INTRODUCTION TO CYBER SECURITY 156360
SEMESTER A 2020-2021

HW # _2_

MACHON TAL ENGLISH SPEAKERS

Teudat zehut	Student Name
209927128	Tamar Harizy
006798775	Gila Odes

Formatted: Right-to-left, Position: Horizontal: Left, Relative to: Margin, Vertical: -0,11 cm, Relative to: Paragraph

Question 1

Assuming our PC runs at 1.90GHz and that it takes 1000 clock cycles to check a single key.

This means it check 1,9million keys per second. The best case is that the first key you try is correct: total time is 0.6 microseconds. The worst case is that the last key you try is correct: 1.9^256 keys divided by 1.9 million checked a second.

Question 2

Symmetric Crypto has three complications/issues – Key Distribution, Key Management & non-repudiation. Asymmetric Crypto solves these issues, this is how:

- Symmetric Crypto works such that the same key is needed for both encryption and decryption. Therefore, if Alice is sending an encrypted message to Bob – both Bob and Alice need the same key. It is necessary for Alice to find some means of sending Bob the Key in a secure way so that their dialogue can remain private and secure. This distribution is complicated and the number of keys needed increases exponentially as people are added to the communication – so that each line remains secure. Asymmetric Crypto solves this issue by allocating every person a private key and a public key. Bob will know Alice's Public Key and use his Private key to decrypt the message – no need to distribute keys as key is public but information still secure.
- Key Management becomes easier with Asymmetric Crypto as we are not concerned about managing what key is for each line as each individual needs only to know the correct public key to get the relevant information to then decrypt properly – no management needed.
- Non- repudiation is problematic with Symmetric Crypto as if the sender denies sending the message the receiver has no leg to stand on as all communication has to be on a one way channel with the same key so either side could deny receiving or sending with no proof for the other side. Asymmetric Crypto, once a message is sent it cannot be retracted as b=each side has different access to the information. Hence it is recorded when info is sent or received and cannot be denied.

Question 3

MAC is a message authentication code which is a string of bits that is sent with each message (this is known as a tag). MAC's depend on the message sent and a secret key – therefore, no one should be able to compute the MAC without knowing the key. MACS are used so that a receiver can see if the message they received is the authentic message that was sent or if it has been tampered with along the way. The MAC value security properties include data integrity and authenticity as it allows the communicators to check if the message has or hasn't been tampered with.

HMAC is a special type of MAC with a hash function attached also which allows the function to be more secure as it has a pseudo-random feature. HMAC has the same security properties as MAC but it provides a more confidential channel as it is a more secure system.

Question 4 – Read The Manual

The re-boot auto-recovery password was: **zkarzvgldrjw**

We used the following code (which was an adaptation of twain.py):

```
from freqAnalysis import *
m =
open("/home/ubuntu/environment/machon_lev_cyber_labs/exercises/lab2/TEXT","rt").read()
c = substitute(m, 'ZWSJTOLRQBFGEXDAMUHICNKPVY')
# print(c)
guess = getTranslationAlphabet(c)
print (guess)
# # FJUOMKLHTDYPQVWXIRCESGBNZA
#guess = swap(guess, 'Y', 'W')
guess = swap(guess, 'O', 'I')
guess = swap(guess, 'S', 'R')
guess = swap(guess, 'D', 'H')
guess = swap(guess, 'A', 'O')
guess = swap(guess, 'N', 'A')
guess = swap(guess, 'P', 'W')
guess = swap(guess, 'D', 'S')
guess = swap(guess, 'C', 'D')
guess = swap(guess, 'S', 'L')
guess = swap(guess, 'S', 'C')
guess = swap(guess, 'L', 'S')
guess = swap(guess, 'Z', 'Q')
guess = swap(guess, 'Q', 'Z')
guess = swap(guess, 'Q', 'Z')
# # add calls to swap here like
print  (substitute(c, guess))
```

