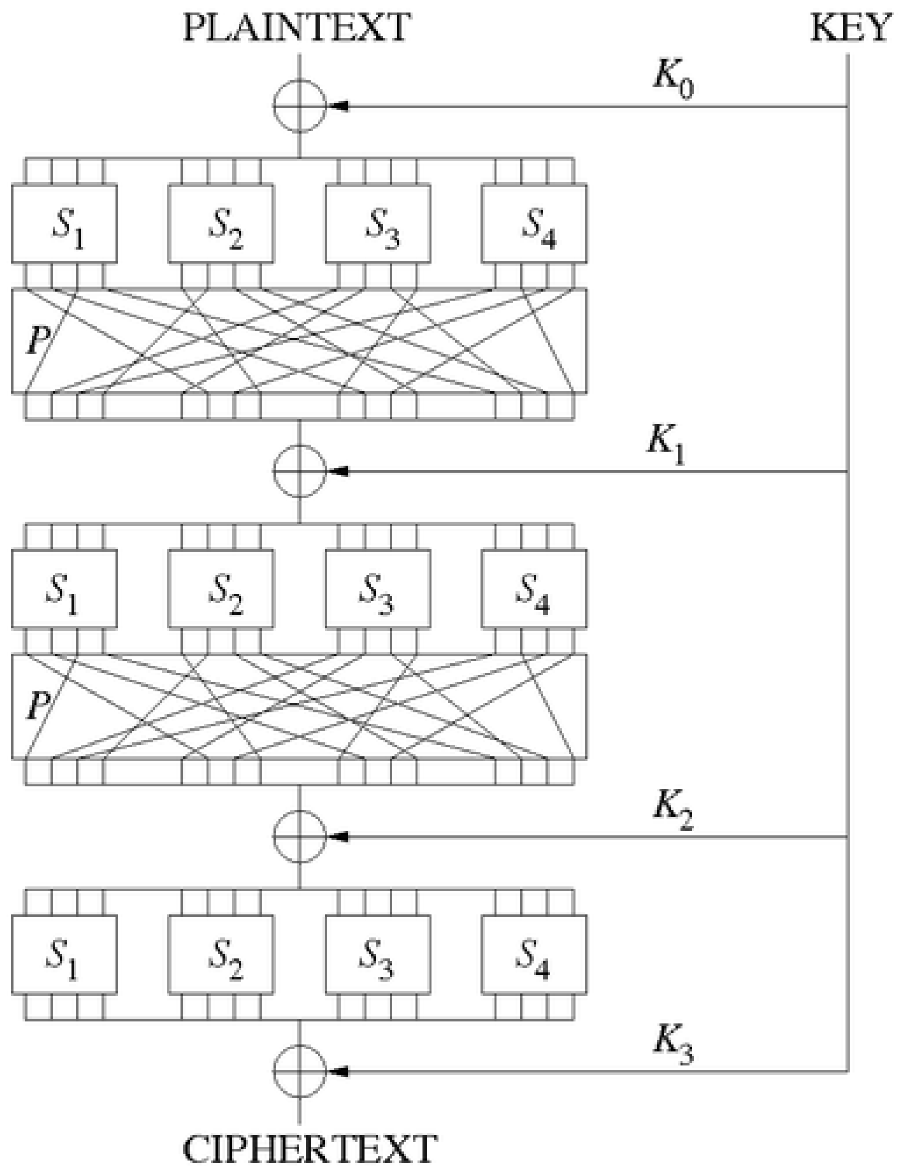
A Block Cipher: How It Typically Works

- How does a block cipher work?
  - Typically it is *not* a single gigantic algorithm, but **an iteration of rounds**: DES = 16 rounds, 3DES = 48 rounds, AES-128 = 10 rounds
  - Two **main techniques** for each round:
    substitution–permutation network (as in AES) and
    Feistel scheme (as in DES)
- A block cipher's **round** *(see the next slide for an illustration)*:
  - Simple operations in each round: easy to specify, implement and analyze
  - A **round function** $f(x,k)$
  - The key may have first undergone a **pre-processing**, i.e. key expansion
  - A **key schedule function** produces a sequence of ***round keys (subkeys)*** $k_1, k_2, \ldots k_n$:
    the same round functions with two different round keys behave *differently*

Encryptions in Block Ciphers: Rounds and Key Scheduling



Example of a SPN with Three Rounds



Source: Wikipedia

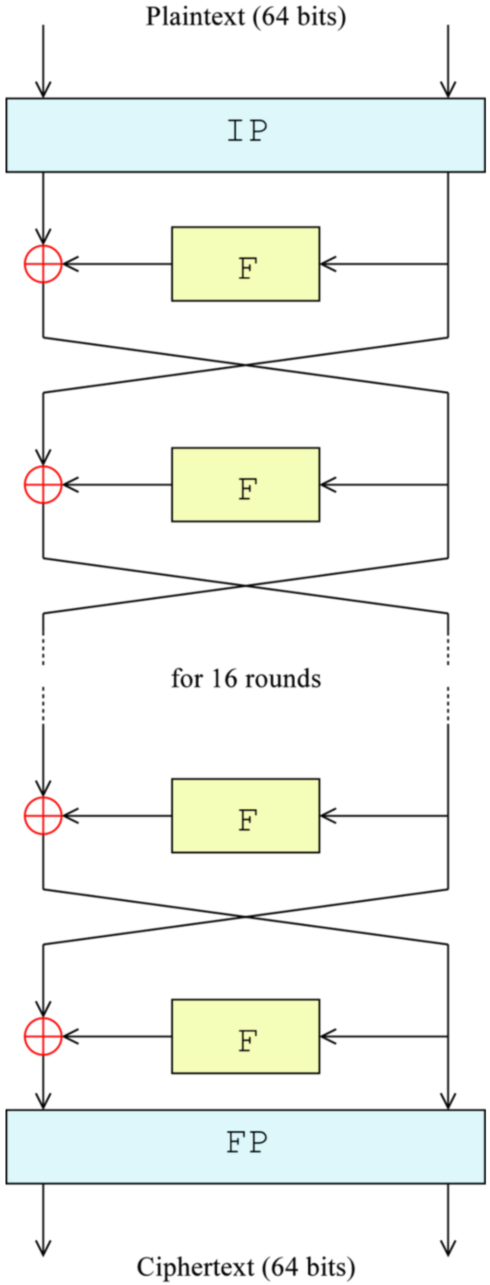
1.5.2 Popular Block Ciphers

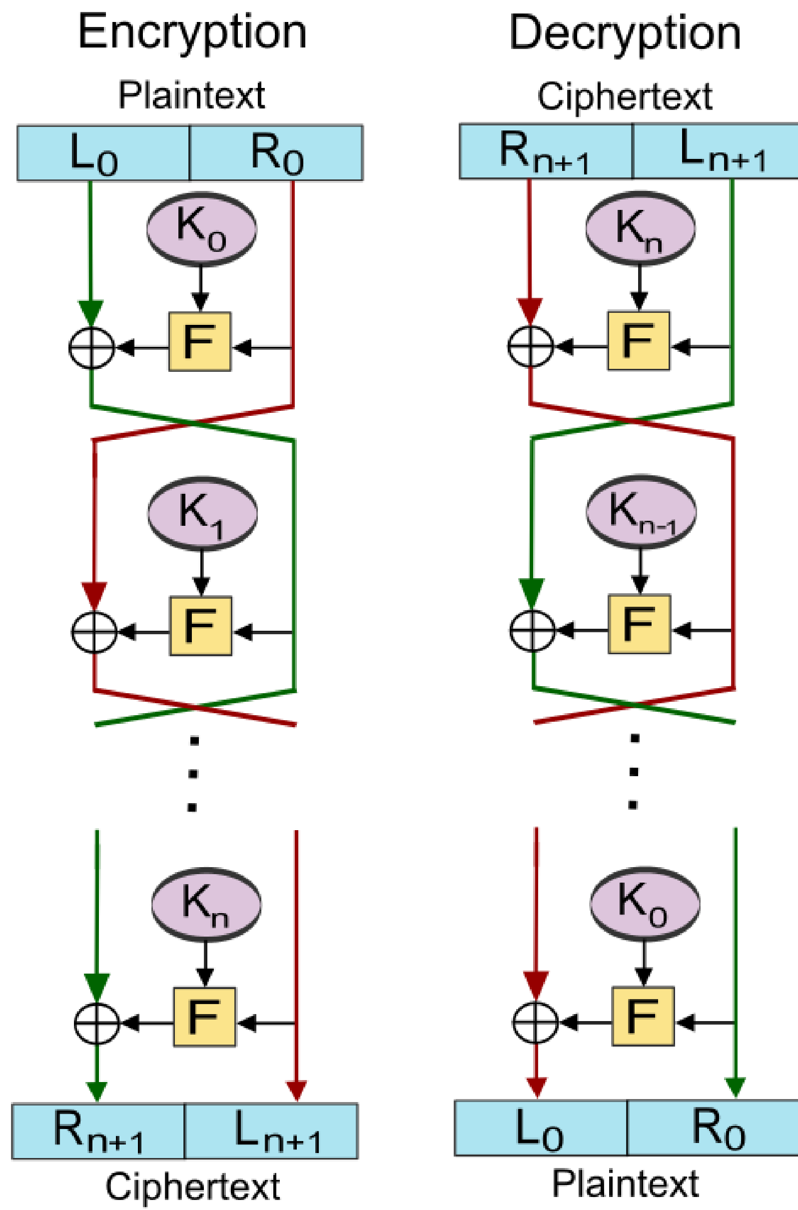
DES (Data Encryption Standard)

- Block length: **64** bits
- Key length: **56**
  (***not* long enough** for now, can be easily brute-forced!)
- Made as a federal standard in the US,
  and was widely used in banking and commerce
- Replaced by AES

- It works in **16 rounds** using a round function called ***Feistel function***), thus forming a Feistel Network
- Operations in Feistel function:
  - S-box performing substitution: for ***confusion***
  - P-box performing permutation: for ***diffusion***
- A special flow/circuit arrangement of the round functions, so that the encryption process is also **invertible**

