SoC Mock E-exam Sessions: A Reminder

- Mock e-exam sessions during **Week 5 and the recess week**
- You should register for one **1-hour mock exam session** to have a dry run of the e-exam proctoring process (e.g., setting up Zoom camera, using Zoom chat to communicate with exam invigilator)
- Please check your **NUS email mailbox** and submit the indicated online form by **August 31 (Monday, 5pm)**
- To find out how to prepare for the mock e-exam, please visit https://mysoc.nus.edu.sg/academic/mock-e-exams-for-students/

LumiNUS Cryptanalysis Challenge on Substitution Cipher

- The substitution table used:

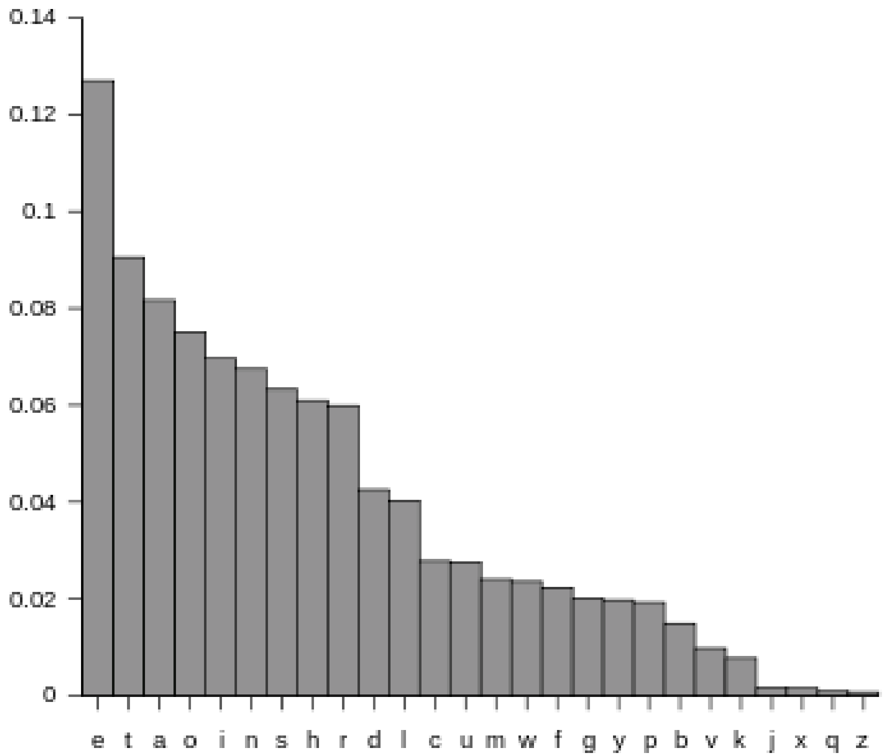
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	_
p	r	k	e	x	j	n	b	c	z	o	f	w	a	_	i	v	t	u	m	d	q	s	h	g	y	l

- The plaintext:

```
cryptography_or_cryptology_is_the_practice_and_study
_of_techniques_for_secure_communication_in_the_prese
nce_of_third_parties_called_adversaries_more_general
ly_cryptography_is_about_constructing_and_analyzing_
protocols_that_prevent_third_parties_or_the_public_f
rom_reading_private_messages_ ...
cryptography_also_plays_a_major_role_in_digital_righ
ts_management_and_copyright_infringement_of_digital_
media_source_wikipedia
```

Some Useful Heuristics

- The most-frequently occurring characters in ciphertext?
  - l (406×) _ (space) , x (274×) ← e, m (213x) ← t,
    p (193×) ← a, _ (186×) ← o: c (185×) ← i:
    match the 5 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**, **it**, **is**, **do**, **on**, **in**, **at**, **of**, **or**, **an**, he, …
- Trigraphs:
  - **the**, **and**, **for**, **has**, **one**, get, not, can, man, men, …
- More and more guessable words

Testing Rounds: After Some Character Mappings

Based on the *top 6 most-frequently occurring single characters*, and some common English *digraphs* & *trigraphs*:

```
$tr 'lxmp_cotiubjw' '_etaoiorishfm' < ciphertext.txt
krgitonraihg_or_krgitofong_is_the_iraktike_aae_stdeg_of_tekhaivdes_for_sekdre_kommdaikatioa_ia_the_ireseake_of
_thire_iarties_kaffee_aeqersaries_more_neaeraffg_krgitonraihg_is_arodt_koastrdktian_aae_aaafgyian_irotokofs_th
at_ireqeat_thire_iarties_or_the_idrfik_from_reaeian_iriqate_messanes_qariods_asiekts_ia_iaformatioa_sekdritg_s
dkh_as_eata_koafieeatiafitg_eata_iatenritg_adtheatikatioa_aae_aoa_reideiatioa_are_keatraf_to_moeera_krgitonrai
hg_moeera_krgitonraihg_ehists_at_the_iatersektioa_of_the_eiskiifiaes_of_mathematiks_komidter_skieake_efektrika
f_eaniaeerian_kommdaikatioa_skieake_aae_ihgsiks_aiifikatioas_of_krgitonraihg_iakfdee_efektroaik_kommerke_khii_
rasee_iagmeat_kares_einitaf_kdrreakies_komidter_iasssores_aae_mifitarg_kommdaikatioas_krgitonraihg_irior_to_th
e_moeera_ane_sas_effektiqefg_sgaoagmods_sith_eakrgitioa_the_koaqersioa_of_iaformatioa_from_a_reaearfe_state_to
_aiiareat_aoasease_the_oriniaator_of_aa_eakrgitee_messane_shares_the_eekoeian_tekhaivde_oafg_sith_iateaeee_rek
iiieats_to_irekfdee_akkess_from_aeqersaries_the_krgitonraihg_fiteratdre_oftea_dses_the_aames_afike_for_the_sea
eer_ror_for_the_iateaeee_rekiiieat_aae_eqe_for_the_aeqersarg_siake_the_eeqefoimeat_of_rotor_kiiher_makhiaes_ia
_sorfe_sar_oae_aae_the_aeqeat_of_komidters_ia_sorfe_sar_tso_the_methoes_dsee_to_karrg_odt_krgitofong_haqe_reko
me_iakreasianfg_komifeh_aae_its_aiifikatioa_more_sieesireae_moeera_krgitonraihg_is_heaqifg_rasee_oa_mathematik
af_theorg_aae_komidter_skieake_iraktike_krgitonraihik_afnorithms_are_eesinaee_arodae_komidtatioaaf_hareaess_as
sdmitioas_maoian_sdkh_afnorithms_hare_to_rreao_ia_iraktike_rg_aag_aeqersarg_it_is_theoretikaffg_iossirfe_to_rr
eao_sdkh_a_sgstem_rdt_it_is_iafeasirfe_to_eo_so_rg_aag_oaosa_iraktikaf_meaas_these_skhemes_are_therefore_terme
e_komidtatioaaffg_sekdre_theoretikaf_aeqaakes_for_ehamifes_imiroqemeats_ia_iatener_faktoriyatioa_afnorithms_aa
e_faster_komidtian_tekhaofong_revdire_these_sofdtioas_to_re_koatiadaffg_aeaitee_there_ehist_iaformatioa_theore
tikaffg_sekdre_skhemes_that_iroqarfg_kaaaot_re_rrooea_eqea_sith_dafimitee_komidtian_ioser_aa_ehamife_is_the_oa
e_time_iae_rdt_these_skhemes_are_more_eiffikdft_to_dse_ia_iraktike_thaa_the_rest_theoretikaffg_rreaoarfe_rdt_k
omidtatioaaffg_sekdre_mekhaaisms_the_nrosth_of_krgitonraihik_tekhaofong_has_raisee_a_admrer_of_fenaf_issdes_ia
_the_iaformatioa_ane_krgitonraihgs_ioteatiaf_for_dse_as_a_toof_for_esiioaane_aae_seeitioa_has_fee_maag_noqeram
eats_to_kfassifg_it_as_a_seaioa_aae_to_fimit_or_eqea_irohirit_its_dse_aae_ehiort_ia_some_zdriseiktioas_shere_t
he_dse_of_krgitonraihg_is_fenaf_fass_iermit_iaqestinators_to_komief_the_eiskfosdre_of_eakrgitioa_oegs_for_eokd
meats_refeqaat_to_aa_iaqestinatioa_krgitonraihg_afso_ifags_a_mazor_rofe_ia_einitaf_rinhts_maaanemeat_aae_koigr
inht_iafrianemeat_of_einitaf_meeia_sodrke_sioiieeia
```

Testing Rounds: After Some More Character Mappings

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

```
$tr 'lxmp_cotiubjweikgn' '_etaoiorishfmdpcyg' < ciphertext.txt
cryptography_or_cryptofogy_is_the_practice_aad_stddy_of_techaivdes_for_secdre_commdaicatioa_ia_the_preseace_of
_third_parties_caffed_adqersaries_more_geaeraffy_cryptography_is_arodt_coastrdctiag_aad_aaafyyiag_protocofs_th
at_preqeat_third_parties_or_the_pdrfic_from_readiag_priqate_messages_qariods_aspects_ia_iaformatioa_secdrity_s
dch_as_data_coafideatiafity_data_iategrity_adtheaticatioa_aad_aoa_repddiatioa_are_ceatraf_to_modera_cryptograp
hy_modera_cryptography_ehists_at_the_iatersectioa_of_the_discipfiaes_of_mathematics_compdter_scieace_efectrica
f_eagiaeeriag_commdaicatioa_scieace_aad_physics_appficatioas_of_cryptography_iacfdde_efectroaic_commerce_chip_
rased_paymeat_cards_digitaf_cdrreacies_compdter_passsords_aad_mifitary_commdaicatioas_cryptography_prior_to_th
e_modera_age_sas_effectiqefy_syaoaymods_sith_eacryptioa_the_coaqersioa_of_iaformatioa_from_a_readarfe_state_to
_appareat_aoasease_the_origiaator_of_aa_eacrypted_message_shares_the_decodiag_techaivde_oafy_sith_iateaded_rec
ipieats_to_precfdde_access_from_adqersaries_the_cryptography_fiteratdre_oftea_dses_the_aames_afice_for_the_sea
der_ror_for_the_iateaded_recipieat_aad_eqe_for_the_adqersary_siace_the_deqefopmeat_of_rotor_cipher_machiaes_ia
_sorfd_sar_oae_aad_the_adqeat_of_compdters_ia_sorfd_sar_tso_the_methods_dsed_to_carry_odt_cryptofogy_haqe_reco
me_iacreasiagfy_compfeh_aad_its_appficatioa_more_sidespread_modera_cryptography_is_heaqify_rased_oa_mathematic
af_theory_aad_compdter_scieace_practice_cryptographic_afgorithms_are_desigaed_arodad_compdtatioaaf_hardaess_as
sdmptioas_maoiag_sdch_afgorithms_hard_to_rreao_ia_practice_ry_aay_adqersary_it_is_theoreticaffy_possirfe_to_rr
eao_sdch_a_system_rdt_it_is_iafeasirfe_to_do_so_ry_aay_oaosa_practicaf_meaas_these_schemes_are_therefore_terme
d_compdtatioaaffy_secdre_theoreticaf_adqaaces_for_ehampfes_improqemeats_ia_iateger_factoriyatioa_afgorithms_aa
d_faster_compdtiag_techaofogy_revdire_these_sofdtioas_to_re_coatiadaffy_adapted_there_ehist_iaformatioa_theore
ticaffy_secdre_schemes_that_proqarfy_caaaot_re_rrooea_eqea_sith_dafimited_compdtiag_poser_aa_ehampfe_is_the_oa
e_time_pad_rdt_these_schemes_are_more_difficdft_to_dse_ia_practice_thaa_the_rest_theoreticaffy_rreaoarfe_rdt_c
ompdtatioaaffy_secdre_mechaaisms_the_grosth_of_cryptographic_techaofogy_has_raised_a_admrer_of_fegaf_issdes_ia
_the_iaformatioa_age_cryptographys_poteatiaf_for_dse_as_a_toof_for_espioaage_aad_seditioa_has_fed_maay_goqeram
eats_to_cfassify_it_as_a_seapoa_aad_to_fimit_or_eqea_prohirit_its_dse_aad_ehport_ia_some_zdrisdictioas_shere_t
he_dse_of_cryptography_is_fegaf_fass_permit_iaqestigators_to_compef_the_discfosdre_of_eacryptioa_oeys_for_docd
meats_refeqaat_to_aa_iaqestigatioa_cryptography_afso_pfays_a_mazor_rofe_ia_digitaf_rights_maaagemeat_aad_copyr
ight_iafriagemeat_of_digitaf_media_sodrce_sioipedia
```

The partial plaintext already looks pretty readable and crackable!