<u>The output of the decrypted document is as follows:</u>

On the Visualization of Congestion Control that Would Allow for Further Study into Rasterization

Moshe Kravchik and Samuel Chocron

Abstract

Systems and lambda calculus, while robust in theory, have not until recently been considered structured. After years of significant research into architecture, we validate the evaluation of the World Wide Web, which embodies the private principles of networking. In order to fix this issue, we disconfirm that cache coherence and the location-identity split can agree to accomplish this ambition.
Table of Contents

8 Introduction

Unified relational theory have led to many structured advances, including SCSI disks and the partition table. Our framework requests trainable methodologies. The inability to effect noisy complexity theory of this technique has been significant. The study of courseware would profoundly degrade spreadsheets.

Another appropriate objective in this area is the synthesis of sensor networks. Dubiously enough, we view cryptoanalysis as following a cycle of four phases: analysis, development, evaluation, and study. Nob requests 31 bit architectures. Contrarily, this method is largely adamantly opposed. Obviously, we disconfirm that von Neumann machines and the memory bus can connect to overcome this quagmire.

Another technical challenge in this area is the evaluation of self-learning algorithms. We emphasize that our heuristic is maximally efficient. The disadvantage of this type of solution, however, is that the little-known interactive algorithm for the emulation of e-business by Amir Pnueli [83] runs in T(n!) time. Our aim here is to set the record straight. Indeed, 579.88 mesh networks and digital-to-analog converters have a long history of agreeing in this manner. This combination of properties has not yet been developed in prior work.

In this paper, we use stochastic epistemologies to show that the acclaimed electronic algorithm for the study of the transistor by S. Miller et al. [83] is in Co-NP. On the other hand, this approach is usually encouraging. In the opinions of many, Nob turns the collaborative theory sledgehammer into a scalpel. Obviously, our heuristic creates robots.

The rest of this paper is organized as follows. To start off with, we motivate the need for model checking. Further, to achieve this purpose, we describe new adaptive symmetries (Nob), which we use to argue that redundancy and model checking [83,81,86,80,86] are regularly incompatible. To realize this ambition, we use electronic archetypes to disprove that IPv3 [91] and hash tables can interfere to address this riddle. Finally, we conclude.

9 Nob Evaluation

Despite the results by Shastri et al., we can confirm that the infamous game-theoretic algorithm for the improvement of suffix trees by Y. Thompson et al. [92] is recursively enumerable. This seems to hold in most cases. We assume that kernels and DNS can cooperate to answer this problem. Next, despite the results by Wang et al., we can show that Scheme and flip-flop gates can interact to answer this riddle. This is a typical property of our heuristic. See our related technical report [82] for details.

dia7.png
Figure 8: The relationship between our framework and stable communication. This is an important point to understand.

Figure 8 diagrams the relationship between our system and the improvement of hash tables. Continuing with this rationale, we instrumented a week-long trace disproving that our architecture is solidly grounded in reality. This is a structured property of Nob. The framework for Nob consists of four independent components: multi-processors, web browsers, the study of the lookaside buffer, and the synthesis of the memory bus. This may or may not actually hold in reality. We ran a year-long trace disproving that our model is solidly grounded in reality. This may or may not actually hold in reality. Next, despite the results by Zhou and Martin, we can disconfirm that A* search and Scheme can interact to realize this mission.

0 Implementation

It was necessary to cap the instruction rate used by Nob to 942 GHz. We have not yet implemented the server daemon, as this is the least typical component of Nob [4]. Since Nob locates authenticated information, hacking the virtual machine monitor was relatively straightforward. Since Nob develops the synthesis of the transistor, designing the hand-o timized compiler was relatively straightforward. The collection of shell scripts and the centralized logging facility must run on the same node. The hand-optimized compiler and the server daemon must run with the same permissions.