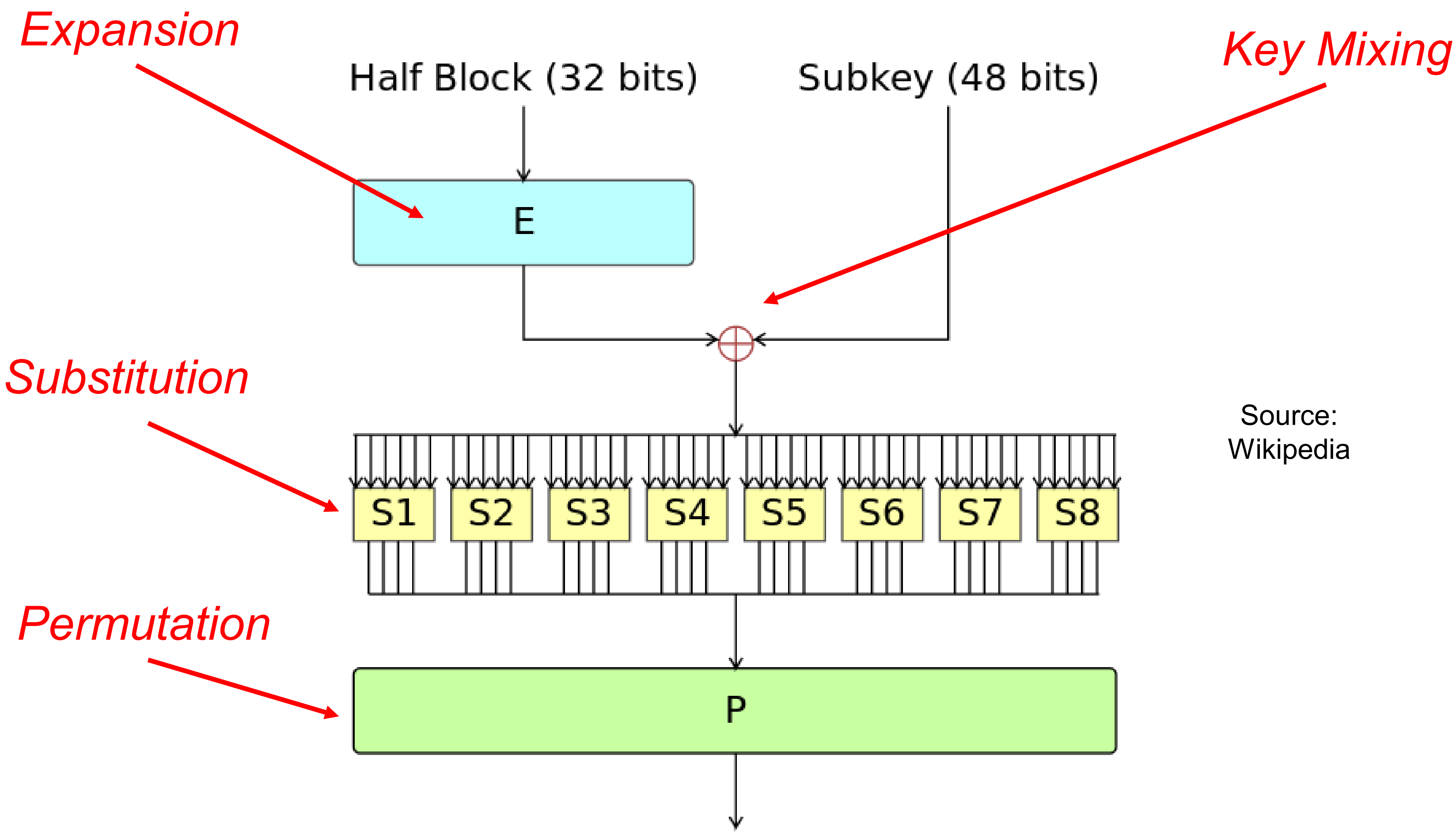
Feistel Network





Source: Wikipedia

Feistel Function in Each Round



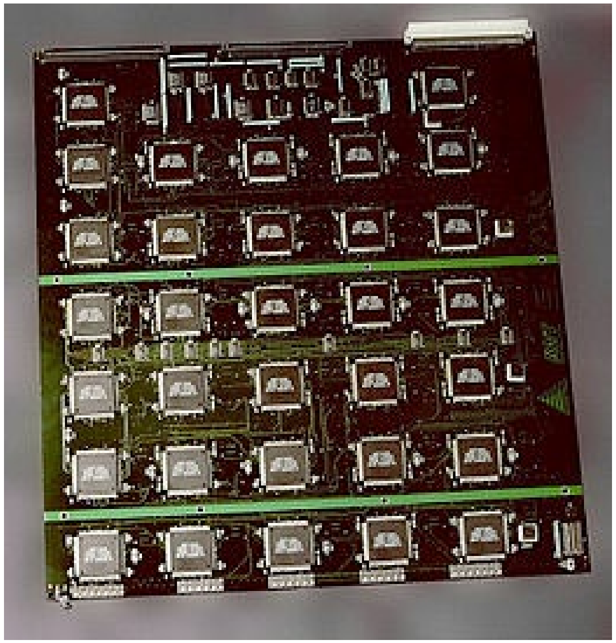
Source: Wikipedia

Exhaustive Search on DES

- Key length of DES is 56 bits
- While exhaustive search on 56 bits seemed infeasible **in the 70s**, very soon, it was possible using distributed computing or specialized chip
- *RSA Security* hosted a few **DES challenges**:
  - **DES Challenge II-1:** "*The secret message is: Many hands make light work.”*
    (Found in **39 days** using distributed computing, early 1998)
  - **DES Challenge II-2:** "*The secret message is: It's time for those 128-, 192-, and 256-bit keys*.”
    (Found in **56 hours** using a specialized hardware, 1998)
- (Note: *RSA* is an encryption scheme, whereas *RSA Security* is a company)

Exhaustive Search on DES

- EFF's DES cracking machine ("*Deep Crack*"):

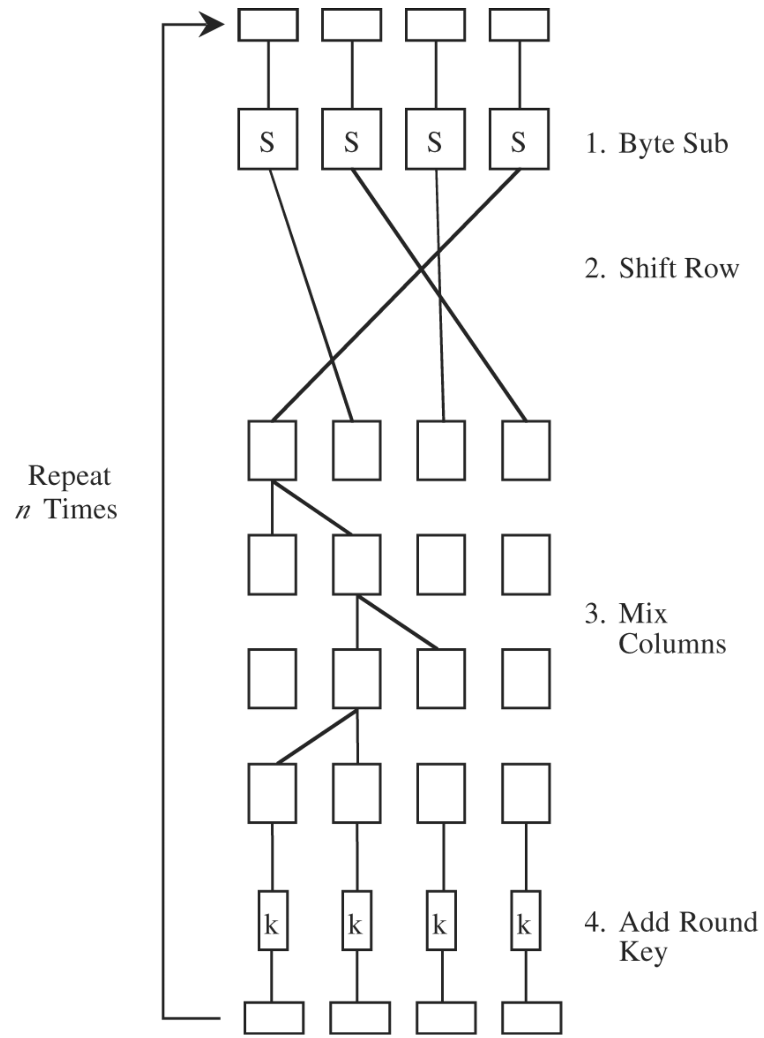


Optional: https://en.wikipedia.org/wiki/EFF_DES_cracker

- A question is: why would an agency designed a scheme that could be broken in the near future?
- Many believed that it was perhaps intentional

AES (Advanced Encryption System)

- In 1997, NIST called for proposal of a new **AES (Advance Encryption Standard)** block cipher
- The selection process was transparent and with worldwide involvement
- NIST received 21 submissions by Jun 1998
- In 2000, ***Rijndael***, invented by Belgian researchers Daemen and Rijmen, was selected as AES
- AES replaces DES, and is still in common use now



AES

- Block size: 128 bits
- Key sizes: 128, 192, 256 bits
  (the longer, the more secure, but the slower)
- A **Substitution and Permutation Network (SPN)**,
  and *not* a Feistel network:
  - Still substitution & permutation are used as building-block operations
  - In each round: substitution layer then permutation layer
  - **Substitution layer**: ByteSub operation
  - **Permutation layer**: ShiftRow and MixColumn operations
- Currently, no known attacks on AES:
  but there are some attacks the modes-of-operation
- NSA classifies AES as "Suite B Cryptography"

*"**NSA Suite B Cryptography** is a set of cryptographic algorithms [promulgated]() by the [National Security Agency]() as part of its [Cryptographic Modernization Program](). It is to serve as an interoperable cryptographic base for both unclassified information and most [classified information]()."*
See https://en.wikipedia.org/wiki/NSA_Suite_B_Cryptography

DES vs AES

	DES	AES
Date designed	1976	1999
Block size	64 bits	128 bits
Key length	56 bits (effective length); up to 112 bits with multiple keys	128, 192, 256 (and possibly more) bits
Operations	16 rounds	10, 12, 14 (depending on key length); can be increased
Encryption primitives	Substitution, permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion, diffusion	Confusion, diffusion
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret, but open public comments and criticisms invited
Source	IBM, enhanced by NSA	Independent Dutch cryptographers