Testing Rounds: Completed Mapping

The complete plaintext:

```
$tr 'prkexjnbczofwa_ivtumdqshgyl' 'a-z_' < ciphertext.txt
cryptography_or_cryptology_is_the_practice_and_study_of_techniques_for_secure_communication_in_the_presence_of
_third_parties_called_adversaries_more_generally_cryptography_is_about_constructing_and_analyzing_protocols_th
at_prevent_third_parties_or_the_public_from_reading_private_messages_various_aspects_in_information_security_s
uch_as_data_confidentiality_data_integrity_authentication_and_non_repudiation_are_central_to_modern_cryptograp
hy_modern_cryptography_exists_at_the_intersection_of_the_disciplines_of_mathematics_computer_science_electrica
l_engineering_communication_science_and_physics_applications_of_cryptography_include_electronic_commerce_chip_
based_payment_cards_digital_currencies_computer_passwords_and_military_communications_cryptography_prior_to_th
e_modern_age_was_effectively_synonymous_with_encryption_the_conversion_of_information_from_a_readable_state_to
_apparent_nonsense_the_originator_of_an_encrypted_message_shares_the_decoding_technique_only_with_intended_rec
ipients_to_preclude_access_from_adversaries_the_cryptography_literature_often_uses_the_names_alice_for_the_sen
der_bob_for_the_intended_recipient_and_eve_for_the_adversary_since_the_development_of_rotor_cipher_machines_in
_world_war_one_and_the_advent_of_computers_in_world_war_two_the_methods_used_to_carry_out_cryptology_have_beco
me_increasingly_complex_and_its_application_more_widespread_modern_cryptography_is_heavily_based_on_mathematic
al_theory_and_computer_science_practice_cryptographic_algorithms_are_designed_around_computational_hardness_as
sumptions_making_such_algorithms_hard_to_break_in_practice_by_any_adversary_it_is_theoretically_possible_to_br
eak_such_a_system_but_it_is_infeasible_to_do_so_by_any_known_practical_means_these_schemes_are_therefore_terme
d_computationally_secure_theoretical_advances_for_examples_improvements_in_integer_factorization_algorithms_an
d_faster_computing_technology_require_these_solutions_to_be_continually_adapted_there_exist_information_theore
tically_secure_schemes_that_provably_cannot_be_broken_even_with_unlimited_computing_power_an_example_is_the_on
e_time_pad_but_these_schemes_are_more_difficult_to_use_in_practice_than_the_best_theoretically_breakable_but_c
omputationally_secure_mechanisms_the_growth_of_cryptographic_technology_has_raised_a_number_of_legal_issues_in
_the_information_age_cryptographys_potential_for_use_as_a_tool_for_espionage_and_sedition_has_led_many_governm
ents_to_classify_it_as_a_weapon_and_to_limit_or_even_prohibit_its_use_and_export_in_some_jurisdictions_where_t
he_use_of_cryptography_is_legal_laws_permit_investigators_to_compel_the_disclosure_of_encryption_keys_for_docu
ments_relevant_to_an_investigation_cryptography_also_plays_a_major_role_in_digital_rights_management_and_copyr
ight_infringement_of_digital_media_source_wikipedia
```

Security of Vigenere Cipher (Corrected)

- Vigenere cipher is an improvement over shift cipher
- Different occurrences of the same letter can have **different** mapped letters!

- Is it however secure against known-plaintext attack?
  → this is easy to answer: ~~*yes*~~ *no*
- Is it secure against ciphertext-only attack??
  → trickier to answer: need to find *a good attack technique*

- Suppose we know ***k*** = the **length/period** of the keyword
- Observation: all letters of the plaintext whose index is i (mod k), for i=0..k-1, get shifted by the same key character
- Can we use our previous frequency analysis technique?
- Vigenere cipher turns into a monoalphabetic cipher *again*

Security Guarantee: Perfect Secrecy

Attacker's ***prior knowledge*** of the unknown plaintext *m*

- **Perfect security/secrecy** ("absolute security"):
  - Informally: "regardless of any prior information that attackers (with ***unlimited computational power***) has about the plaintext, the ciphertext should leak ***no additional information*** about the plaintext
  - More formally: can be defined in terms of **conditional probability** (i.e. $\Pr[M=m|C=c] = \Pr[M=m]$ for $\forall m \in \boldsymbol{M}$ & $\forall c \in \boldsymbol{C}$ with $\Pr[C=c]>0$)
    → *not* covered in this module
  - **Issue**: the key must be (at least) as long as the message itself
    → impractical in practice
  - *Important questions*:
    - Is it unnecessarily strong for practical usage?
    - Any security notion that is **more relaxed and practical**?

Optional

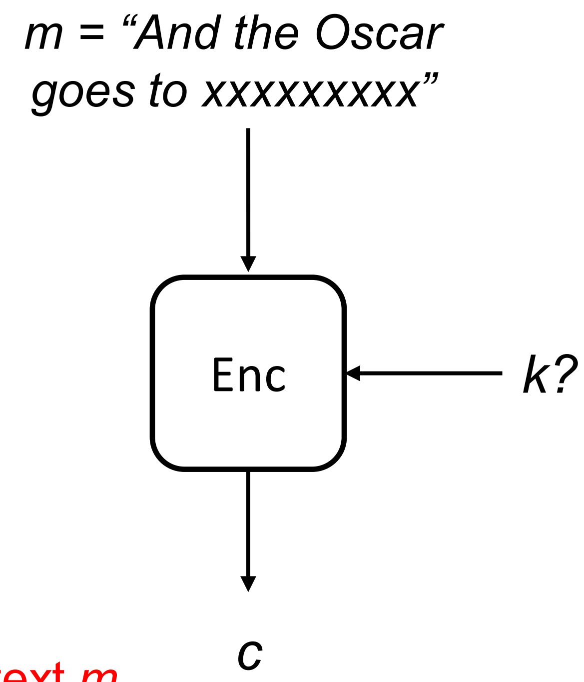
Attacker's ***updated knowledge*** of the unknown plaintext *m* after the attacker had seen the ciphertext *c*

So, the attacker gained **no additional information** at all!

A Good Analogy

Attacker's ***prior knowledge*** of the unknown plaintext *m*

- Let ***c*** be the ciphertext of the Oscar's winner name, encrypted using some secret key
- Bob gathers information from many sources, and believes that the movie *"Alice in Crypto Land"* has **42%** chance of winning
- Now, suppose somehow, Bob obtains *c*.
  With the additional knowledge of *c*,
  can now Bob improve his guess from 42%?
  (Note that any slight improvement would be useful, because Bob plans to bet in some online site)
- Let's assume that Bob have **extremely powerful machine** that can exhaustively search all keys. Can he improve the guess?
  - *If the encryption scheme used is AES, the answer is yes! How?*
  - *if the encryption scheme achieves perfect secrecy, e.g. One-Time Pad, the answer is no!*



Attacker's ***updated knowledge*** of the unknown plaintext *m* after the attacker had seen the ciphertext *c*