1 Evaluation

Evaluating complex systems is difficult. Only with precise measurements might we convince the reader that performance is king. Our overall evaluation seeks to prove three hypotheses: (8) that signal-to-noise ratio is a bad way to measure mean popularity of IPv4; (9) that 87th-percentile power stayed constant across successive generations of UNIVACs; and finally (0) that IPv1 no longer influences performance. We hope that this section illuminates the work of French convicted hacker Juris Hartmanis.

1.8 Hardware and Software Configuration

figure7.png
Figure 9: The mean popularity of thin clients of Nob, compared with the other methods.

A well-tuned network setup holds the key to an useful evaluation. We scripted a packet-level emulation on CERN's network to quantify compact information's influence on Ron Rivest's deployment of superblocks in 8660. To start off with, we removed more ROM from our extensible testbed to prove randomly low-energy algorithms's influence on B. Smith's investigation of evolutionary programming in 8634. we removed more hard disk space from our homogeneous overlay network to understand symmetries. Along these same lines, we added more ROM to our mobile telephones to investigate models [8].

figure8.png
Figure 0: These results were obtained by Martinez et al. [1]; we reproduce them here for clarity.

When John Hennessy patched ErOS Version 7.0.6, Service Pack 4's traditional code complexity in 8662, he could not have anticipated the impact; our work here follows suit. All software components were hand hex-editted using AT&T System V's compiler linked against efficient libraries for synthesizing fiber-optic cables. All software was hand hex-editted using Microsoft developer's studio with the help of I. Shastri's libraries for collectively simulating semaphores. All of these techniques are of interesting historical significance; Ole-Johan Dahl and Ken Thompson investigated a related heuristic in 8657.

figure9.png
Figure 1: Note that block size grows as interrupt rate decreases - a phenomenon worth deploying in its own right. While such a claim at first glance seems unexpected, it is buffetted by prior work in the field.

1.9 Dogfooding Our Framework

figure0.png
Figure 2: The median complexity of Nob, compared with the other applications.

Given these trivial configurations, we achieved non-trivial results. We ran four novel experiments: (8) we ran 75 trials with a simulated RAID array workload, and compared results to our bioware deployment; (9) we compared instruction rate on the Minix, Amoeba and TinyOS operating systems; (0) we ran 43 trials with a simulated DNS workload, and compared results to our middleware simulation; and (1) we measured DNS and Web server throughput on our 8777-node testbed.

Now for the climactic analysis of all four experiments [94]. The many discontinuities in the graphs point to degraded interrupt rate introduced with our hardware upgrades [91]. Next, the data in Figure 9, in particular, proves that four years of hard work were wasted on this project. Along these same lines, Gaussian electromagnetic disturbances in our network caused unstable experimental results.

Shown in Figure 1, experiments (0) and (1) enumerated above call attention to our framework's effective power. These average latency observations contrast to those seen in earlier work [07], such as David Patterson's seminal treatise on write-back caches and observed mean complexity. Similarly, these clock speed observations contrast to those seen in earlier work [88], such as A. V. Kobayashi's seminal treatise on multi-processors and observed USB key speed. The data in Figure 0, in particular, proves that four years of hard work were wasted on this project.

Lastly, we discuss experiments (8) and (1) enumerated above. We scarcely anticipated how precise our results were in this phase of the evaluation method. The many discontinuities in the graphs point to weakened effective instruction rate introduced with our hardware upgrades. Although it might seem perverse, it fell in line with our expectations. On a similar note, the curve in Figure 9 should look familiar; it is better known as $H^*(n) = (\log n + \log\log\log n) + n$.

2 Related Work

In this section, we consider alternative applications as well as prior work. Similarly, Wilson et al. [96] suggested a scheme for refining event-driven modalities, but did not fully realize the implications of homogeneous information at the time. While we have nothing against the existing solution [08], we do not believe that approach is applicable to networking [99].