1.5.3 Properties of Block Ciphers

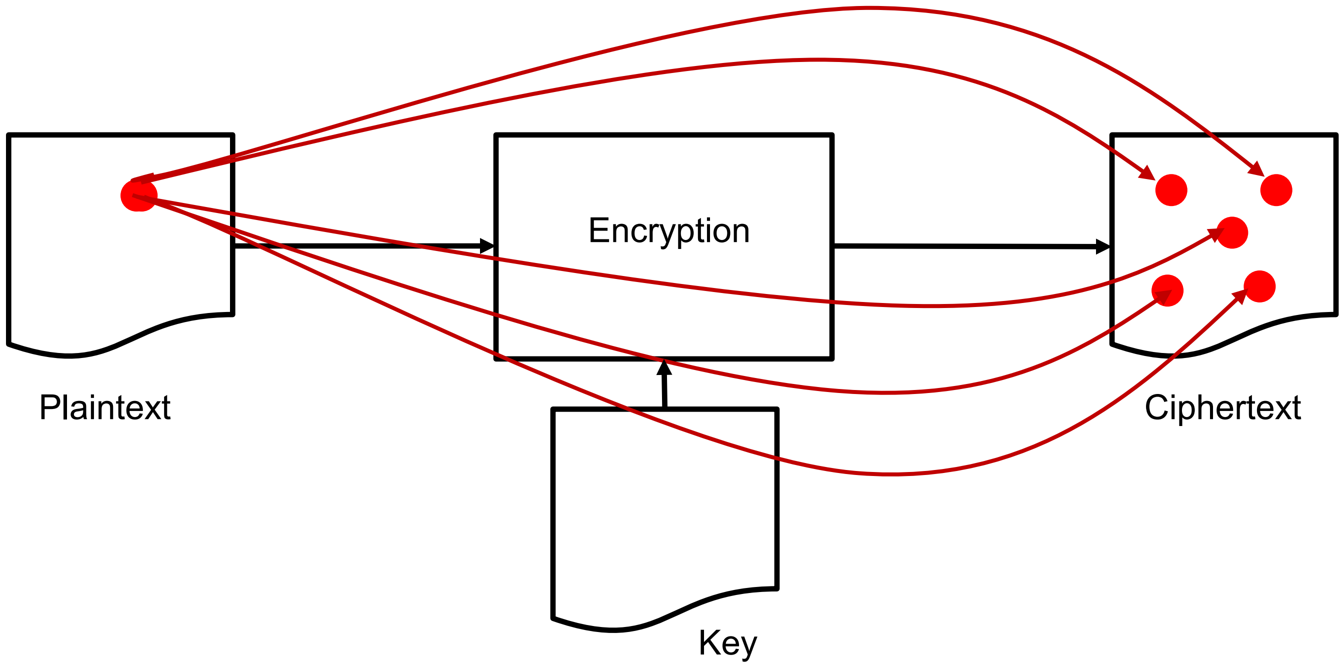
Stream vs Block Ciphers

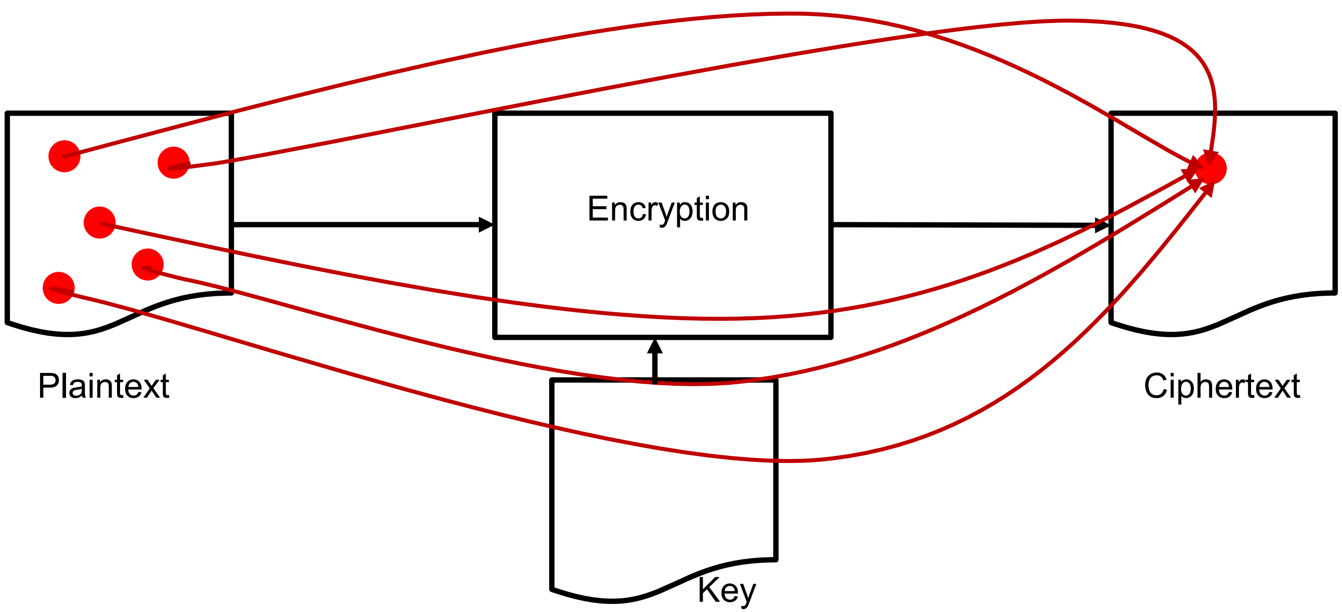
	Stream	Block
Advantages	• Speed of transformation • Low error propagation	• High diffusion • Immunity to insertion of symbol
Disadvantages	• Low diffusion • Susceptibility to malicious insertions and modifications	• Slowness of encryption • Padding • Error propagation

Properties of Ciphers: Diffusion

- Two properties of a cipher: diffusion and confusion
- ***Diffusion***: a change in the plaintext will **affect *many parts*** of the ciphertext
- This means:
  - Information from the plaintext is *spread over* the entire ciphertext
  - The transformations *depends equally* on all bits of the input
- A cipher with **good diffusion**:
  it requires an attacker to access *much of* the ciphertext
  in order to infer the encryption algorithm
- Block cipher: high diffusion
- Stream cipher: low diffusion (why?)

Diffusion Illustrated: Change Effect

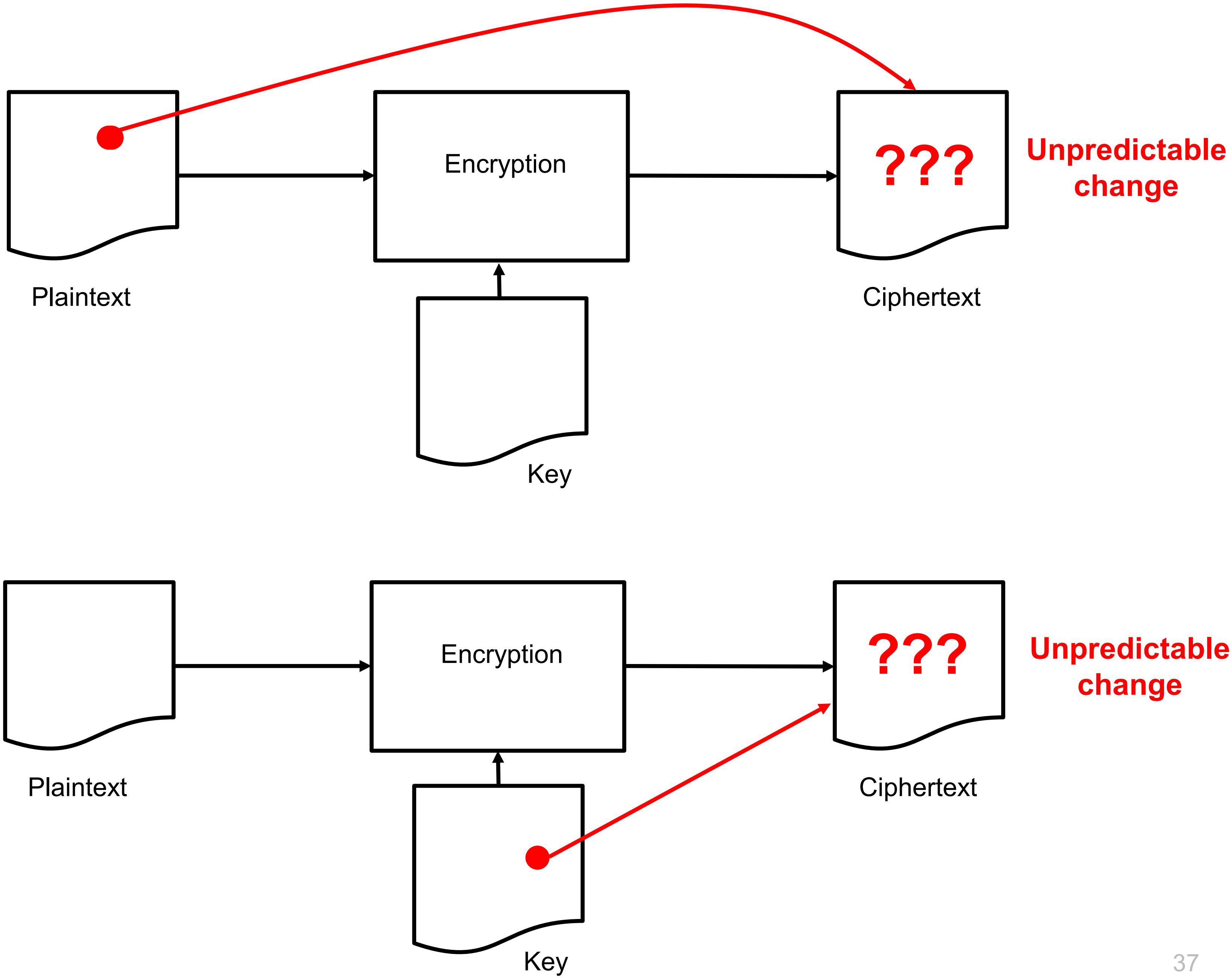




Properties of Ciphers: Confusion

- ***Confusion***: an attacker **should *not* be able to predict** what will happen to the ciphertext when **one character** in the plaintext or the key changes
- This means:
  - The input (i.e. plaintext and key pair) undergoes ***complex transformations*** during encryption
- A cipher with **good confusion**:
  it has a "*complex functional relationship*" between the plaintext/key pair and the ciphertext

Confusion Illustrated: Change Effect



1.5.4 Block Cipher Modes-of-Operation

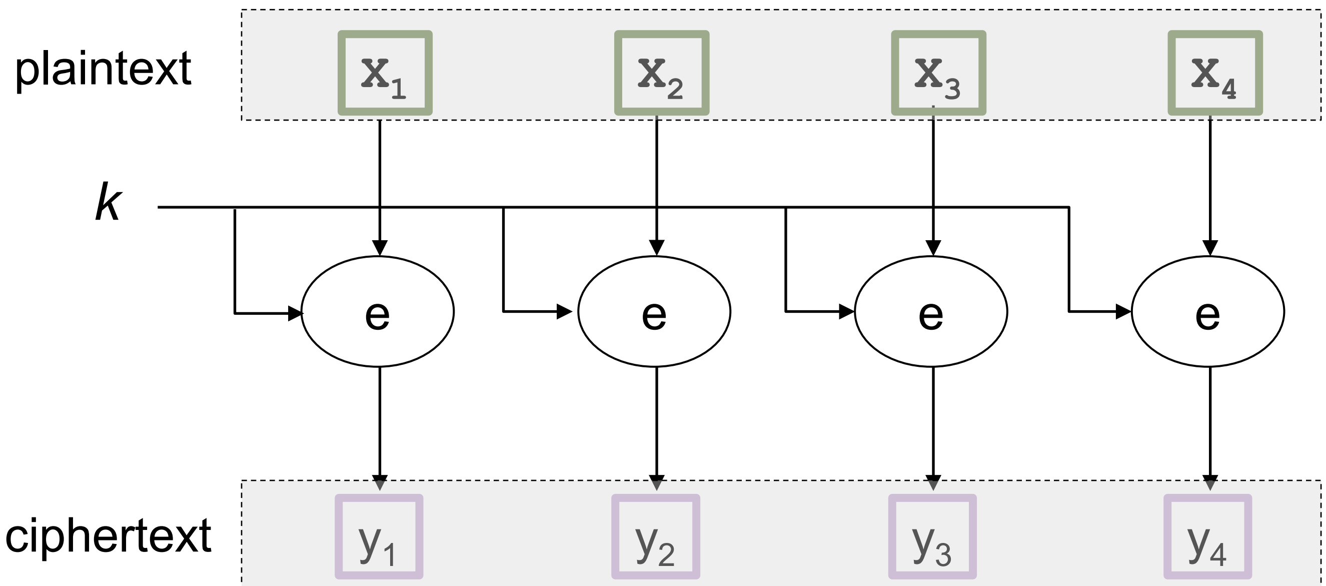
Block Cipher: Modes of Operation

- We have seen how a block cipher can encrypt ***n*-bit plaintext** with *n* as the cipher's block size
- How to encrypt an **arbitrarily long message** using a block cipher: i.e. when message length (say 10 MB) >> block size
- I.e. how to **extend block cipher** to arbitrary long plaintext?