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

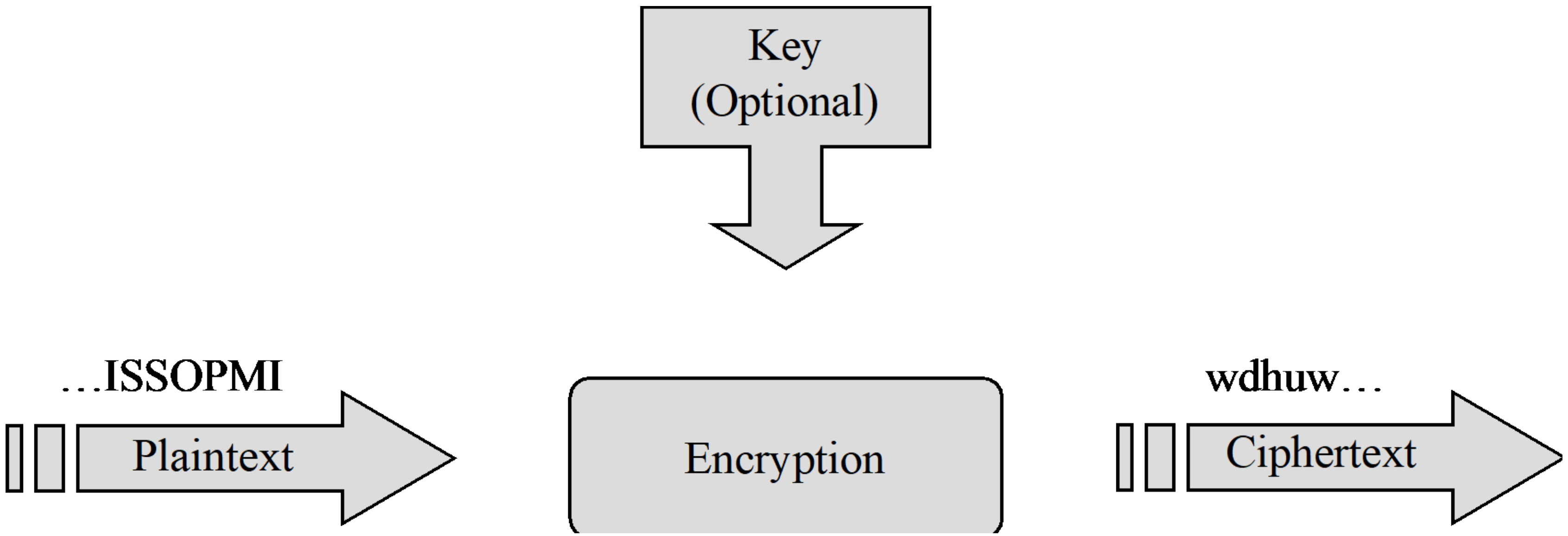
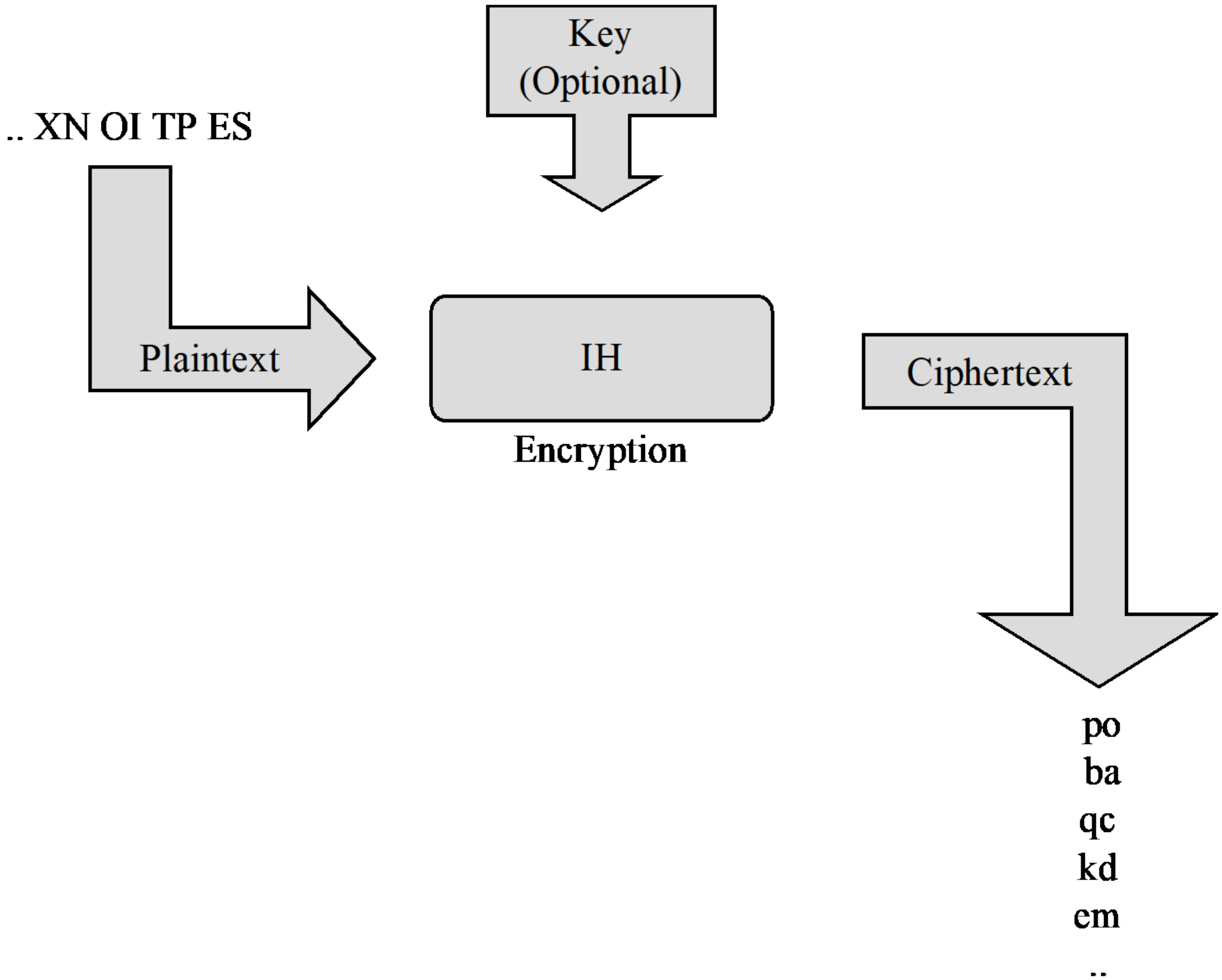
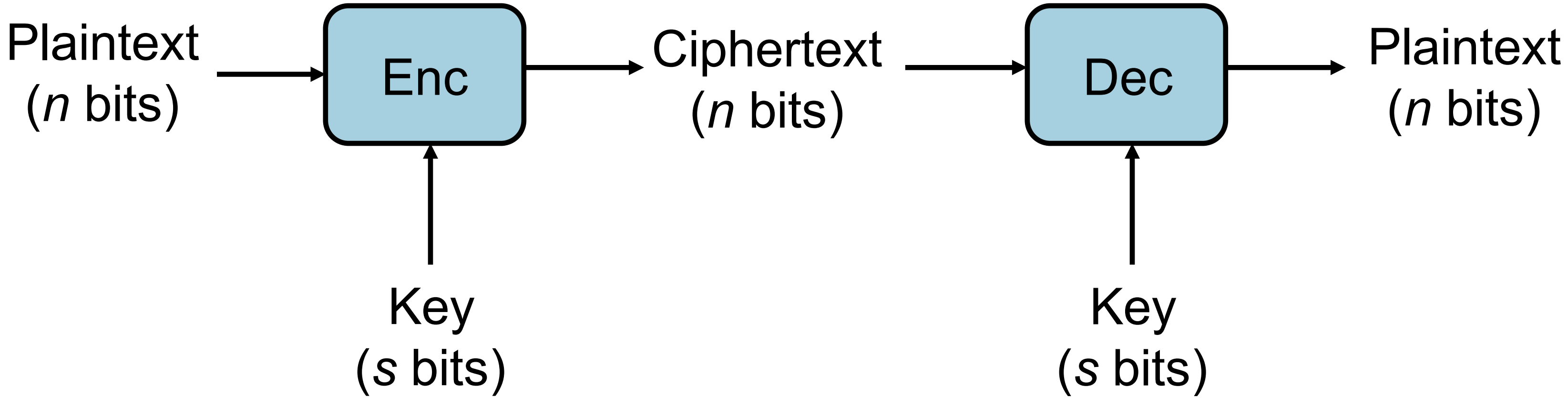


Illustration of a Block Cipher



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): $\boldsymbol{K \times M \to C}$ and **D (Dec)**: $\boldsymbol{K \times C \to M}$
- $\boldsymbol{M = C}$ = $\{0,1\}^n$, with $\boldsymbol{n}$ **= block size**
- $\boldsymbol{K}$ (key space) = $\{0,1\}^s$, with $\boldsymbol{s}$ **= key length/size**



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 slide)
  - AES : n = 128 bits, s = 128, 192, 256 bits
- **The longer** the key is:
  - The more secure the scheme is
  - The slower it is
- Can the block size be **too small** (i.e. <64 bits)?
  See Tutorial 2 for a possible attack

A Block Cipher: Encryption and Decryption Requirements

- 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

- Don't confuse **pseudorandom permutation (PRP)** with:
  - 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

- 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**"

→ ***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 abstraction used for block ciphers, and can still improve our understanding about block ciphers and their requirements.

Security Goal: Indistinguishability

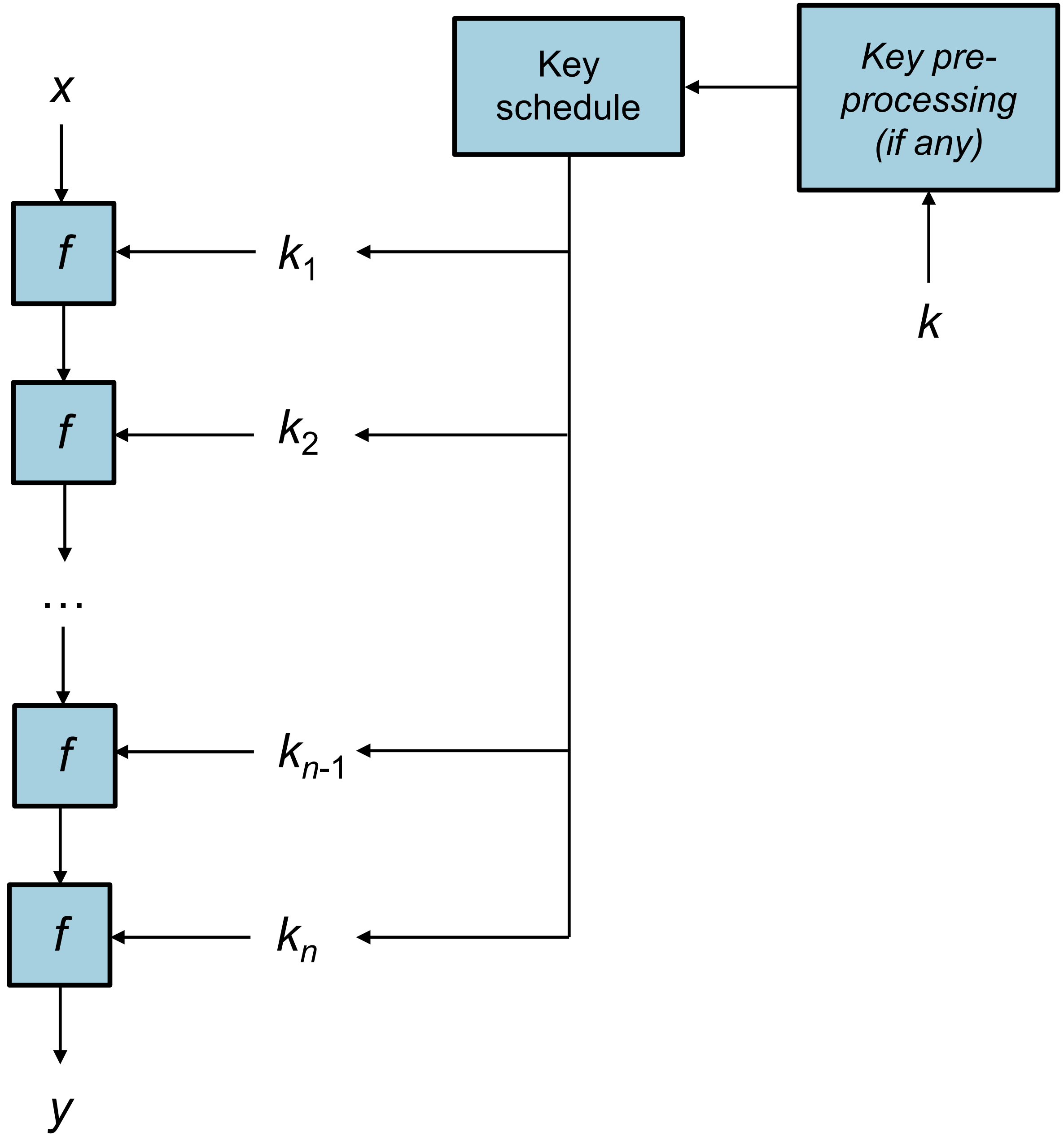
Optional

- Btw, what does "***look random***" really mean?
  Any more formal notion about "ciphertext randomness"?
- ***Indistinguishability***:
  - One security goal of a cipher
  - (Informally) **Ciphertexts should be indistinguishable from random strings**
- **A hypothetical game**: if an attacker **picks 2 plaintexts** and then receives a ciphertext of **1** of the two (**chosen at random**), they shouldn't be able to tell which plaintext was encrypted
- This applies even when the attacker can perform encryption queries with the 2 plaintexts (in Chosen Plaintext Attack)
  but without knowing the secret key used