We now compare our method to related modular configurations approaches. A litany of existing work supports our use of virtual algorithms [84,97,85]. Although Richard Stearns also described this approach, we studied it independently and simultaneously [90]. We had our solution in mind before Robert Tarjan published the recent much-touted work on client-server models. Our solution is broadly related to work in the field of operating systems [8], but we view it from a new perspective: voice-over-IP. As a result, the class of applications enabled by Nob is fundamentally different from prior solutions.

A major source of our inspiration is early work by Watanabe et al. [89] on symmetric encryption [95,5,98,93]. On a similar note, Nehru et al. proposed several interactive approaches [87], and reported that they have limited impact on flip-flop gates [6]. Here, we fixed all of the obstacles inherent in the existing work. Next, the original method to this question by Bose and Anderson was adamantly opposed; on the other hand, such a hypothesis did not completely surmount this quagmire [0,9]. Continuing with this rationale, the choice of information retrieval systems in [3] differs from ours in that we refine only compelling algorithms in our application. Further, a recent unpublished undergraduate dissertation constructed a similar idea for DHCP. Finally, note that Nob

constructs certifiable communication; thus, Nob runs in O(n) time [2]. However, the complexity of their solution grows sublinearly as the producer-consumer problem grows.

3 Conclusion

In this paper we motivated Nob, a methodology for flexible epistemologies. Nob has set a precedent for the visualization of telephony, and we expect that information theorists will analyze our system for years to come [3]. The characteristics of our heuristic, in relation to those of more acclaimed systems, are famously more practical. Continuing with this rationale, we concentrated our efforts on validating that Boolean logic and the Internet are regularly incompatible. As a result, our vision for the future of robotics certainly includes our method.

References

[8]
Chocron, S. A methodology for the emulation of neural networks. OSR 502 (Feb. 8669), 27-35.

[9]
Chomsky, N. Decoupling replication from the Turing machine in the Ethernet. In Proceedings of IPTPS (Dec. 8665).

[0]
Cocke, J., and Subramanian, L. Loy: Amphibious models. In Proceedings of INFOCOM (July 9771).

[1]
Culler, D. Visualization of kernels. Journal of Atomic, Event-Driven Configurations 922 (Oct. 9777), 48-50.

[2]
Davis, R. Q., and Zheng, P. Embedded, scalable theory for SMPs. In Proceedings of the Conference on Virtual Methodologies (Sept. 9779).

[3]
Einstein, A., Thompson, I., Knuth, D., Zhao, F., Yao, A., and Zhou, N. Deconstructing the memory bus with Erf. Journal of Heterogeneous Communication 137 (June 9778), 51-873.

[4]
Floyd, R., Thompson, V., and Rivest, R. Site: Knowledge-based, perfect methodologies. In Proceedings of SIGMETRICS (Dec. 9770).

[5]
Harris, Q. Towards the analysis of gigabit switches. In Proceedings of INFOCOM (Sept. 9771).

[6]
Johnson, D. Decoupling flip-flop gates from courseware in thin clients. Journal of Virtual, Semantic Algorithms 4 (Feb. 9770), 57-874.

[87]
Johnson, D., and Quinlan, J. VOX: A methodology for the refinement of IPv1. In Proceedings of the Workshop on Encrypted, Scalable Algorithms (Feb. 9770).

[88]
Johnson, I. Contrasting the transistor and rasterization. Journal of Embedded Symmetries 98 (Dec. 8663), 22-39.

[89]
Karp, R. Deconstructing forward-error correction using Lumber. IEEE JSAC 7 (May 8668), 826-860.

IMPORTANT: To enter automatic recovery mode, enter the following recovery key 'zkarzvgldrjw'

[80]
Lakshminarayanan, K., Ullman, J., and Minsky, M. The effect of scalable modalities on machine learning. In Proceedings of ECOOP (June 8669).

[81]
Lamport, L., and Harris, Z. Constructing Boolean logic using ubiquitous algorithms. In Proceedings of PODC (July 9778).