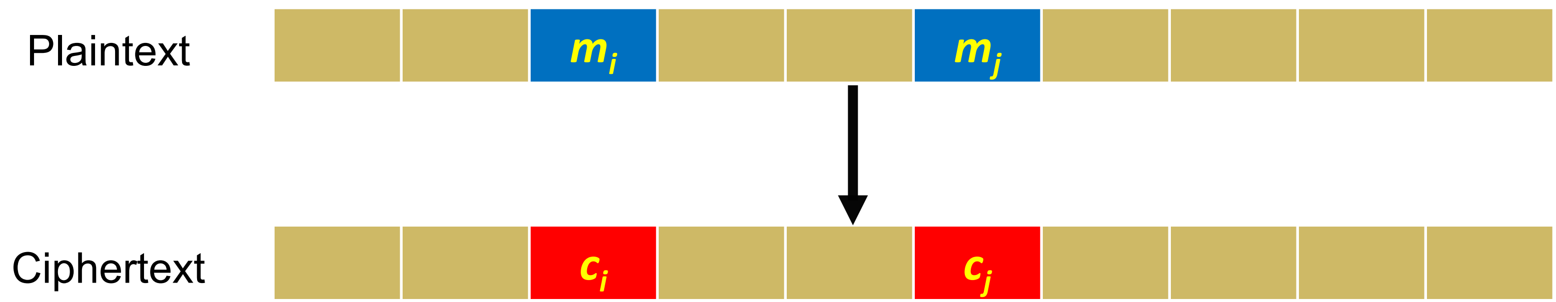
- A ***mode of operation***: a method of encrypting messages of arbitrary size using a block cipher
- Extending encryption from a single block to multiple blocks is however ***not*** straightforward: there are some **security implications** (see later slides)

Mode-of-Operation: ECB Mode

- (***Insecure***) **Electronic Code Book (ECB)** is the simplest mode
- It divides the plaintext into blocks, and then applies the block cipher in use to each block with the same key



Mode-of-Operation: Problem with ECB Mode

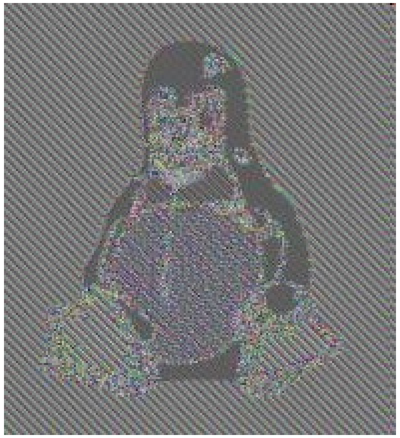


- What if the attacker find $c_i = c_j$?
- The attacker **can tell** that $m_i = m_j$
- Some information about the plaintext is **leaked**!
- *But, what's the big deal with this??*

Encrypting Tux, the Penguin

- ECB ***could leak*** information
- Suppose the **image** below is divided into blocks, and encrypted with some ***deterministic encryption scheme**** using the same key
- Since it is deterministic, any two plaintext blocks that are the same (e.g. from the white background) will be encrypted into the **same ciphertext**
- Tux, the Penguin, can be seen!

Plaintext



Ciphertext

Encrypting Tux, the Penguin: Additional Notes

- An encryption scheme is "**deterministic**" in a sense that the encryption algorithm always produces **the same output** (i.e. ciphertext) when given the same input (i.e. the key and plaintext)
- An example: **AES without the IV**
- In contrast, a "**probabilistic/randomized**" encryption scheme produces **different ciphertexts** even with the same input is given
- AES is deterministic, but if we employ **AES with a randomly-chosen IV**, then it becomes probabilistic (since the IV is different)

Problem Analysis and Possible Solution

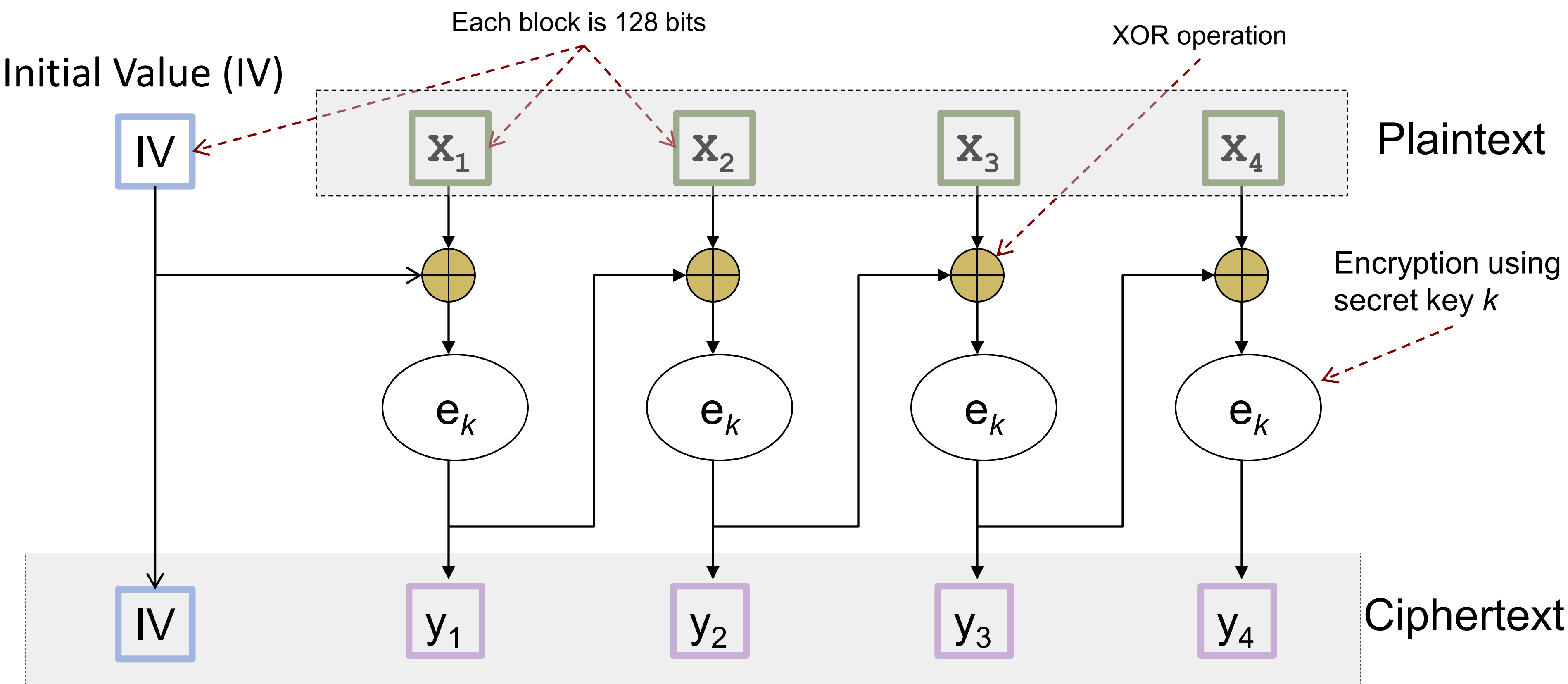
- What's the really **the issue** behind the problem?
- The same "***two-key problem***":
  - The same key is used for two (or multiple) different encryptions
  - The same plaintext block always gives the same ciphertext block
  - This is due to the deterministic encryption process

- *Additional mechanisms are required!*
- ***Question***: Why not just randomly choose an IV **for each block**, and hence achieve a probabilistic encryption so as to prevent the leakage?
  *(Answer: It will significantly increase the size of the final ciphertext, with **ciphertext-message expansion** of a factor of 2)*

Solution using Mode-of-Operation

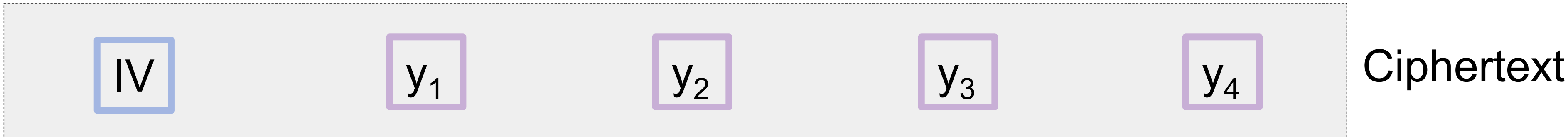
- A ***mode-of-operation*** describes how the blocks are to be "**linked**" so that different blocks at different locations would give different ciphertext, even if all the blocks have the same content
- Popular modes-of-operation: ***Cipher Block Chaining (CBC)*** and ***CTR (counter)*** modes
- Avoid the *Electronic Codebook (ECB)* mode, where "we can see the penguin"!

Mode-of-Operation: Cipher Block Chaining (CBC) on AES



Note: In the above figure, we treat **IV as part of the final ciphertext**. The terminology can be inconsistent in the literature. Some documents may state that "the final message to be sent are the IV and the ciphertext" (i.e. IV is not called the "ciphertext"). In this module, when it is crucial, we will explicitly state whether IV is excluded (e.g. AES without IV).

Cipher Block Chaining (CBC) Decryption



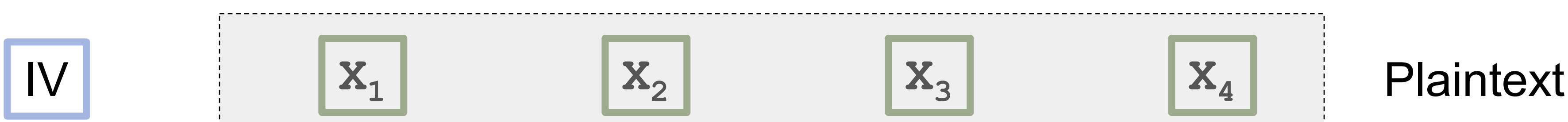
Some questions:

How about the decryption process?

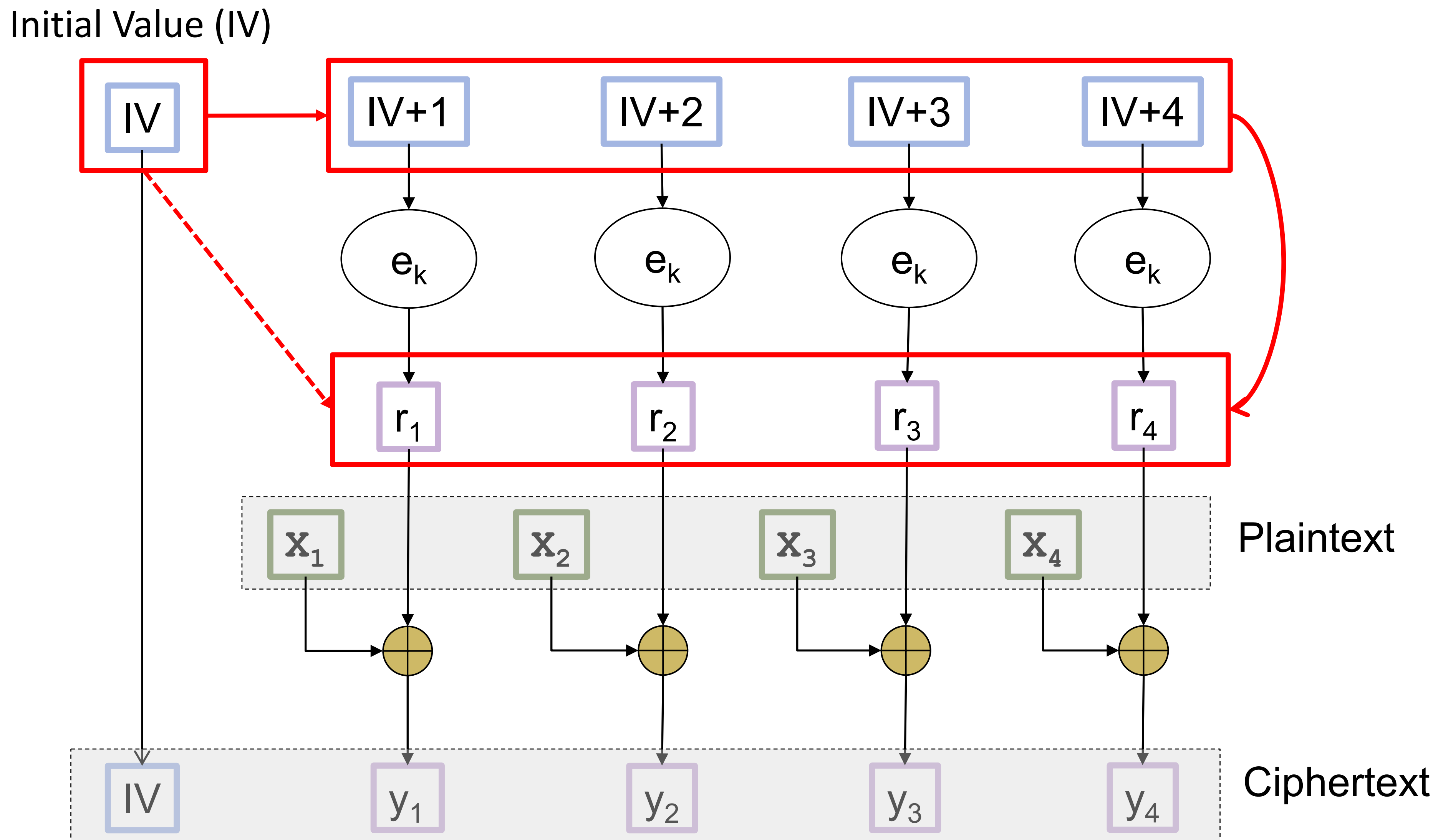
Also, what will happen if a ciphertext block gets corrupted?