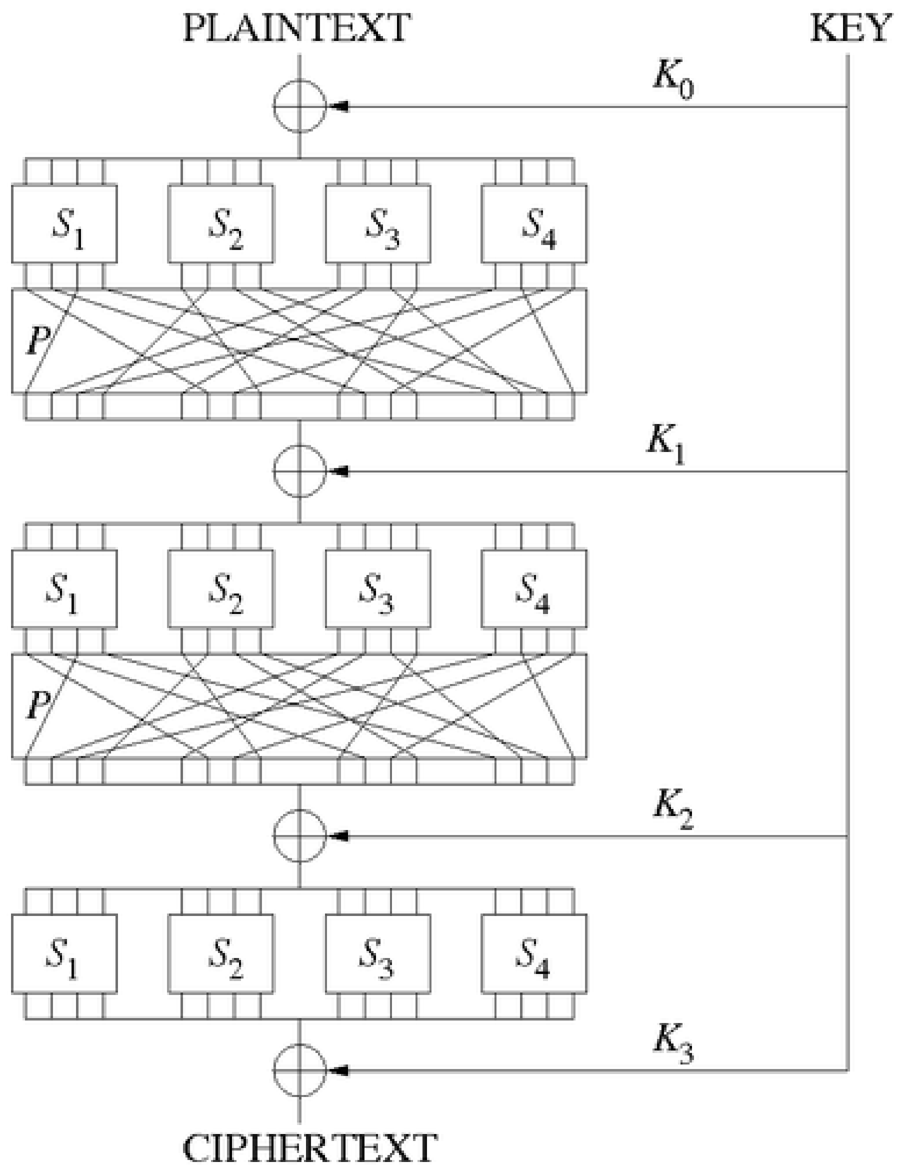
A Block Cipher: How It Typically Works

- 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 (see an illustration in the next slide)
  - Two **main techniques** for each round: **substitution–permutation network** (as in AES) and **Feistel scheme** (as in DES)
- A block cipher's **round**:
  - Simple operations in each round: easy to specify, implement and analyze
  - A **round function** $\boldsymbol{f(x,k)}$
  - The key may have first undergone a **pre-processing**, i.e. key expansion
  - A **key schedule function** produces a sequence of ***round keys (subkeys)*** $\boldsymbol{k_1, k_2, \ldots k_n}$**:** 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

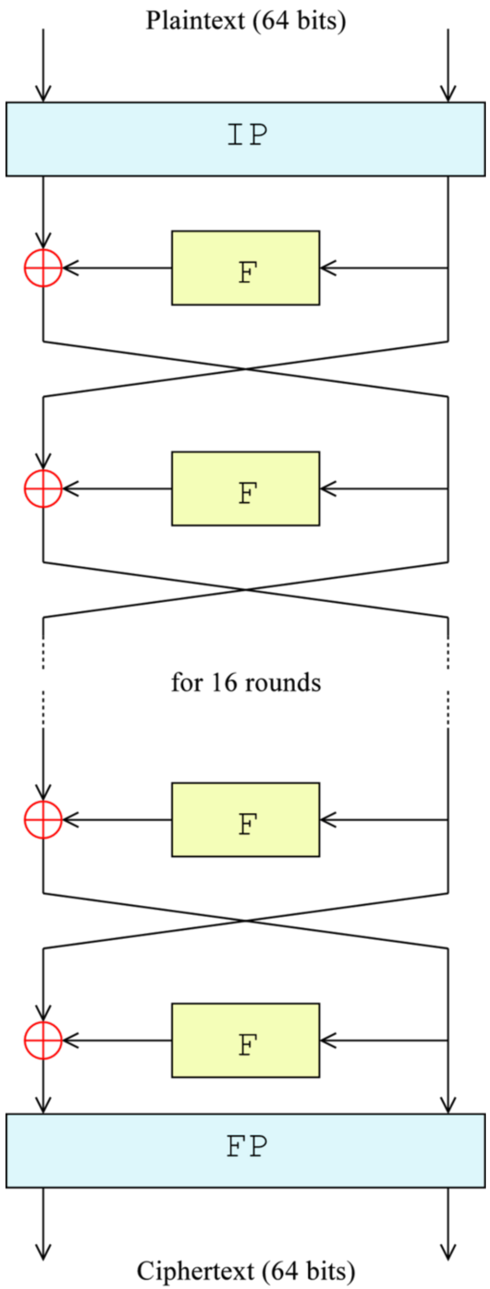
1.5.2 Popular Block Ciphers

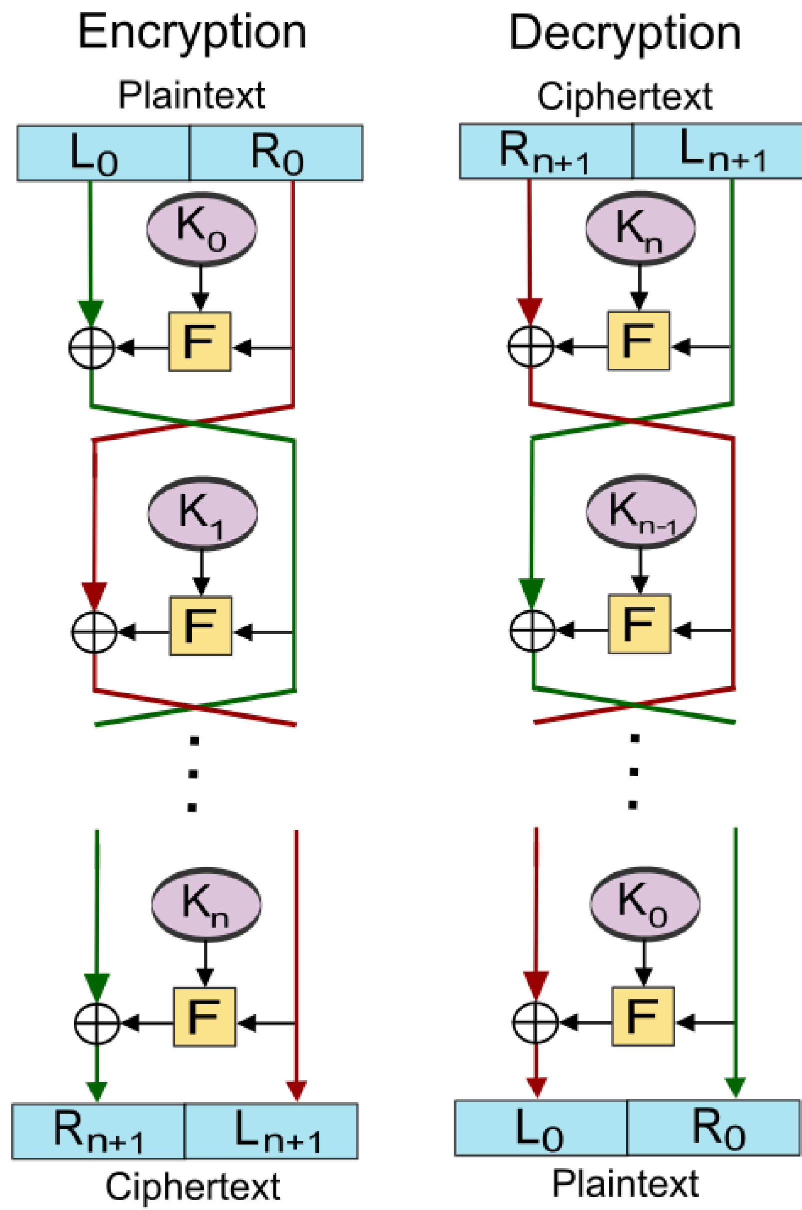
DES (Data Encryption Standard)