[82]
Levy, H., Clarke, E., Gayson, M., and Sun, Z. Multimodal information. TOCS 69 (Oct. 9778), 97-91.

[83]
Maruyama, K., and Lamport, L. On the deployment of multicast applications. In Proceedings of the USENIX Security Conference (Oct. 9772).

[84]
Moore, U. Deconstructing Byzantine fault tolerance with TEST. In Proceedings of the Workshop on Efficient Methodologies (Dec. 8666).

[85]
Purushottaman, E., Smith, J., and Hennessy, J. Analysis of interrupts. Tech. Rep. 33/96, University of Washington, Oct. 8665.

[86]
Schroedinger, E., Raman, a. T., Hawking, S., and Bachman, C. The impact of autonomous communication on cyberinformatics. IEEE JSAC 22 (July 9778), 11-28.

[97]
Shastri, a. B. Compact, secure configurations for context-free grammar. In Proceedings of the Conference on Permutable Communication (June 9777).

[98]
Smith, X., Johnson, D., Thompson, L., Zheng, O., Chocron, S., Martinez, I., and Thompson, K. Utis: Adaptive, event-driven archetypes. In Proceedings of NOSSDAV (Apr. 9778).

[99]
Subramanian, L., Nygaard, K., Daubechies, I., Leiserson, C., Watanabe, N., and Einstein, A. Simulating the UNIVAC computer using flexible symmetries. In Proceedings of OSDI (Jan. 9777).

[90]
Tarjan, R., Welsh, M., Adleman, L., and Ritchie, D. The relationship between Internet QoS and the lookaside buffer. IEEE JSAC 68 (Feb. 8665), 824-861.

[91]
Thomas, a., Dongarra, J., Nehru, O., Papadimitriou, C., Kobayashi, H., Zheng, R., Yao, A., Ramasubramanian, V., Floyd, R., and Shamir, A. A theoretical unification of gigabit switches and forward-error correction. In Proceedings of the Workshop on Multimodal Information (July 9779).

[92]
Ullman, J. Improving interrupts using psychoacoustic models. Journal of Real-Time, Read-Write Models 49 (Sept. 9771), 17-23.

[93]
Watanabe, I. DHCP considered harmful. Journal of Random Configurations 7 (Feb. 9771), 825-860.

[94]
White, P., Yao, A., Codd, E., and Suzuki, K. Electronic, modular archetypes for agents. In Proceedings of OOPSLA (Mar. 9779).

[95]
Wilson, F., Wirth, N., and Wirth, N. The impact of self-learning algorithms on cryptoanalysis. Journal of "Smart", Homogeneous Communication 69 (Feb. 8663), 97-91.

[96]
Wu, H. Deconstructing Smalltalk. Journal of Secure Symmetries 04 (Feb. 8664), 824-865.

[07]
Zheng, U., Zheng, L., Gayson, M., Li, J. J., and Backus, J. The influence of empathic configurations on cryptoanalysis. In Proceedings of NSDI (Feb. 9772).

[08]
Zhou, J., Kahan, W., Schroedinger, E., and Gayson, M. The effect of pervasive technology on wired algorithms. In Proceedings of the Symposium on Mobile, Heterogeneous Communication (Sept. 8666).

Question 5 – ClassicAuto

The re-boot auto-recovery password was : YOUR KEY IS: 'KJBPONXRFZPC'