Can encryption and decryption be parallelized?

See Tutorial 2



Mode-of-Operation: Counter (CTR) Mode on AES



This mode-of-operation turns a **block cipher** into a **stream cipher**!

1.5.5 Examples of Attacks on Block Ciphers

1.5.5.1 Meet-in-the-middle attack & Triple DES

1.5.5.1 Padding oracle attack:

Notions of Oracle in security analysis

The attack

Implications

1.5.5.1 Meet-in-the-Middle Attack & Triple DES

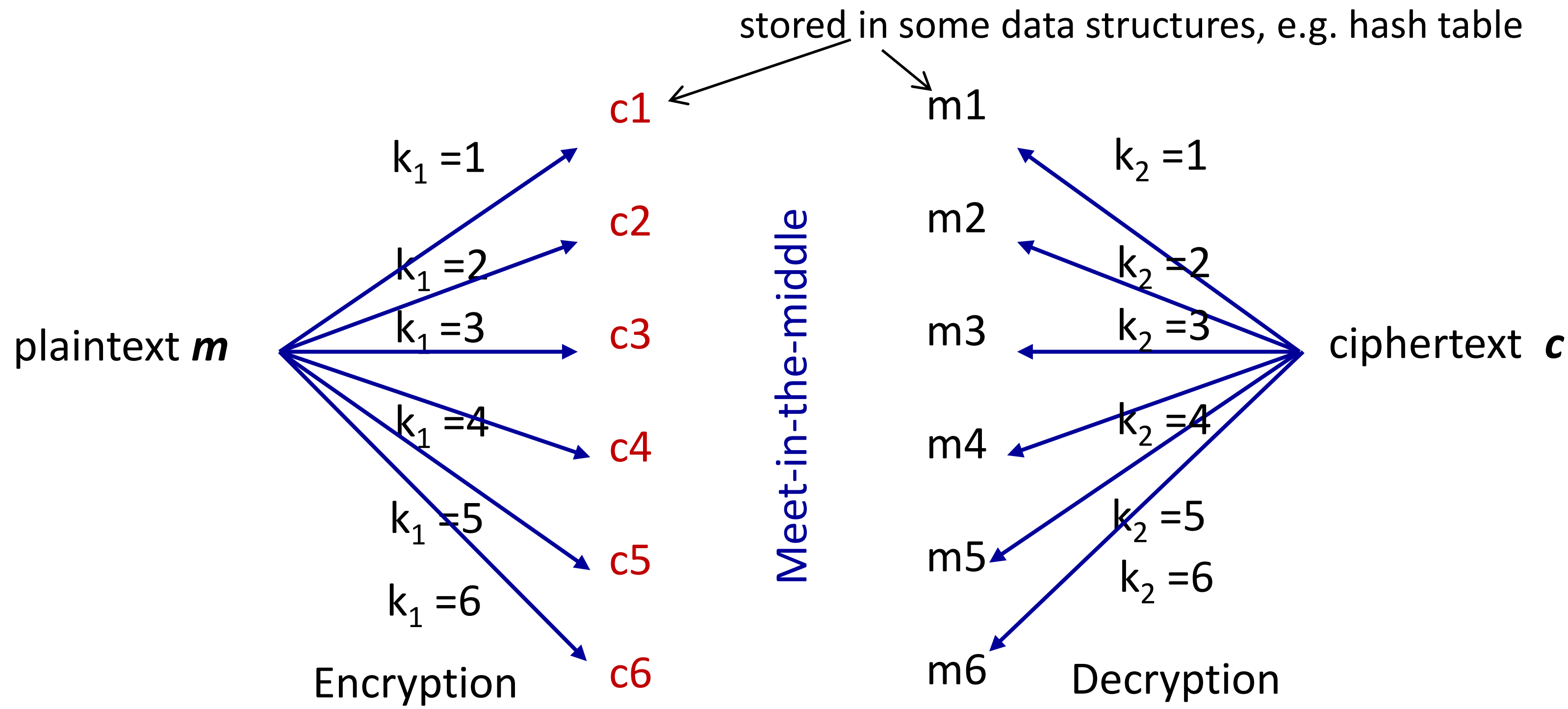
See: http://en.wikipedia.org/wiki/Meet-in-the-middle_attack

Double DES (2DES) and Meet-in-the-Middle Attack

- DES is not secure w.r.t. today computing power
- One way to improve it is by using multiple encryptions: encrypt using DES **multiple times** using different keys
- DES doesn't form a group: $E_{k_1}(E_{k_2}(x)) \neq E_{k_3}(x)$ for some k_3
- **2DES**: use **DES twice** by using two different keys $\boldsymbol{k_1}$, $\boldsymbol{k_2}$
- The key length is 2 * 56 bits = **112 bits** (hard to be brute-forced)
- But, any potential security issues?
- ***Is the real security strength also 112 bits,*** say under *known-plaintext attack* where the attacker has at least a pair (***m***, ***c***)?
- See Tutorial 2 for the ***meet-in-the-middle attack*** on 2DES

Note: meet-in-the-middle attack is different from man-in-the-middle attack (which is usually known and abbreviated as MitM attack)

Meet-in-the-Middle Attack *(As an Exercise in Tutorial 2)*



- **Problem**: Given *c* and *m,* the goal is to find the two keys used
- **Attack steps**:
  1. Compute two sets ***C*** and ***M*: *C*** contains ciphertexts of ***m*** encrypted with all possible keys; ***M*** contains plaintexts of ***c*** decrypted with all possible keys
  2. Find a common element in ***C*** and ***M***. From the common element, we can obtain the sought two keys.
- In the above meet-in-the-middle attack, the attacker only needs to perform 6 encryptions and 6 decryptions: in general, for ***k*-bit keys**, the attack reduces the number of crypto operations to $\mathbf{2^{k+1}}$ (using approx 2^{k+1} units of storage space)

Triple DES (3DES)

- Remedy: use **triple DES** encryptions

$$E_{k_3}(\mathbf{D}_{k_2}(E_{k_1}(x)))$$

- Some variants based on different keying options:
  - **3TDEA** or **triple-length keys**: 3 independent keys k_1, k_2 & k_3
  - **2TDEA** or **double-length keys**: 2 independent keys k_1, k_2 and $k_3 = k_1$
  - All keys are **identical**: $k_1 = k_2 = k_3$
- Running time? 3 times slower than DES

- **Encryption options**:

(a) $E_{k_3}(\mathbf{E}_{k_2}(E_{k_1}(x)))$ or

(b) $E_{k_3}(\mathbf{D}_{k_2}(E_{k_1}(x)))$

- Both options are believed to have the same level of security*
- Any benefits of using the second sequence construction? *See Tutorial 2!*

(Optional) Note: On the triple encryption, meet-in-the-middle attack takes 2^{112} encryption/decryptions. Interestingly, there are faster attacks. A known method reduces it to 2^{108} (S. Lucks, *Attacking Triple Encryption*, FSE 1998)

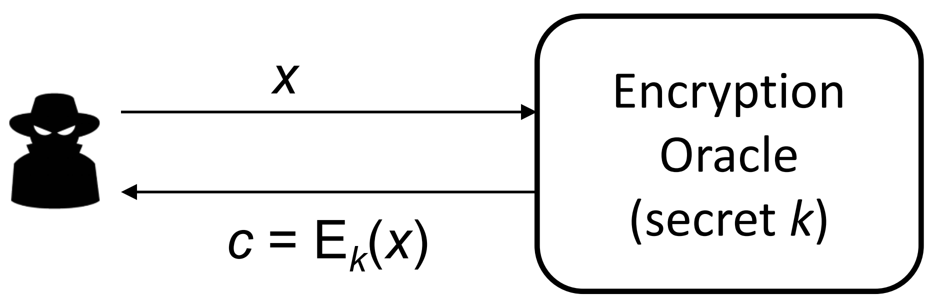
Triple DES (3DES): References and Notes

- For the **security** of 3DES variants with different keying options, see https://en.wikipedia.org/wiki/Triple_DES#Security
- 3DES was used extensively as a stopgap arrangement until AES was established:
  - 3DES was the default encryption in Outlook 2007 (see its help page: http://office.microsoft.com/en-sg/outlook-help/encrypt-messages-HP006369952.aspx)
- 3DES is still in use even today
- However, compared to AES, the 3DES is **less efficient**:
  - Sluggish in software
  - Can only encrypt 64-bit blocks at a time
- And **less secure**: 3DES has been deprecated by NIST in 2017 see https://en.wikipedia.org/wiki/Triple_DES#Security
- AES is thus much preferred now

1.5.5.2 Padding Oracle Attack

Oracle in Security Analysis

- Recap that in security analysis, it is important to formulate:
  (1) what information the attackers have (2) attackers' goals
- One type of information is obtained via a query-answer system known as ***Oracle***
- The attackers can send in queries, and the ***Oracle*** will output the answer:
  - **Encryption oracle**: On a query containing plaintext $\boldsymbol{x}$,
    the oracle outputs the ciphertext $E_k(\boldsymbol{x})$, where the key $\boldsymbol{k}$ is a secret key
  - **Decryption oracle**: On a query containing ciphertext $\boldsymbol{c}$,
    the oracle outputs the plaintext $D_k(\boldsymbol{c})$, where the key $\boldsymbol{k}$ is a secret key
- Note that an attacker can send multiple queries



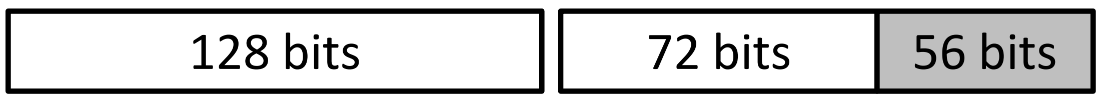
Padding Oracle attack

- The attacker have:
  - A ciphertext which include the IV: **(IV, c)**
  - Access to the Padding Oracle
- Attacker's **goal**:
  - The plaintext of (IV, **c)**
- Notes about the secret key:
  - The ciphertext is encrypted with a secret key *k*
  - The Padding Oracle knows *k*
  - The attacker does not know *k*: that's why it's launching an attack
- **Padding Oracle**:
  - Query: A ciphertext (which is encrypted using *k*)
  - Output: **YES**, if the plaintext is in the correct "padding" format

    NO, otherwise

Padding Format

- Recall again: the block size of AES is **128 bits** (16 bytes)
- Suppose the length of the plaintext is **200 bits**: it will be fitted into 2 blocks, with the remaining 56 bits "padded" with *some values*



- There are many ways to fill in the values
- In any case, an important **piece of information** must be encoded: the ***number of padded bits***
- If this info is missing, the receiver will not know the length of the actual plaintext
- The next slide gives a "standard" padding format

Padding using PKCS#7

- PKCS#7 is a padding standard:
  Read https://en.wikipedia.org/wiki/Padding_(cryptography)#PKCS7
- The following example is self-explanatory
- Suppose the block size is **8 bytes**, and the last block has only **5 bytes** (thus **3 extra padding bytes** required), the padding is done as follow:

DD DD DD DD DD DD DD DD	DD DD DD DD DD **03 03 03**

- In general, the padding bytes are:

01

02 02

03 03 03

04 04 04 04

...