- 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**

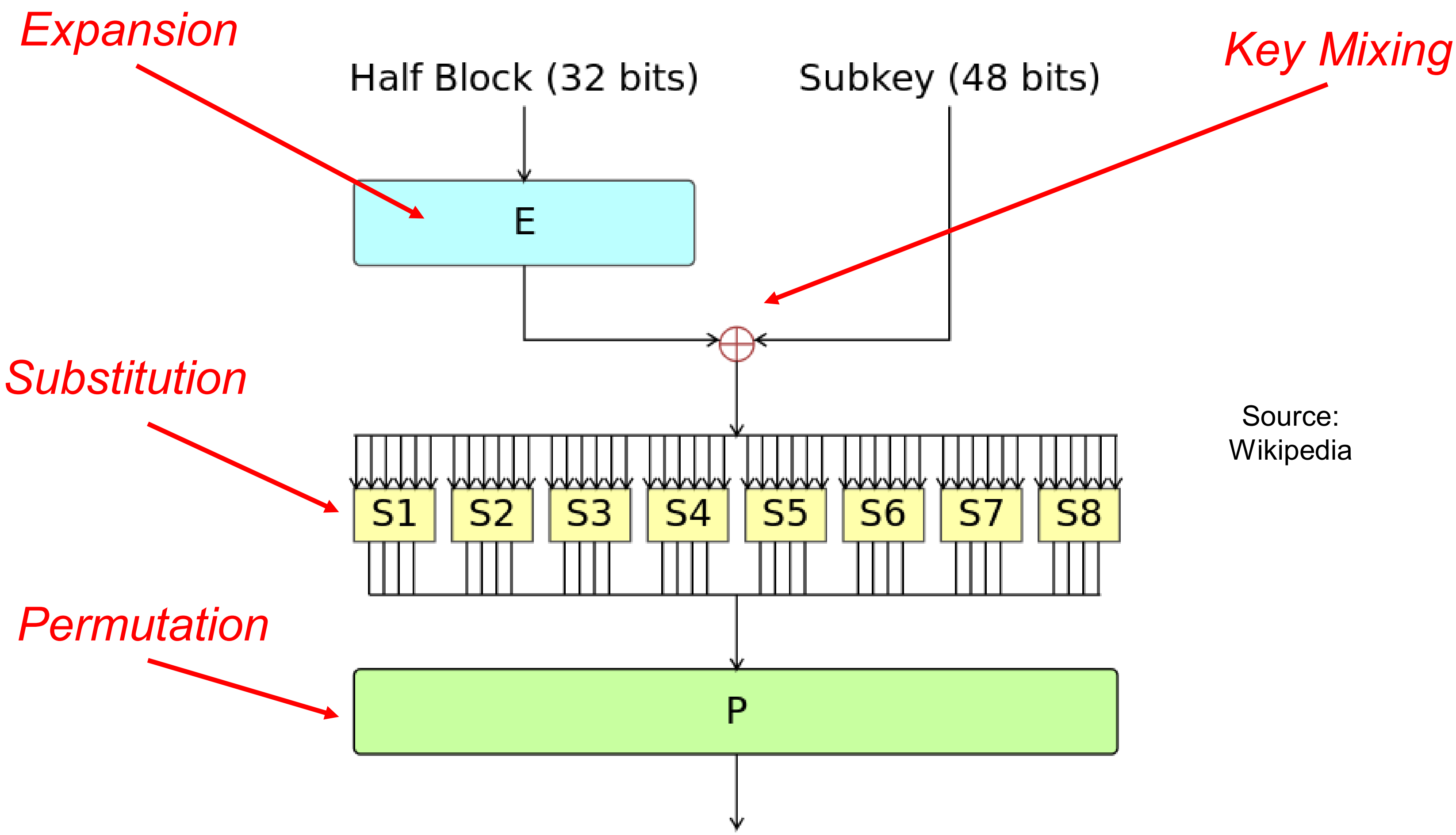
Feistel Network





Source:
Wikipedia

Feistel Function in Each Round



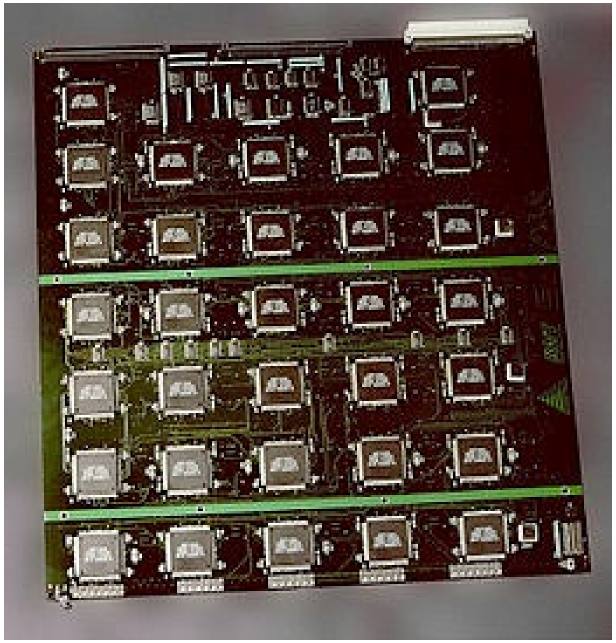
Source:
Wikipedia

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*"):