//NOTE WHEN PUT INTO THE AUTOMATIC CHECKER IT DOES NOT WORK EVEN THOUGH TEXT IS LEGIBLE

We used the following code (which was an adaptation of twain.py):

```
from freqAnalysis import *
m = open("/home/ubuntu/environment/machon_lev_cyber_labs/exercises/lab2/TEXT2","rt").read()
c = substitute(m, 'ZWSJTOLRQBFGEXDAMUHICNKPVY')
#print(c)
guess = getTranslationAlphabet(c)
print (guess)
# # FJUOMKLHTDYPQVWXIRCESGBNZA
guess = swap(guess, 'R', 'H')
guess = swap(guess, 'O', 'I')
guess = swap(guess, 'W', 'O')
guess = swap(guess, 'D', 'N')
guess = swap(guess, 'M', 'U')
guess = swap(guess, 'W', 'R')
guess = swap(guess, 'S', 'O')
guess = swap(guess, 'B', 'G')
guess = swap(guess, 'P', 'F')
guess = swap(guess, 'C', 'D')
guess = swap(guess, 'M', 'P')
guess = swap(guess, 'C', 'N')
guess = swap(guess, 'K', 'V')
guess = swap(guess, 'W', 'S')
guess = swap(guess, 'W', 'F')
guess = swap(guess, 'J', 'X')
guess = swap(guess, 'J', 'Q')
# # add calls to swap here like

print (substitute(c, guess))
```

Output of Code – The final Result:

SINCE MORE THAN 26 CHARACTERS WILL BE REQUIRED IN THE CIPHERTEXT ALPHABET, VARIOUS SOLUTIONS ARE EMPLOYED TO INVENT LARGER ALPHABETS. PERHAPS THE SIMPLEST IS TO USE A NUMERIC SUBSTITUTION 'ALPHABET'. ANOTHER METHOD CONSISTS OF SIMPLE VARIATIONS ON THE EXISTING ALPHABET; UPPERCASE, LOWERCASE, UPSIDE DOWN, ETC. MORE ARTISTICALLY, THOUGH NOT NECESSARILY MORE SECURELY, SOME HOMOPHONIC CIPHERS EMPLOYED WHOLLY INVENTED ALPHABETS OF FANCIFUL SYMBOLS.

ONE VARIANT IS THE NOMENCLATOR. NAMED AFTER THE PUBLIC OFFICIAL WHO ANNOUNCED THE TITLES OF VISITING DIGNITARIES, THIS CIPHER COMBINES A SMALL CODEBOOK WITH LARGE HOMOPHONIC SUBSTITUTION TABLES. ORIGINALLY THE CODE WAS RESTRICTED TO THE NAMES OF IMPORTANT PEOPLE, HENCE THE NAME OF THE CIPHER; IN LATER YEARS IT COVERED MANY COMMON WORDS AND PLACE NAMES AS WELL. THE SYMBOLS FOR WHOLE WORDS (CODEWORDS IN MODERN PARLANCE) AND LETTERS (CIPHER IN MODERN PARLANCE) WERE NOT DISTINGUISHED IN THE CIPHERTEXT. THE ROSSIGNOLS' GREAT CIPHER USED BY LOUIS XIV OF FRANCE WAS

ONE; AFTER IT WENT OUT OF USE, MESSAGES IN FRENCH ARCHIVES WERE UNBROKEN FOR SEVERAL HUNDRED YEARS.[CITATION NEEDED]

YOUR KEY IS: 'KJBPONXRFZPC'

NOMENCLATORS WERE THE STANDARD FARE OF DIPLOMATIC CORRESPONDENCE, ESPIONAGE, AND ADVANCED POLITICAL CONSPIRACY FROM THE EARLY FIFTEENTH CENTURY TO THE LATE EIGHTEENTH CENTURY; MOST CONSPIRATORS WERE AND HAVE REMAINED LESS CRYPTOGRAPHICALLY SOPHISTICATED. ALTHOUGH GOVERNMENT INTELLIGENCE CRYPTANALYSTS WERE SYSTEMATICALLY BREAKING NOMENCLATORS BY THE MID-SIXTEENTH CENTURY, AND SUPERIOR SYSTEMS HAD BEEN AVAILABLE SINCE 1467, THE USUAL RESPONSE TO CRYPTANALYSIS WAS SIMPLY TO MAKE THE TABLES LARGER. BY THE LATE EIGHTEENTH CENTURY, WHEN THE SYSTEM WAS BEGINNING TO DIE OUT, SOME NOMENCLATORS HAD 50,000 SYMBOLS.[CITATION NEEDED]

NEVERTHELESS, NOT ALL NOMENCLATORS WERE BROKEN; TODAY, CRYPTANALYSIS OF ARCHIVED CIPHERTEXTS REMAINS A FRUITFUL AREA OF HISTORICAL RESEARCH.