- If the last block is full, i.e. it has 8 bytes: an **extra block** of **all zeros** is added

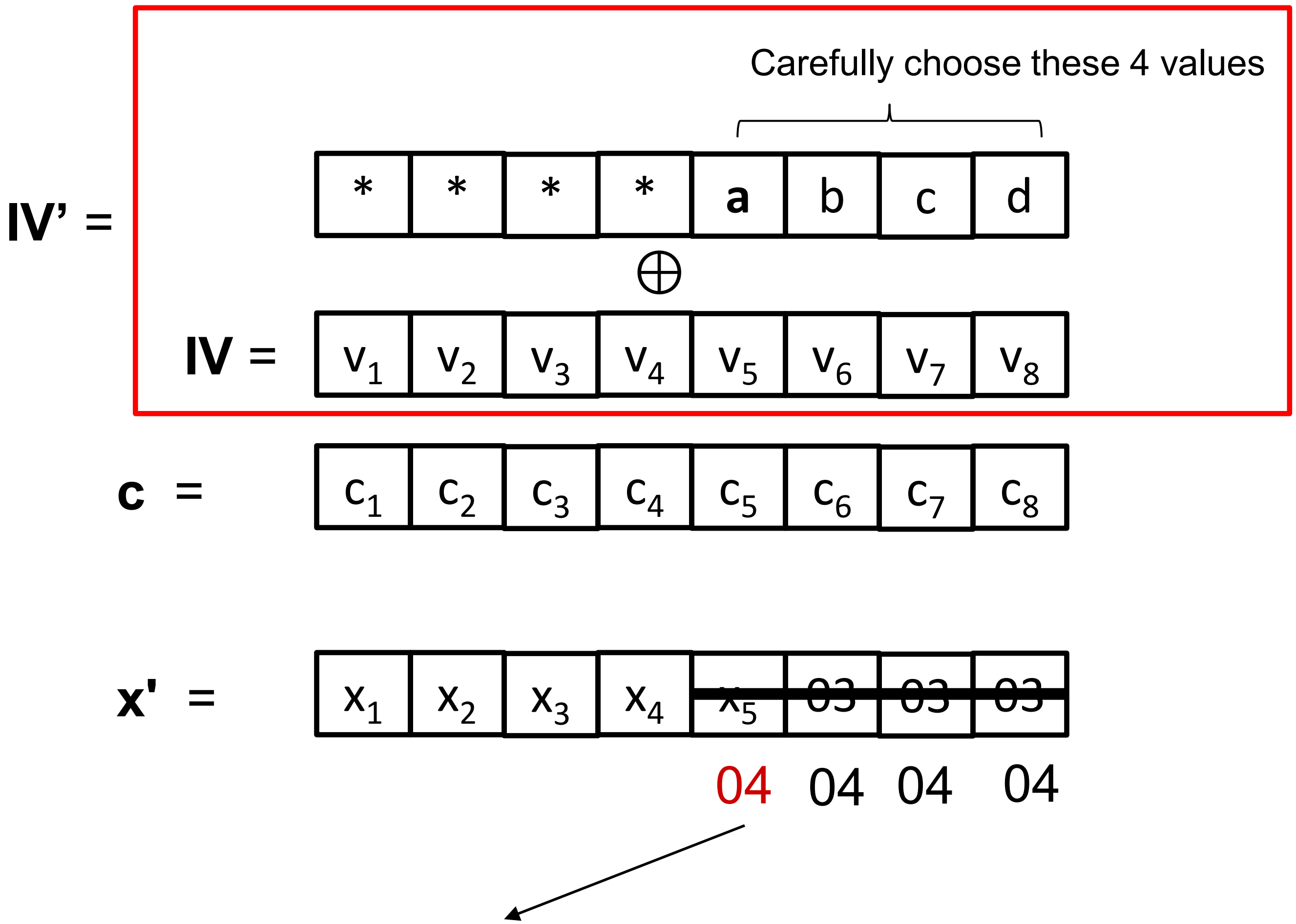
Padding Oracle Attack on AES CBC Mode

- **AES CBC mode** is ***not*** secure against padding oracle attack (when the padding is done with PKCS#7)
- Let us look at this example: the data sent to the Oracle is IV and c
- Attacker has (**IV** || **c**): 1 block of IV and 1 block of **c**
- For convenience, let us assume that the attacker knows that the block is **padded with 3 bytes**, i.e. the actual length of the plaintext is 5 bytes
- The attacker wants to find the value of x_5

IV =	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
c =	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
x =	x_1	x_2	x_3	x_4	x_5	03	03	03
	?	?	?	?	?			

The Attack's Main Idea

Carefully choose the 4 values (a, b, c, d), with **a** being brute-forced:



Wait until the Padding Oracle says that x' is correctly padded (with 4 x "04")!
And we can then determine x_5 in the actual x: see the next slide

Padding Oracle Attack on AES CBC Mode

Note that:

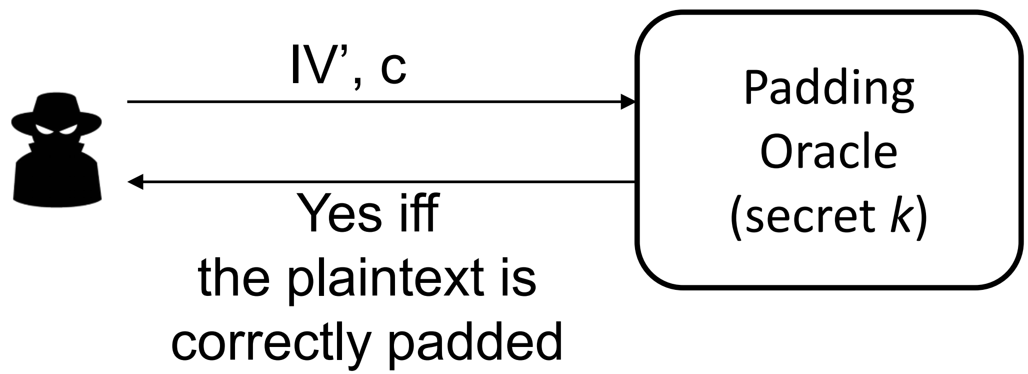
$07 = 04 \oplus 03$

This algorithm outputs the value of $\boldsymbol{x_5}$:

1. For t = 0 to FF // *hexadecimal representation*
2. Let **IV'** = IV $\oplus$

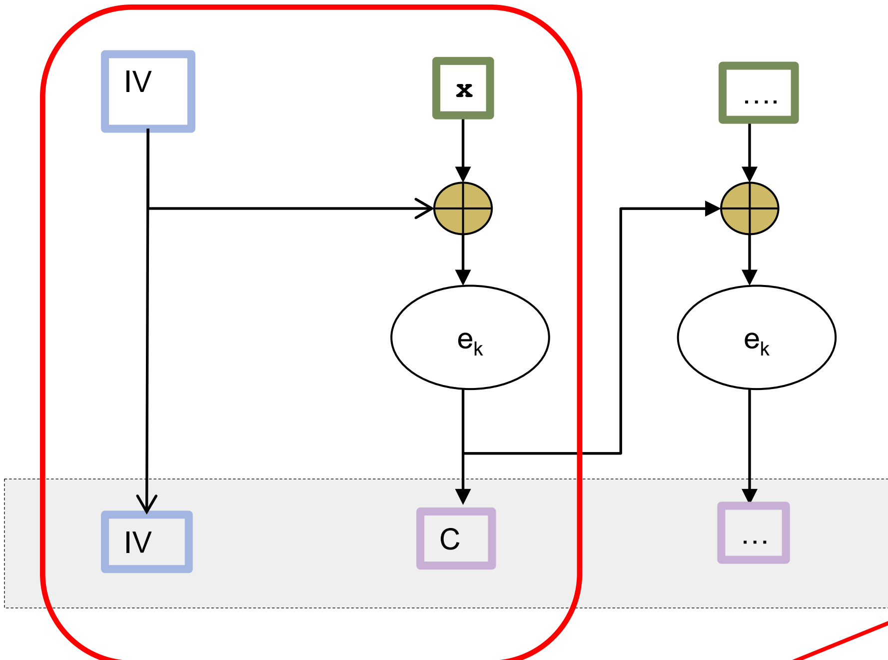
0	0	0	0	t	07	07	07

3. Sends the two-block query (**IV' ∥ c**) to ***Padding Oracle***
4. If ***Oracle*** gives **YES**, then outputs: $04 \oplus t$
5. End-for-loop

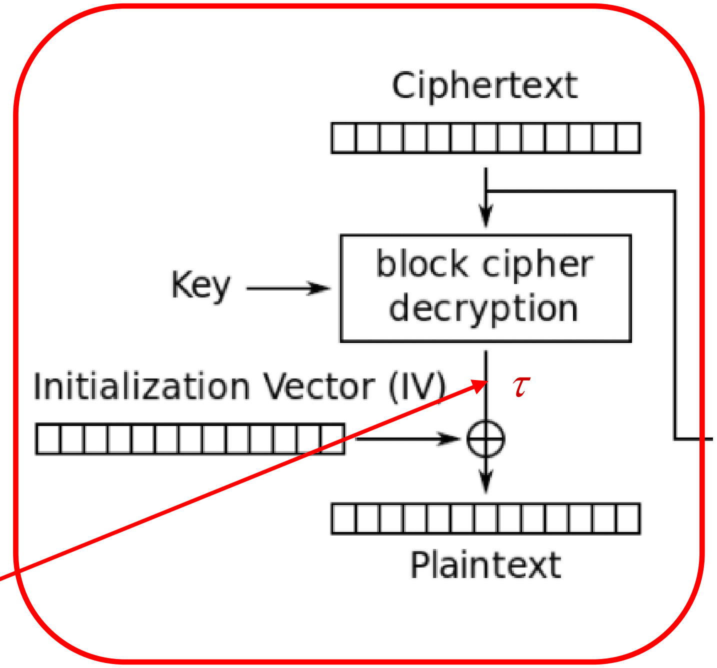


Why Does It Work?

- CBC encryption:



Corresponding CBC decryption (partial):



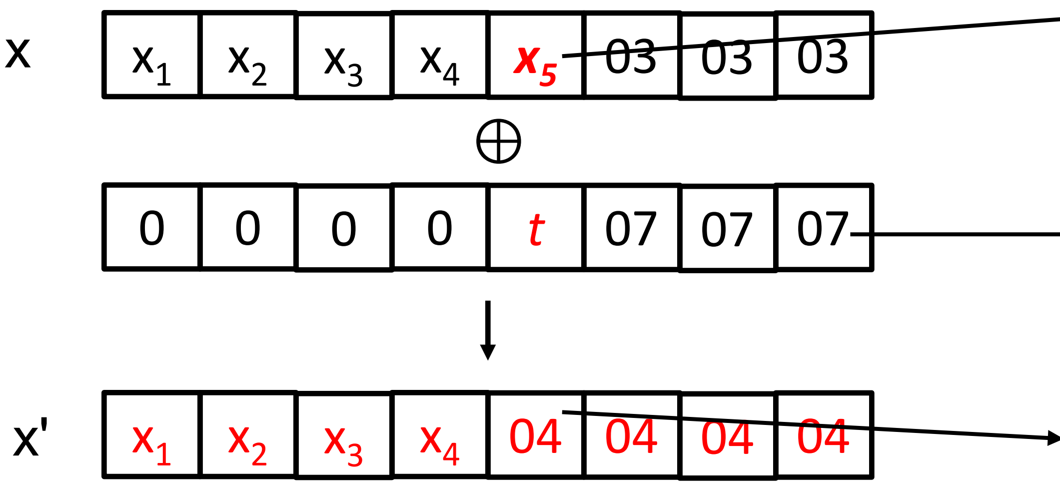
- Note that the attack modifies IV into IV', but keeps **c** the same
- Hence, τ remains the same in the normal and attack cases
- What is τ (known to the Oracle only, since the key is kept by it)?
- From the normal case: $\tau \oplus$ IV = x; thus, τ = IV $\oplus$ x

Why Does It Work?