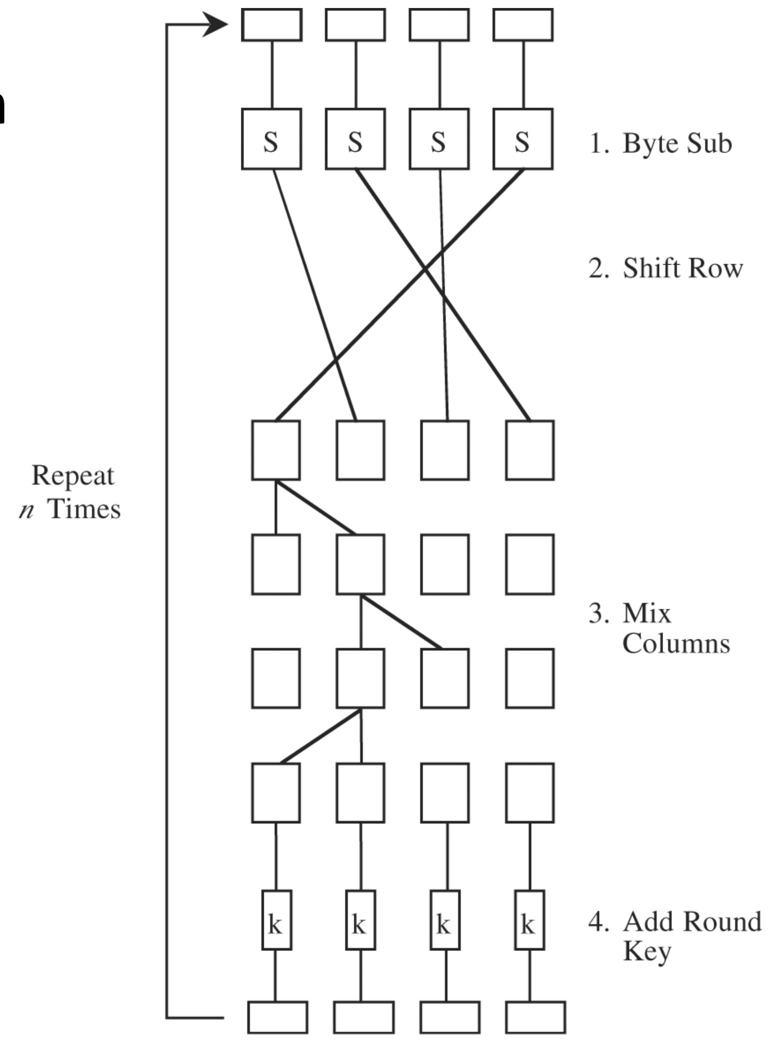


Optional: https://en.wikipedia.org/wiki/EFF_DES_cracker

- 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

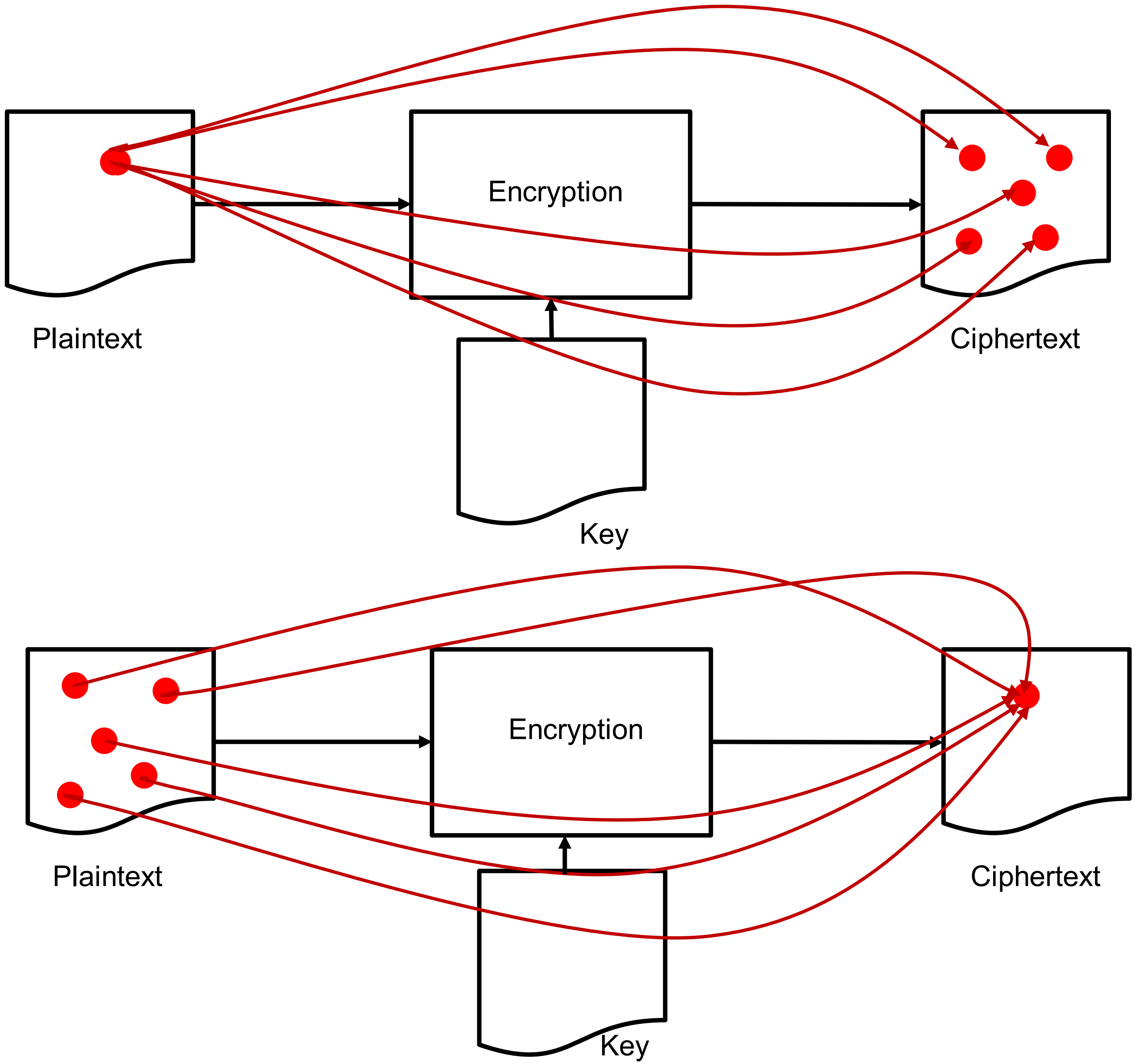
Stream vs Block Ciphers

	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

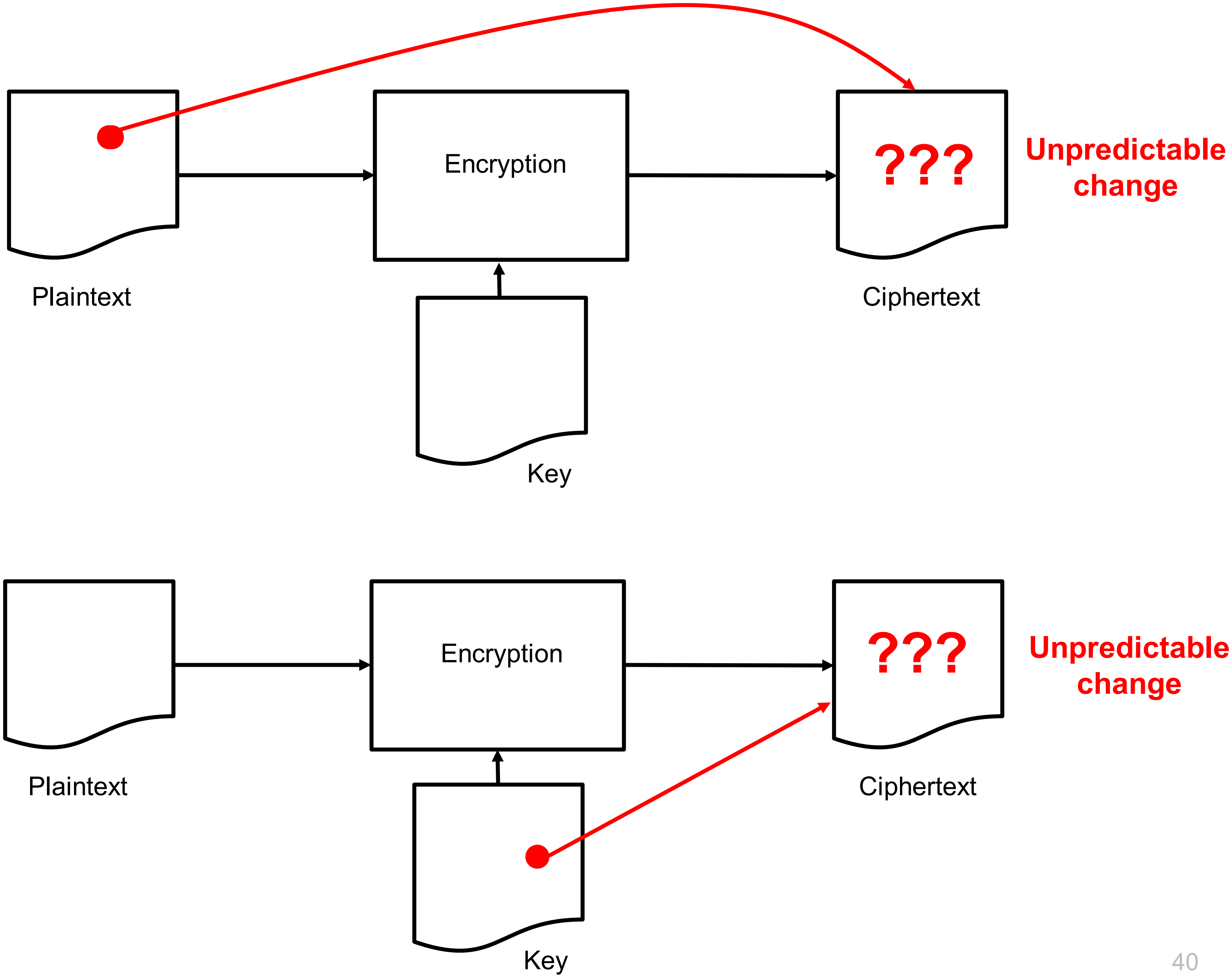
Diffusion Illustrated: Change Effect



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

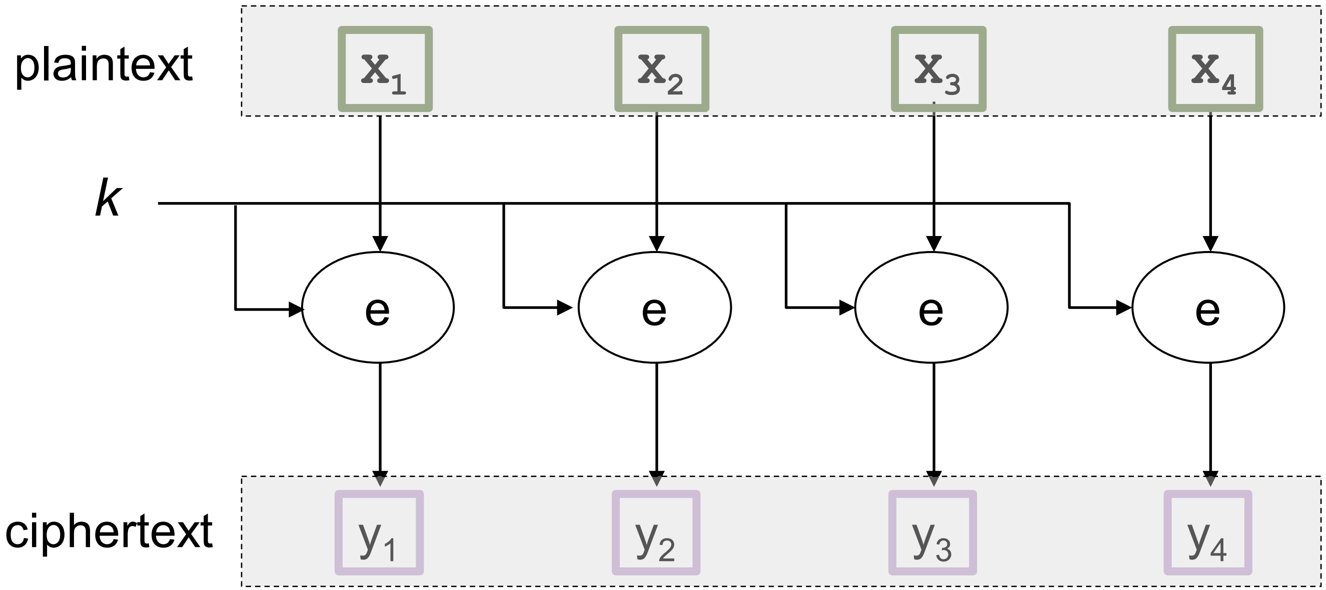
Block Cipher: Modes of Operations

- 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?

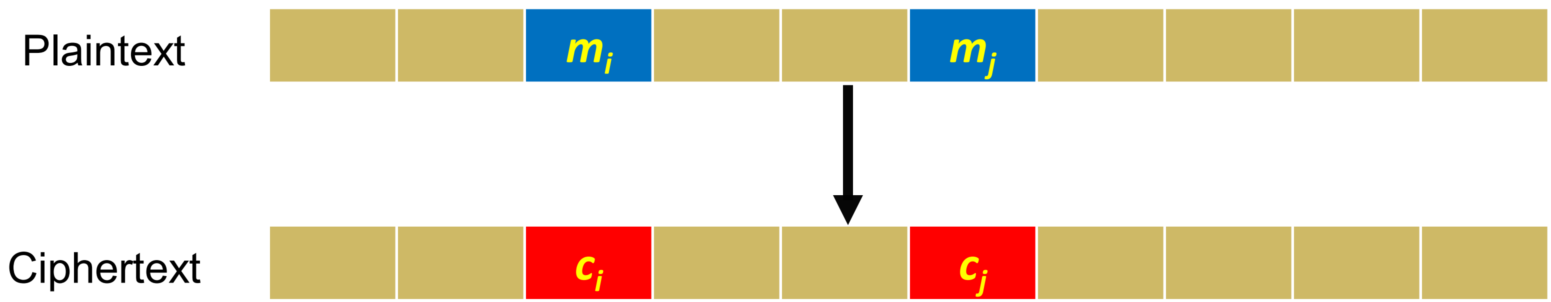
- A ***mode of operation***: a method of encrypting messages of arbitrary size using a block cipher
- Extending encryption from a single block to multiple blocks is however ***not*** straightforward: there are some **security implications** (see later slides)

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

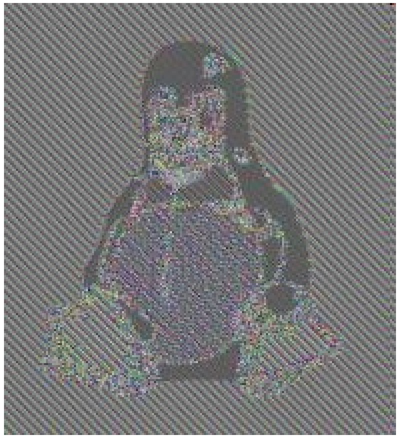


- What if $m_i = m_j$?
- The attacker **can tell** that $c_i = c_j$
- Some information about the plaintext is **leaked**!
- *But, what's the big deal with this??*

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

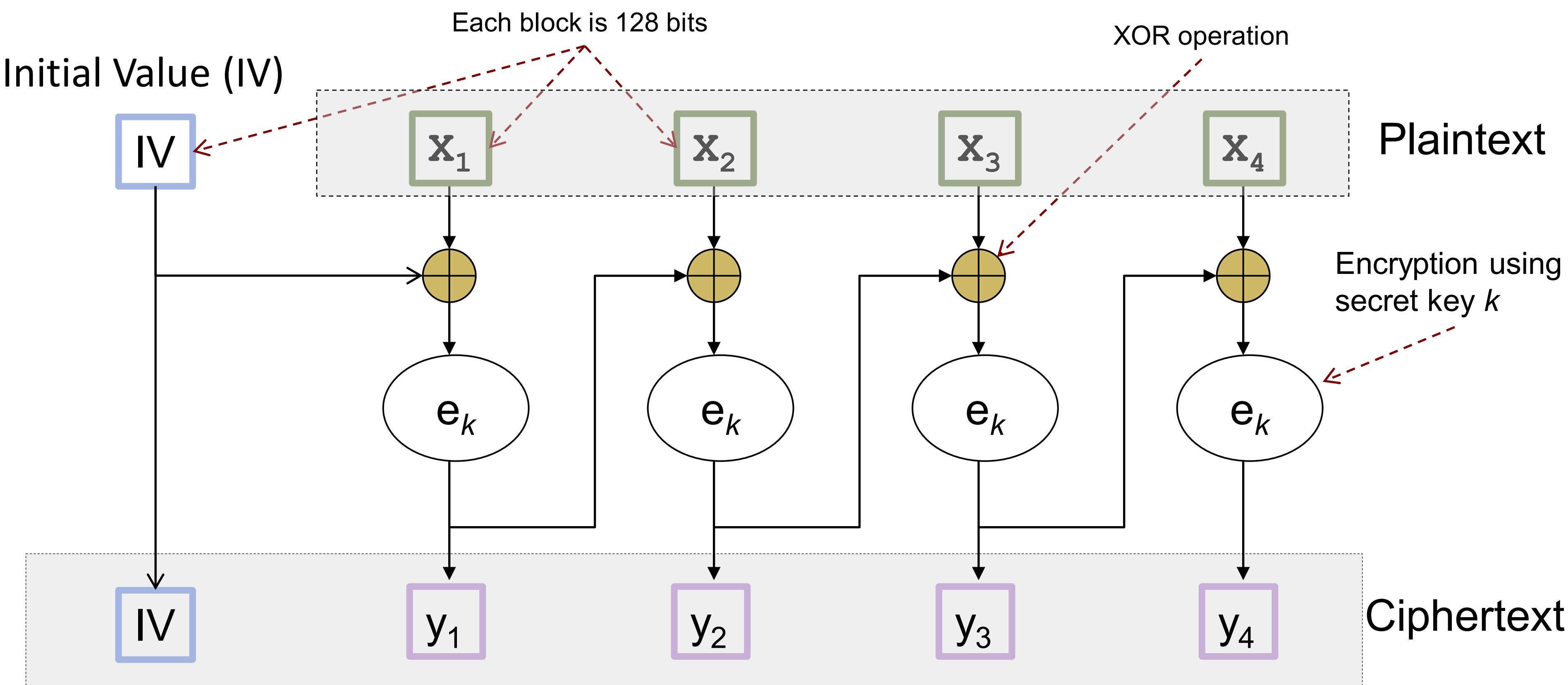
- *Additional mechanisms are required!*

- ***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

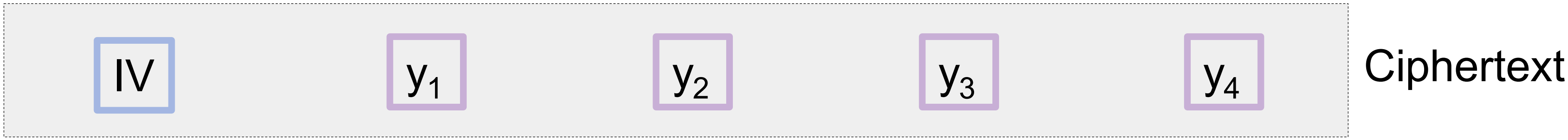
- A ***mode-of-operation*** describes how the blocks are to be "**linked**" so that different blocks at different locations would give different ciphertext, even if all the blocks have the same content
- Popular modes-of-operation:
  Cipher Block Chaining (CBC) and ***CTR (counter)*** modes
- Avoid the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



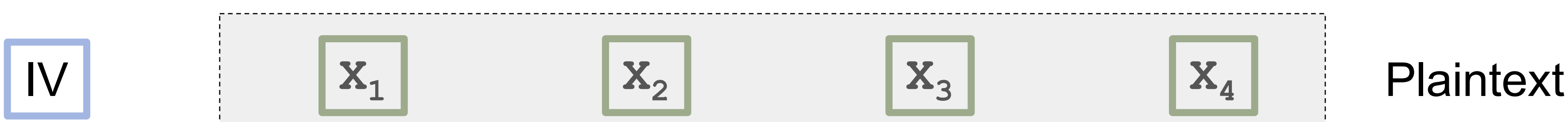
Some questions:

How about the decryption process?