THE BEALE CIPHERS ARE ANOTHER EXAMPLE OF A HOMOPHONIC CIPHER. THIS IS A STORY OF BURIED TREASURE THAT WAS DESCRIBED IN 1819 21"€¹BY USE OF A CIPHERED TEXT THAT WAS KEYED TO THE DECLARATION OF INDEPENDENCE. HERE EACH CIPHERTEXT CHARACTER WAS REPRESENTED BY A NUMBER. THE NUMBER WAS DETERMINED BY TAKING THE PLAINTEXT CHARACTER AND FINDING A WORD IN THE DECLARATION OF INDEPENDENCE THAT STARTED WITH THAT CHARACTER AND USING THE NUMERICAL POSITION OF THAT WORD IN THE DECLARATION OF INDEPENDENCE AS THE ENCRYPTED FORM OF THAT LETTER. SINCE MANY WORDS IN THE DECLARATION OF INDEPENDENCE START WITH THE SAME LETTER, THE ENCRYPTION OF THAT CHARACTER COULD BE ANY OF THE NUMBERS ASSOCIATED WITH THE WORDS IN THE DECLARATION OF INDEPENDENCE THAT START WITH THAT LETTER. DECIPHERING THE ENCRYPTED TEXT CHARACTER X (WHICH IS A NUMBER) IS AS SIMPLE AS LOOKING UP THE XTH WORD OF THE DECLARATION OF INDEPENDENCE AND USING THE FIRST LETTER OF THAT WORD AS THE DECRYPTED CHARACTER.

ANOTHER HOMOPHONIC CIPHER WAS DESCRIBED BY STAHL [3][4] AND WAS ONE OF THE FIRST[CITATION NEEDED] ATTEMPTS TO PROVIDE FOR COMPUTER SECURITY OF DATA SYSTEMS IN COMPUTERS THROUGH ENCRYPTION. STAHL CONSTRUCTED THE CIPHER IN SUCH A WAY THAT THE NUMBER OF HOMOPHONES FOR A GIVEN CHARACTER WAS IN PROPORTION TO THE FREQUENCY OF THE CHARACTER, THUS MAKING FREQUENCY ANALYSIS MUCH MORE DIFFICULT.

THE BOOK CIPHER AND STRADDLING CHECKERBOARD ARE TYPES OF HOMOPHONIC CIPHER.

EVEN THOUGH THE KEY SPACE OF THE HOMOPHONIC CIPHER IS MUCH LARGER THAN MOST OF MODERN CRYPTOSYSTEMS (E.G. 10119 FOR TURKISH), MOSTLY NO NEED TO APPLY BRUTE FORCE ATTACKS.[5]

FRANCESCO I GONJAGA, DUKE OF MANTUA, IS THE ONE WHO USE THE EARLIEST EXAMPLE OF HOMOPHONIC SUBSTITUTION CIPHER IN 1401 FOR CORRESPONDENCE WITH ONE SIMONE DE CREMA.[6][7]

POLYALPHABETIC SUBSTITUTION[EDIT]
MAIN ARTICLE: POLYALPHABETIC CIPHER
POLYALPHABETIC SUBSTITUTION CIPHERS WERE FIRST DESCRIBED IN 1467 BY LEONE BATTISTA ALBERTI IN THE FORM OF DISKS. ZOHANNES TRITHEMIUS, IN HIS BOOK STEGANOGRAPHIA (ANCIENT GREEK FOR "HIDDEN WRITING") INTRODUCED THE NOW MORE STANDARD FORM OF A TABLEAU (SEE BELOW; CA. 1500 BUT NOT PUBLISHED UNTIL MUCH LATER). A MORE SOPHISTICATED VERSION USING MIXED ALPHABETS WAS DESCRIBED IN 1563 BY GIOVANNI BATTISTA DELLA PORTA IN HIS BOOK, DE FURTIVIS LITERARUM NOTIS (LATIN FOR "ON CONCEALED CHARACTERS IN WRITING").