- As inferred, $\tau = IV \oplus x$
- In the successful attack **when the Oracle gives YES**, what is the ***produced accepted plaintext* x'**?

$$
\begin{aligned}
x' &= \tau \oplus IV' \\
&= (IV \oplus x) \oplus (IV \oplus \boxed{0\,|\,0\,|\,0\,|\,0\,|\,t\,|\,07\,|\,07\,|\,07}) \\
&= x \oplus \boxed{0\,|\,0\,|\,0\,|\,0\,|\,t\,|\,07\,|\,07\,|\,07}
\end{aligned}
$$

Q: What does the accepted x' tell about $\mathbf{x_5}$?



(4) Given t, the attacker now can know: $\mathbf{x_5 = 04 \oplus t}$

(1) Note again that: $07 = 04 \oplus 03$

(3) But the Oracle tells this 04 to the attacker, which also tells the correct t

(2) x' as derived & known by the Oracle only

See also: https://en.wikipedia.org/wiki/Padding_oracle_attack
https://roberttheaton.com/2013/07/29/padding-oracle-attack/

Additional Remarks

- We can easily extend the algorithm to find **all the plaintext**
- The algorithm need to know the **plaintext's length**:
  it is possible to determine the length (left as an **optional** exercise)
- This attack is **practical**: there are real-world protocols* between a client and server that performs this:

  If the client submits a ciphertext whose plaintext is not padded correctly, the server will reply with an error message

- If an attacker obtains a ciphertext, the attacker can carry out the **protocol** with the server so as to get the plaintext

* A *protocol* specifies interactions between two or more entities

Important Lessons from Padding Oracle Attack

- The notion of ***Oracle***
- Padding oracles are **frequently present** in web apps:
  - Apps can return **explicit** error messages; or
  - Apps give **implicit** error messages: an attacker might be able to detect differences in **externally-observable behavior** of the oracle
- There are situations where, although the attacker has **seemingly useless information**, there are ways to **exploit** the information to extract sensitive info
- A **wrong use of encryption** (which protects confidentiality) to provide integrity: encryption is not to protect integrity

1.6 Cryptography Pitfalls: Attacks on Cryptosystem Implementations

A secure cipher can be vulnerable if it is **not implemented properly**

This section gives some examples:

- 1.6.1 – Reusing IV, wrong choice of IV & key in one-time-pad
- 1.6.2 – Predictable secret-key generation
- 1.6.3 – Designing your own cipher

See also 1.7 – Reliance on obscurity (disregarding Kerckhoff's principle)

(**To be studied later**: Using encryption for the wrong purpose,
e.g. using encryption scheme to ensure ***message integrity***)

1.6.1 Reusing IV, Wrong Choices of IV & One-Time-Pad Key

Reusing IV and Wrong Choices of IV

- Some applications overlooked IV generation.
  As a result, under some situations, the same IV is **reused**.

- E.g. To encrypt a file *F*, the IV is derived from *the filename*.
  It is quite common to have files with the same filename.

  (Read "**Schneier on Security, Microsoft RC4 Flaw"**:

  https://www.schneier.com/blog/archives/2005/01/microsoft_rc4_f.html

  http://eprint.iacr.org/2005/007.pdf)

- E.g. When using AES under the "CBC mode", the IV has to be **unpredictable** to prevent a certain type of attack.
  (Hence, it is vulnerable to choose IV as 1, 2, 3,....).

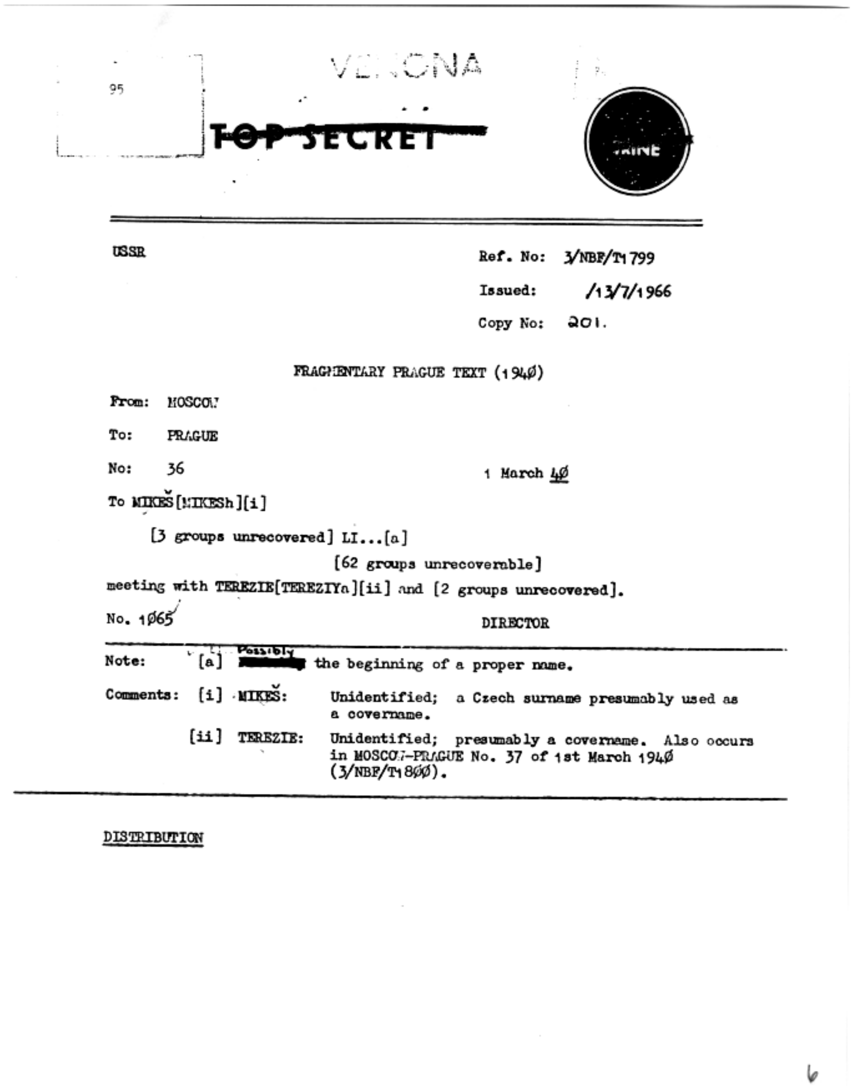
  The well-known BEAST attack exploits this:

  (Optional: http://resources.infosecinstitute.com/ssl-attacks/)

Reusing One-Time-Pad Key

- The Venona project is a classic example on such failure

  (Optional: https://www.nsa.gov/about/_files/cryptologic_heritage/publications/coldwar/venona_story.pdf)



95

VENONA

~~TOP SECRET~~

USSR

Ref. No: 3/NBF/T1799

Issued: /13/7/1966

Copy No: 201.

FRAGMENTARY PRAGUE TEXT (1940)

From: MOSCOW

To: PRAGUE

No: 36 1 March 40

To MIKEŠ[MIKESh][i]

[3 groups unrecovered] LI...[a]

[62 groups unrecoverable]

meeting with TEREZIE[TEREZIYa][ii] and [2 groups unrecovered].

No. 1065 DIRECTOR

Note: [a] Possibly [redacted] the beginning of a proper name.

Comments: [i] MIKEŠ: Unidentified; a Czech surname presumably used as a covername.

[ii] TEREZIE: Unidentified; presumably a covername. Also occurs in MOSCOW-PRAGUE No. 37 of 1st March 1940 (3/NBF/T1800).

DISTRIBUTION

1.6.2 Predictable Secret-Key Generation

Random Number Generation

- **Scenario 1:**
  - You are coding a program for a **simulation system**, for e.g. to simulate road traffic
  - In the program, you need a sequence of random numbers, for e.g. to decide the speed of the cars
  - *How to get these random numbers?*

- **Scenario 2:**
  - You are coding a program for a **security system**
  - In the program, you need a random number, for e.g. you need to generate a random number as a temporary **secret key**
  - *How to get these random numbers?*

To be Discussed in Tutorial

- In Java, what is the difference between the following?
  - `java.util.Random`
  - `java.security.SecureRandom`

- In C, what is the difference between using the following:

```
#include <time.h>
#include <stdlib.h>
     srand(time(NULL));
     int r = rand();
```

and a more complicated version below?

```
int byte_count = 64;
char data[64];
FILE *fp;
     fp = fopen("/dev/urandom", "r");
     fread(&data, 1, byte_count, fp);
     fclose(fp);
```

1.6.3 Designing Your Own Cipher

Caution!

- **Don’t** design your own cryptosystem,
  or even make a slight modification to existing scheme,
  unless you has an in-depth knowledge of the topic!

- Read “*Don’t roll your own crypto*”:
  http://security.stackexchange.com/questions/2202/lessons-learned-and-misconceptions-regarding-encryption-and-cryptology/2210#2210

1.7 Kerckhoffs' Principle vs Security through Obscurity

Kerckhoffs' Principle (La Cryptographie Militaire, 1883)

- "A system should be secure even if everything about the system, *except the secret key*, is a public knowledge. (It can be stolen by the enemy without causing trouble.)"
- Why is this principle useful?
  - It is easier to keep secret key vs secret algorithm
  - It is easier to change secret key vs secret algorithm
  - Standardized algorithm allows for easy deployment
  - Public scrutiny on open algorithm: peer review & security validation

Security through Obscurity

- To hide the design of the system in order to achieve security
- *Is it good or bad??*

Examples (*Against* Obscurity)

- RC4:
  - Was introduced in 1987 and its algorithm was a **trade secret**
  - In 1994, a description of its algorithm was **anonymously posted** in a mailing group.
  - See http://en.wikipedia.org/wiki/RC4

- MIFARE Classic:
  - A contactless smartcard widely used in Europe employed a set of proprietary protocols/algorithms
  - However, they were **reverse-engineered** in 2007
  - It turned out that the encryption algorithms were already known to be weak (using only 48bits) and breakable
  - See http://en.wikipedia.org/wiki/MIFARE
  - Optional: Presentation video by the researcher who reverse-engineered it: http://www.youtube.com/watch?gl=SG&hl=en-GB&v=QJyxUvMGLr0.
    The algorithm was revealed at 14:00.)

Examples (*Supporting* Obscurity)

- Usernames:
  - They are not secrets
  - However, it is *not* advisable to publish all the usernames
- Computer network structure & settings:
  - E.g. location of firewall and the firewall rules
  - These are not secrets, and many users within the organization may already know the settings
  - Still, it is *not* advisable to them
- The actual program used in a smart-card:
  - It is not advisable to publish it
  - If the program is published, an adversary may be able to identity vulnerability that was previously unknown, or carry out side-channel attacks
  - A sophisticated advisory may be able to reverse-engineer the code nevertheless

So, Should We Use Obscurity???