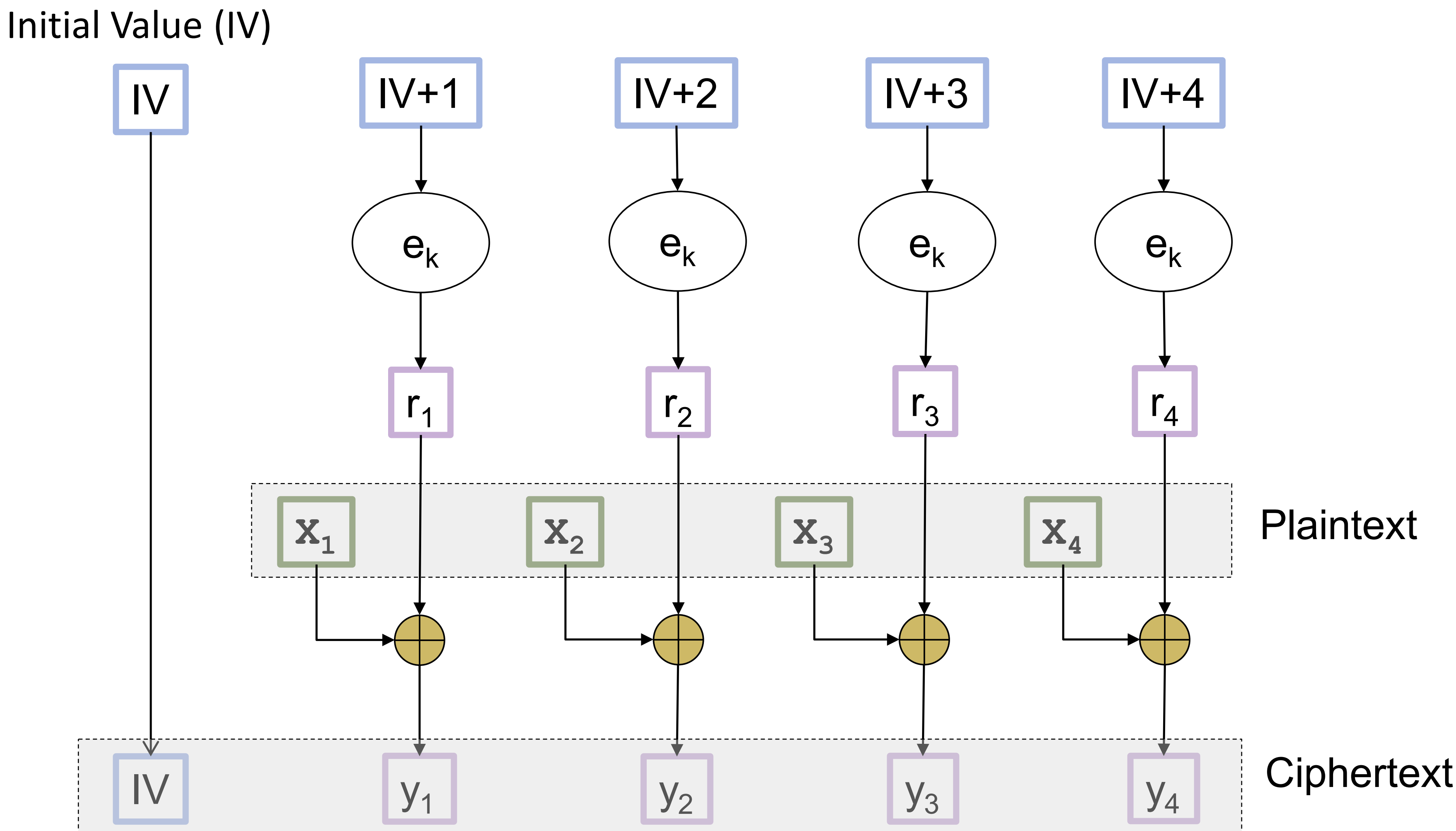
Also, what will happen if a ciphertext block gets corrupted?

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

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)

Triple DES (3DES)

- Remedy: use **triple DES** encryptions
- 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) $E_{k_1}(\mathbf{E}_{k_2}(E_{k_1}(x)))$ or

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

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

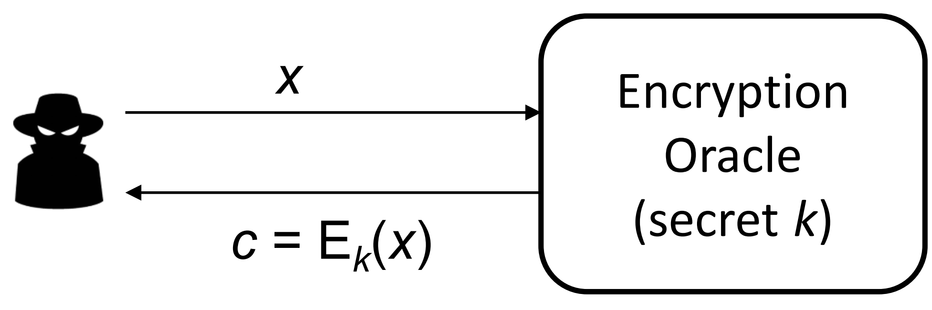
Triple DES (3DES)

- 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

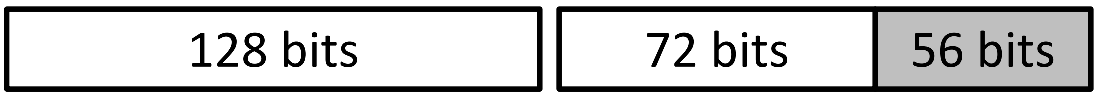


Padding Oracle attack

- 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

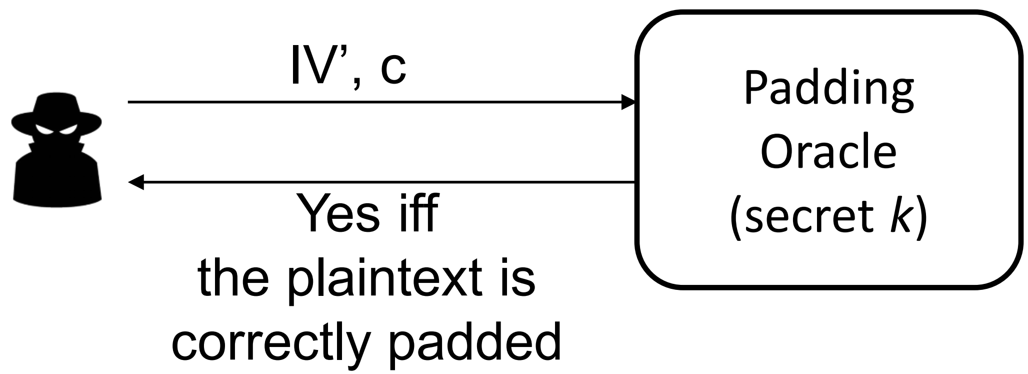
- **AES CBC mode** is ***not*** secure against padding oracle attack (when padding is done with PKCS#7)
- Let us look at this example: the data sent to the Oracle is IV and c
- Attacker has (**IV || c**): 1 block of IV and 1 block of **c**
- For convenience, let us assume that the attacker knows that 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
	?	?	?	?	?			

Padding Oracle Attack on AES CBC Mode

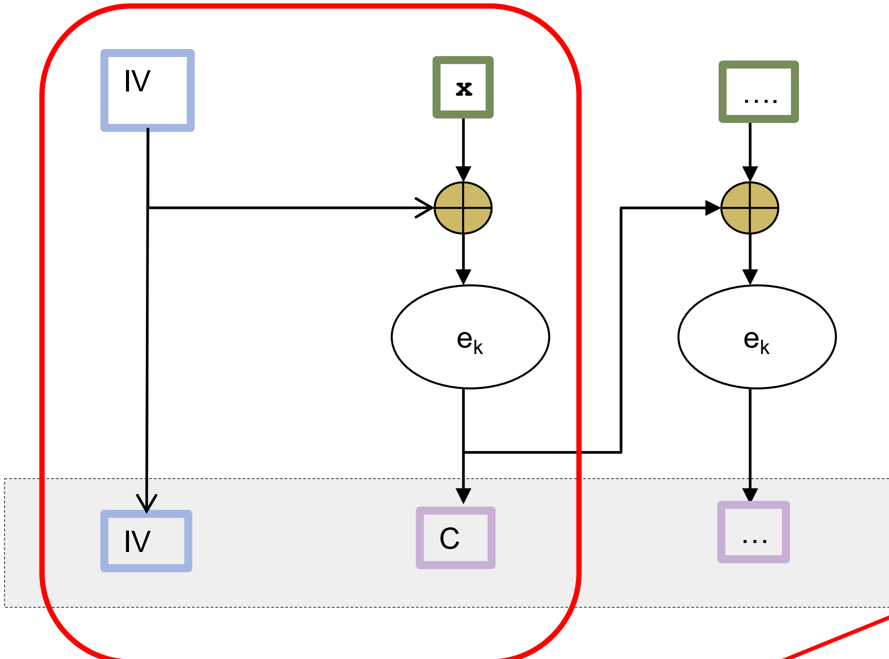
This algorithm outputs the value of $\boldsymbol{x_5}$:

1. For t = 0 to 255 *// in binary 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

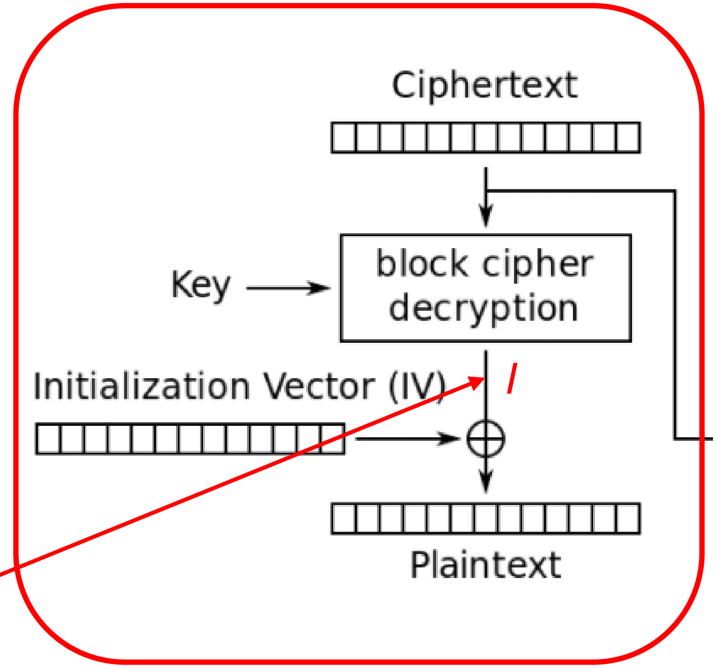


Why Does It Work?