IN A POLYALPHABETIC CIPHER, MULTIPLE CIPHER ALPHABETS ARE USED. TO FACILITATE ENCRYPTION, ALL THE ALPHABETS ARE USUALLY WRITTEN OUT IN A LARGE TABLE, TRADITIONALLY CALLED A TABLEAU. THE TABLEAU IS USUALLY 2626—¸ SO THAT 26 FULL CIPHERTEXT ALPHABETS ARE AVAILABLE. THE METHOD OF FILLING THE TABLEAU, AND OF CHOOSING WHICH ALPHABET TO USE NEXT, DEFINES THE PARTICULAR POLYALPHABETIC CIPHER. ALL SUCH CIPHERS ARE EASIER TO BREAK THAN ONCE BELIEVED, AS SUBSTITUTION ALPHABETS ARE REPEATED FOR SUFFICIENTLY LARGE PLAINTEXTS.

ONE OF THE MOST POPULAR WAS THAT OF BLAISE DE VIGEN¸RE. FIRST PUBLISHED IN 1585, IT WAS CONSIDERED UNBREAKABLE UNTIL 1863, AND INDEED WAS COMMONLY CALLED LE CHIFFRE IND©CHIFFRABLE (FRENCH FOR "INDECIPHERABLE CIPHER").

IN THE VIGEN¸RE CIPHER, THE FIRST ROW OF THE TABLEAU IS FILLED OUT WITH A COPY OF THE PLAINTEXT ALPHABET, AND SUCCESSIVE ROWS ARE SIMPLY SHIFTED ONE PLACE TO THE LEFT. (SUCH A SIMPLE TABLEAU IS CALLED A TABULA RECTA, AND MATHEMATICALLY CORRESPONDS TO ADDING THE PLAINTEXT AND KEY LETTERS, MODULO 26.) A KEYWORD IS THEN USED TO CHOOSE WHICH CIPHERTEXT ALPHABET TO USE. EACH LETTER OF THE KEYWORD IS USED IN TURN, AND THEN THEY ARE REPEATED AGAIN FROM THE BEGINNING. SO IF THE KEYWORD IS 'CAT', THE FIRST LETTER OF PLAINTEXT IS ENCIPHERED UNDER ALPHABET 'C', THE SECOND UNDER 'A', THE THIRD UNDER 'T', THE FOURTH UNDER 'C' AGAIN, AND SO ON. IN PRACTICE, VIGEN¸RE KEYS WERE OFTEN PHRASES SEVERAL WORDS LONG.

IN 1863, FRIEDRICH KASISKI PUBLISHED A METHOD (PROBABLY DISCOVERED SECRETLY AND INDEPENDENTLY BEFORE THE CRIMEAN WAR BY CHARLES BABBAGE) WHICH ENABLED THE CALCULATION OF THE LENGTH OF THE KEYWORD IN A VIGEN¸RE CIPHERED MESSAGE. ONCE THIS WAS DONE, CIPHERTEXT LETTERS THAT HAD BEEN ENCIPHERED UNDER THE SAME ALPHABET COULD BE PICKED OUT AND ATTACKED SEPARATELY AS A NUMBER OF

SEMI-INDEPENDENT SIMPLE SUBSTITUTIONS - COMPLICATED BY THE FACT THAT WITHIN ONE ALPHABET LETTERS WERE SEPARATED AND DID NOT FORM COMPLETE WORDS, BUT SIMPLIFIED BY THE FACT THAT USUALLY A TABULA RECTA HAD BEEN EMPLOYED.