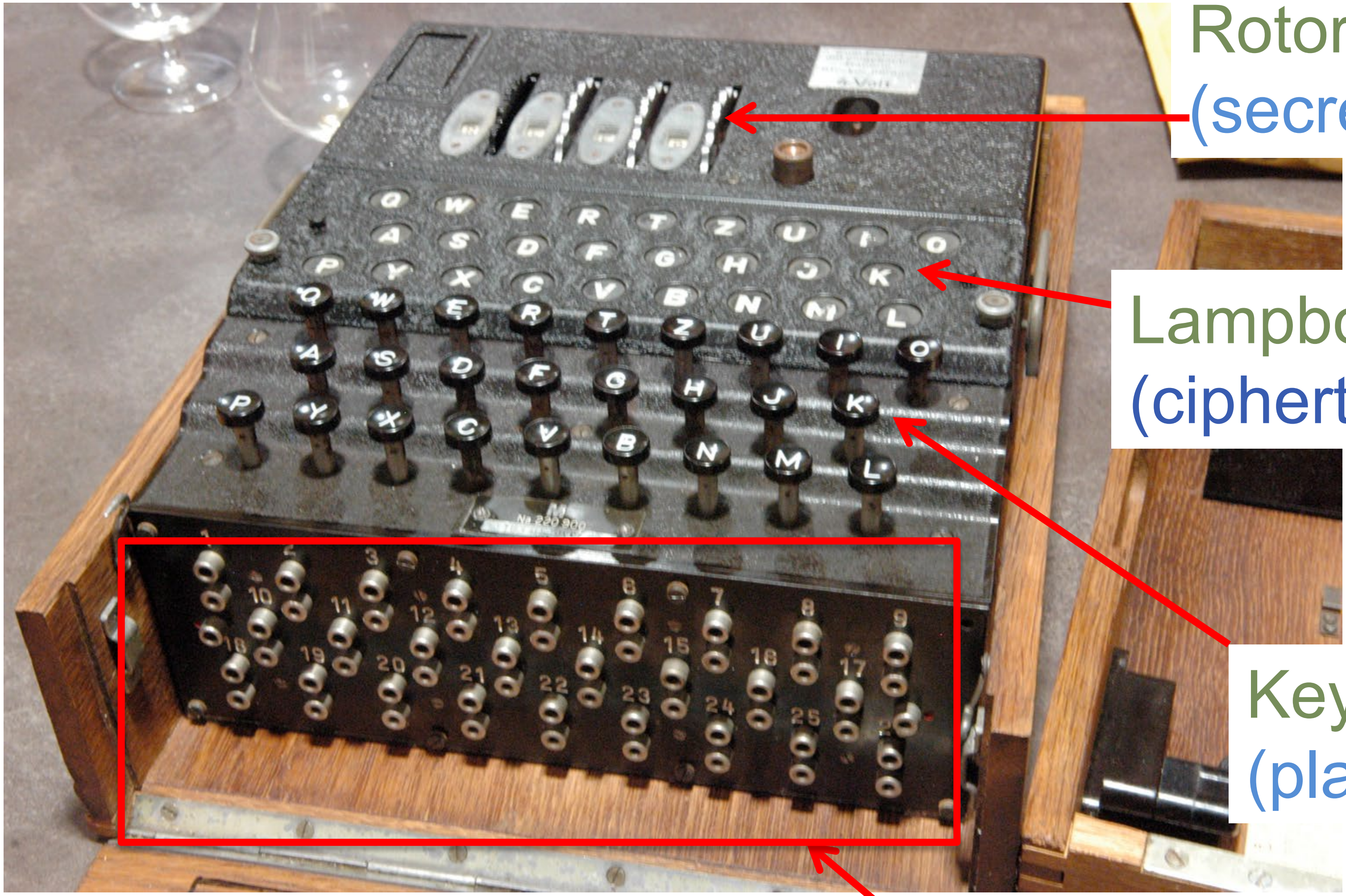
- In general, obscurity can be used as one layer in a ***defense in depth*** strategy
- It could deter or discourage novice attackers, but is ineffective against attackers with high skill and motivation
- The system **must remain secure** even if everything about it, *except its secret key*, becomes known
- In this module, we always assume that the attackers **know** the algorithms
- See:
  - http://technet.microsoft.com/en-us/magazine/2008.06.obscurity.aspx
  - http://en.wikipedia.org/wiki/Security_through_obscurity

1.8 Some Historical Facts

Cryptography: History

- Cryptography is closely related to **warfare** and can be traced back to ancient Greece
- Its role became significant when information is sent **over the air**
- **Cryptanalysis** is one of the driving forces to the invention of computer (e.g. Colossus computer, See https://en.wikipedia.org/wiki/Colossus_computer)
- WWII: Famous encryption machines include the **Enigma**, and the **Bombe** (that helped to break Engima)

Enigma (Replica)



Rotors
(secret key)

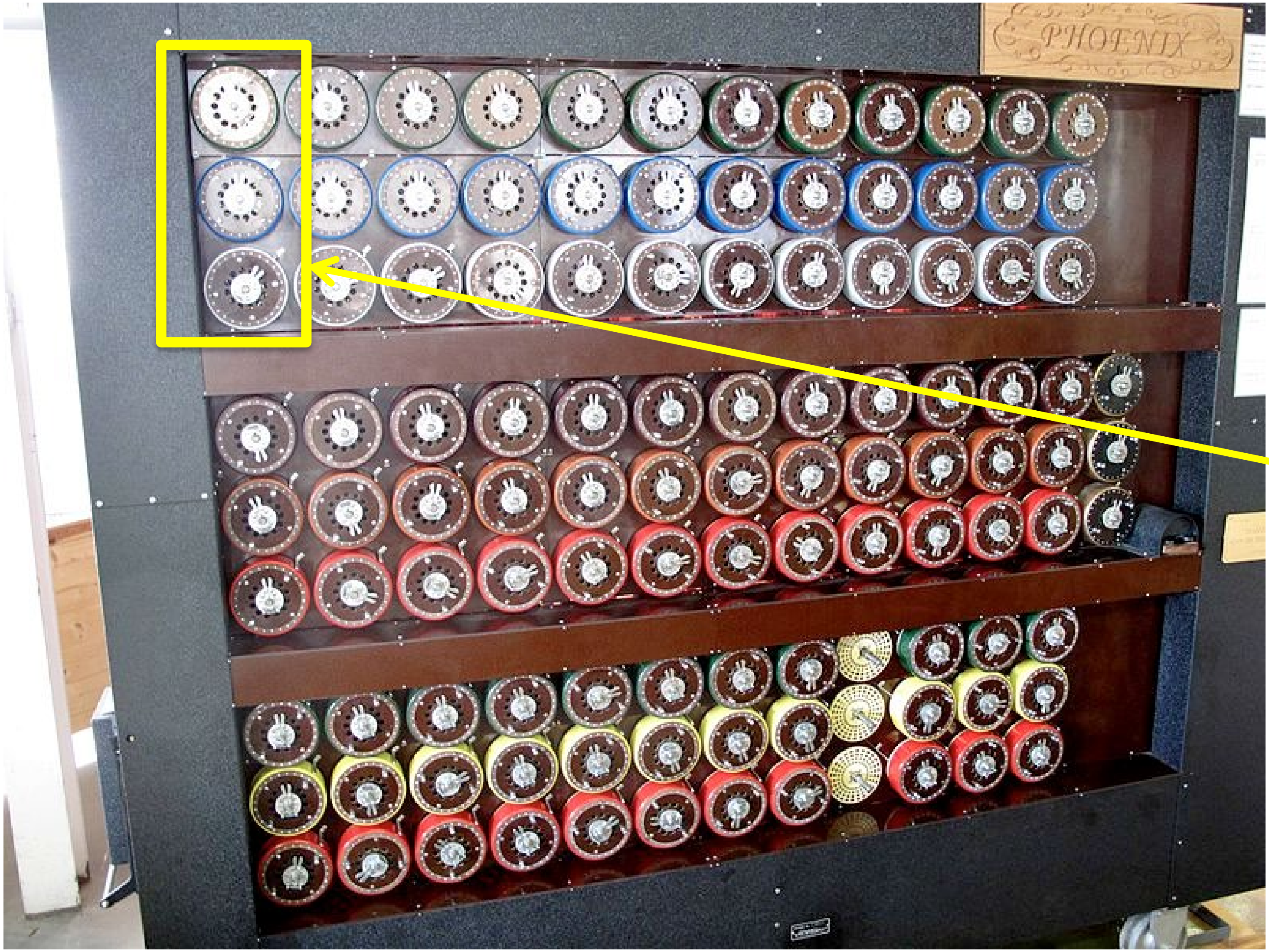
Lampboard
(ciphertext)

Keyboard
(plaintext)

Plugboard
(secret key)

http://www.enigma-replica.com/Glens_Enigma.JPG

Working Rebuilt Bombe at Bletchley Park Museum



Simulates the 3 rotors in one Enigma machine

http://en.wikipedia.org/wiki/Cryptanalysis_of_the_Enigma#Crib-based_decryption

Modern Ciphers

DES (Data Encryption Standard):

- 1977: DES, 56 bits

  (During cold war, cryptography, in particular DES was considered as "**munition**", and subjected to export control. Currently, export of certain cryptography products is still controlled by US.

  Read the crypto law survey's section on Singapore at http://www.cryptolaw.org/cls2.htm)
- 1998: A DES key broken in 56 hours
- Triple DES (112 bits) is still in used

AES (Advanced Encryption Standard):

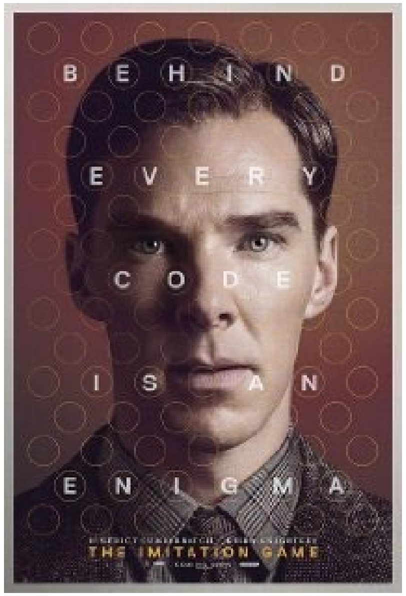
- 2001: NIST. 128, 192, 256 bits

Modern Ciphers

RC4:

- 1987: Designed by Ron Rivest (RSA Security), initially a trade secret
- 1994: Algorithm leaked in
- 1999: Used in widely popular **WEP** (for WiFi);
  WEP implementation has 40 or 104-bit key
- 2001: A weakness in how WEP adopts RC4 is published by Fluhrer, Mantin, Shamir
- 2005: A group from FBI demonstrated the attack
- Afterward: Industry switched to WPA2
  (with WPA as an intermediate solution)

Movie About Encryption



"*The Imitation Game*":

During World War II, mathematician **Alan Turing** tries to crack Enigma with help from fellow mathematicians (http://www.imdb.com/title/tt2084970/)

"*U-571*":

A fictional plot on how Enigma was captured. The Actual event was U-110.

Sample Tutorial Questions

Question:

Bob encrypted a video file using Winzip, which employs the 256-bit key AES.

He choose a 6-digit number as password.

Winzip generated the 256-bit key from the 6-digit password using a "hash" function, say SHA1.

Alice obtained the ciphertext.

Alice also knew that Bob used a 6-digit password.

Given a "guess" of the 256-bit key, Alice can determine whether the key can successfully decrypted the file.

How many guesses Alice really needed to make in order to get the video?

Summary & Takeaways

- Encryption are designed for confidentiality (only!)
- All classical ciphers except the One-Time Pad are broken
- The One-Time Pad has a perfect secrecy, but it’s insufficient to provision a secure channel: *see also Tutorial 2*
- Stream cipher simulates the One-Time Pad
- Block ciphers and modes-of-operation
- Quantifying the security of a cipher by exhaustive search: depends on its key length
- Various pitfalls in using encryption: wrong mode, wrong random sources, mishandling of IV, side-channel attack, …