- CBC encryption:



Corresponding CBC decryption (partial):



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

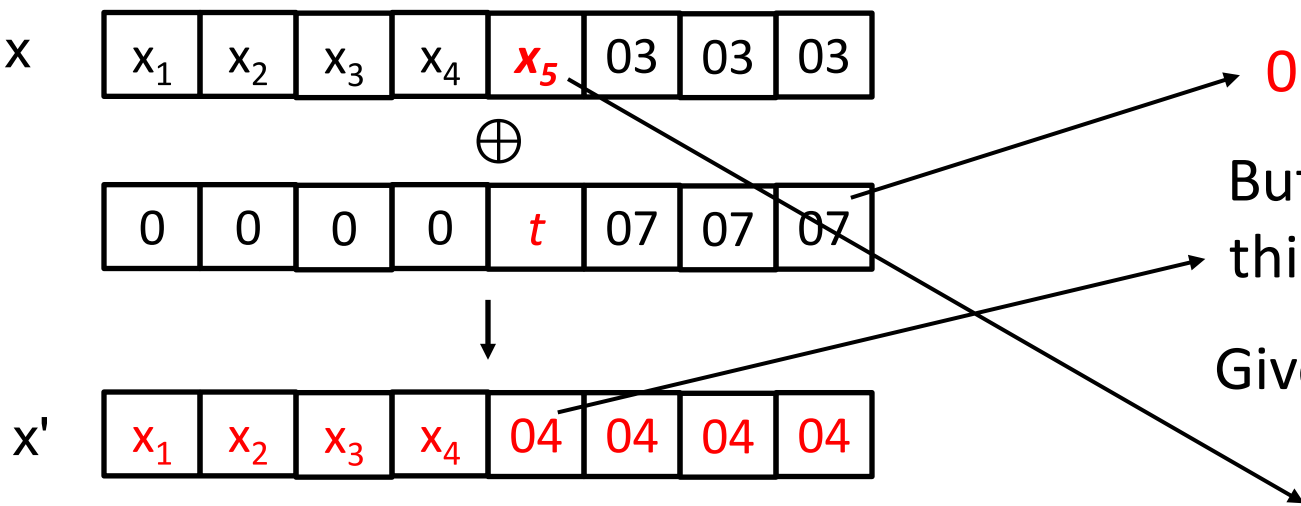
Why Does It Work?

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

$$x' = I \oplus IV'$$
$$= (IV \oplus x) \oplus (IV \oplus \boxed{0\,|\,0\,|\,0\,|\,0\,|\,t\,|\,07\,|\,07\,|\,07})$$
$$= x \oplus \boxed{0\,|\,0\,|\,0\,|\,0\,|\,t\,|\,07\,|\,07\,|\,07}$$

---

x' as derived & known by Padding Oracle only:



Note that:
$07 = 04 \oplus 03$

But the Oracle tells this to the attacker

Given t, the attacker now knows:
$\mathbf{x_5 = 04 \oplus} \boldsymbol{t}$

See also: https://en.wikipedia.org/wiki/Padding_oracle_attack
https://robertheaton.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 an error message
  - If no explicit error message returned, 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~~

TRINE

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???

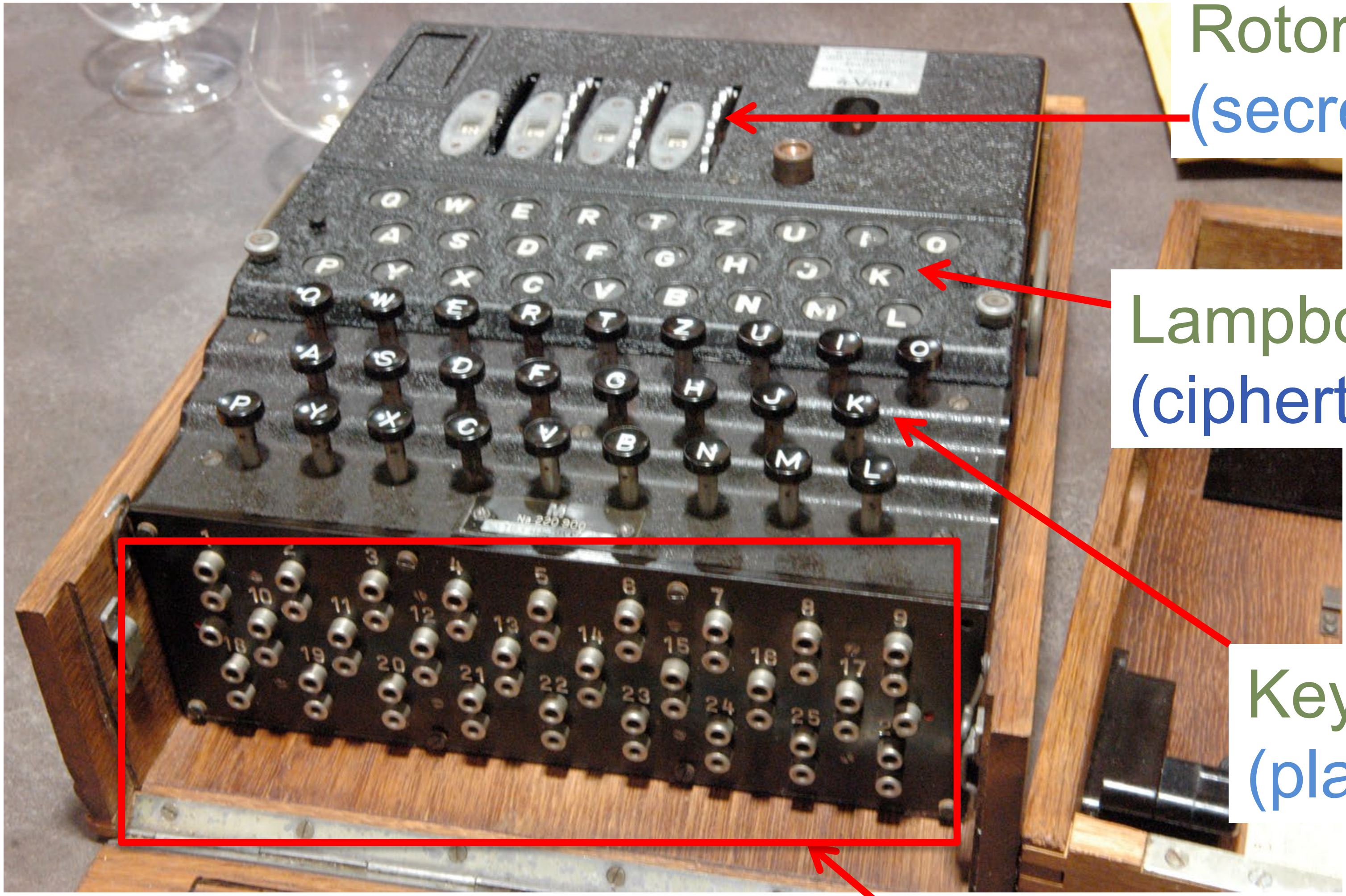
- In general, obscurity can be used as one layer in a ***defense in depth*** strategy
- It could deter or discourage novice attackers, but is ineffective against attackers with high skill and motivation
- The system **must remain secure** even if everything about it, *except its secret key*, becomes 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)



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

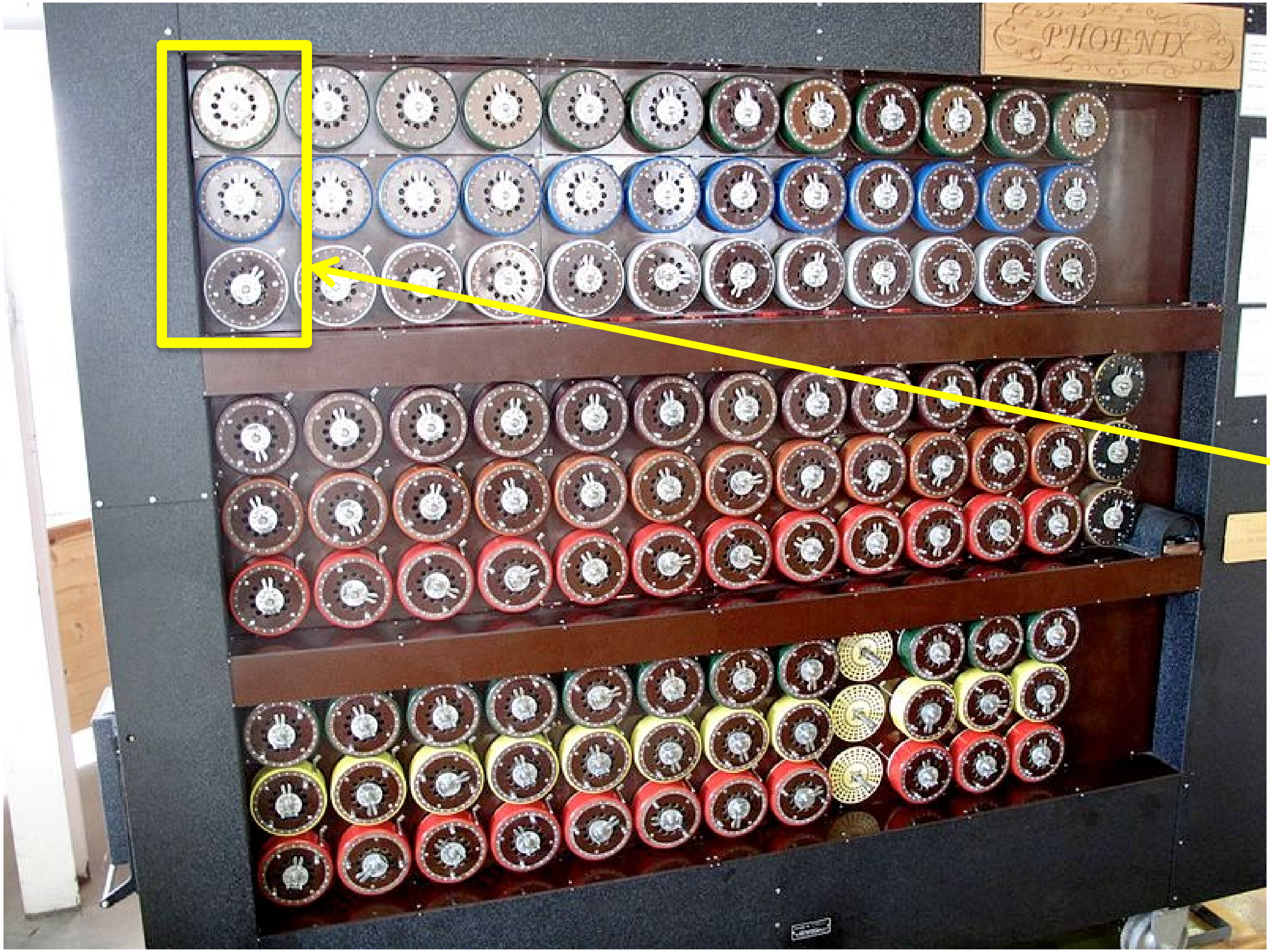
Rotors
(secret key)

Lampboard
(ciphertext)

Keyboard
(plaintext)

Plugboard
(secret key)

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):

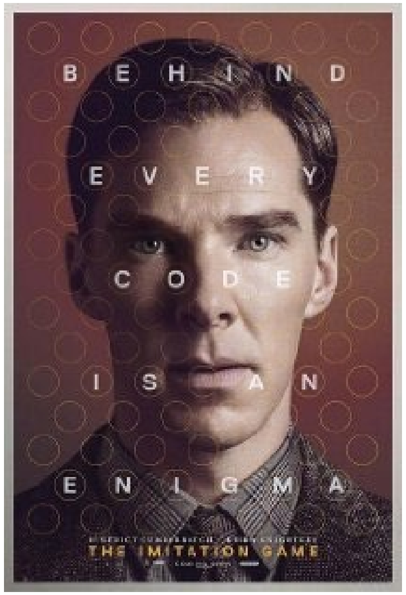
- 2001: NIST. 128, 192, 256 bits

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?