

One-time Pad [Deze pagina in het Nederlands](#)

[Home](#) [Manual One-time pads](#)

Definition of One-time pad

One-time pad (OTP), also called Vernam-cipher or the perfect cipher, is a crypto algorithm where plaintext is combined with a random key. It is the only existing mathematically unbreakable encryption.

Used by Special Operations teams and resistance groups during WW2, popular with intelligence agencies and their spies during the Cold War and beyond, protecting diplomatic and military message traffic around the world for many decades, the one-time pad gained a reputation as a simple yet solid encryption system with an absolute security which is unmatched by today's modern crypto algorithms. Whatever technological progress may come in the future, one-time pad encryption is, and will remain, the only truly unbreakable system that provides real long-term message secrecy.

We can only talk about one-time pad if some important rules are followed. If these rules are applied correctly, the one-time pad can be proven unbreakable (see Claude Shannon's "Communication Theory of Secrecy Systems"). Even infinite computational power and infinite time cannot break one-time pad encryption, simply because it is mathematically impossible. However, if only one of these rules is disregarded, the cipher is no longer unbreakable.

- The key is at least as long as the message or data that must be encrypted.
- The key is truly random (not generated by a simple computer function or such)
- Key and plaintext are calculated modulo 10 (digits), modulo 26 (letters) or modulo 2 (binary)
- Each key is used only once, and both sender and receiver must destroy their key after use.
- There should only be two copies of the key: one for the sender and one for the receiver (some exceptions exist for multiple receivers)

Important note: one-time pads or one-time encryption is not to be confused with one-time keys (OTK) or one-time passwords (sometimes also denoted as OTP). Such one-time keys, limited in size, are only valid for a single encryption session by some crypto-algorithm under control of that key. Small one-time keys are by no means unbreakable, because the security of the encryption depends on the crypto algorithm they are used for.



A miniature paper one-time pad
© Dirk Rijmenants

Origins of One-time pad ▲

The story of one-time pad starts in 1882, when the Californian banker Frank Miller compiles his "Telegraphic Code to Insure Privacy and Secrecy in the Transmission of Telegrams". Such codebooks were commonly used, mainly to reduce telegraph costs by compressing words and phrases into short number-codes or letter-codes. These codebooks provided little or no security. However, Miller's codebook also provided instructions for a superencipherment (a second encipherment layer over the code) by a unique method: he added so-called shift-numbers (the key) to the plaincode (words, converted into a number) and defined the shift-numbers as a list of irregular numbers that should be erased after use and never be used again.

His codebook contained 14,000 words, phrases and blanks (for customizing) and if during enciphering the sum of plaincode and key exceeded 14,000, one had to subtract 14,000 from the sum. If during deciphering the ciphertext value was smaller than the key, one had to add 14,000 to the ciphertext and then subtract the key (this is basically a modulo 14,000 arithmetic). If the shift-numbers were randomly chosen and used once only, the modular arithmetic provided unbreakable encryption. Miller had invented the first ever one-time pad. Unfortunately, Miller's perfect cipher never became generally known, got lost in the history of cryptography and never received the deserved credits. As early as it was invented, so soon it disappeared in oblivion, only to be rediscovered in archives in 2011.

Then, in 1917, AT&T research engineer Gilbert Vernam developed a system to encrypt teletype TTY communications. Although Vernam's invention mathematically resembles Miller's idea, he devised a electromechanical system, completely different to Miller's pen-and-paper algorithm. Therefore, it seems unlikely that Vernam borrowed Miller's idea. Vernam mixed a five-bit Baudot-coded punched paper tape, containing the message, with a second punched paper tape, the key, containing random five-bit values. To mix the punched tapes, a modulo 2 addition (later known as the Boolean XOR or Exclusive OR) was performed with relays, and the key tape ran synchronously on the sending and receiving TELEX machine. It was the first automated instant on-line encryption system.

Vernam realized that encryption with short key tapes (basically a poly-alphabetic cipher) would not provide enough security. Initially, Vernam used a mix of two key tape loops, with relatively prime length, creating one very long random key. Captain Joseph Mauborgne (later Chief of the U.S. Signal Corps) showed that even the double key tape system could not resist cryptanalysis if large volumes of message traffic were encrypted. Mauborgne concluded that only if the key tape is unpredictable, as long as the message and used only once, the message would be secure. Moreover, the encryption proved to be unbreakable. One-time encryption was reborn.

NSA called Vernam's 1919 one-time tape (OTT) patent "perhaps one of the most important in the history of cryptography". AT&T marketed the Vernam system in the 1920s for commercial secure communications, albeit with little success. The production, distribution and consumption of enormous quantities of one-time tapes limited its use to fixed stations (headquarters or communications centers). It was not until the Second World War that the US Signal Corps widely used the OTT system for its high level teleprinter communications. However, three German cryptologists did immediately recognize the advantages of one-time encryption.

In the early 1920s, the German cryptologists Werner Kunze, Rudolf Schauffler and Erich Langlotz cryptanalysed French diplomatic traffic. These pencil-and-paper numerical codes used code books to convert words and phrases into digits. The French added a short repetitive numerical key (by modulo 10) to encrypt the code book values. The German cryptologists had no problem in breaking these short keys but realized that adding a unique random key digit to each individual code group digit would make the message unbreakable. They devised a system with paper sheets containing random digits, each digit to be used once only, and the sheets, of which there were only two copies (one for sender and one for receiver), should be destroyed after use. In fact, they re-invented Frank Miller's 1882 system.

By 1923, the system was introduced in the German foreign office to protect their diplomatic correspondence (see image right). For the first time in history, diplomats had truly unbreakable encryption at their disposal. Later on, many variations on this pencil-and-paper system were devised. The name one-time pad (OTP) refers to small note pads with random digits or letters, usually printed in groups of five. For each new message, a new sheet is torn off. They are often printed as small very booklets or on microfilm for covert communications. (more about the use of such pads is found on the [manual one-time pads](#) page).



Miniature one-time pads and conversion table
from the former East German Intelligence agency
HVA (Hauptverwaltung Aufklärung)
© SAS Chiffrierdienst



One-time pad booklet and microdot reader,
concealed in a toy truck and used by an
illegal agent that operated in Canada.
© Canadian Security Intelligence Service

NEW The early use of one-time pads is hardly mentioned in official documents (for obvious security reasons). Nevertheless, I came across documents from the India Office Records in the British Library. They show how the Bahrain Petroleum Company (BAPCO), a subsidiary of American Standard Oil of California that operated in the Persian Gulf, was given permission in 1943 to use one-time pads to communicate with its offices in New York. The pads were allocated to them by the U.S. Navy Department and vetted by the British Cipher Security Officer of PAIFORCE (Persia and Iraq Force, a British and Commonwealth military formation in the Middle East from 1942 to 1943). They show the official use of one-time letter pads by Political Residents of the British Imperial Civil Administration, the British Army, the Ministry of War Transport in London and the U.S. Navy, at least as early as 1943 and, surprisingly, even shared them with commercial firms. More details in [my blog post about BAPCO](#).

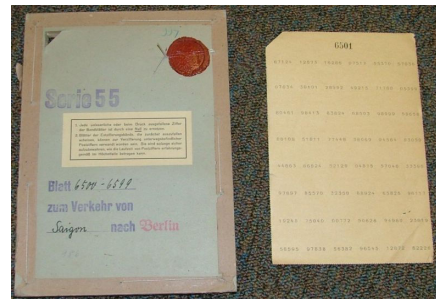
In 1943, one-time pads became the main cipher of the Special Operations Executive (SOE) to replace insecure poem based transposition ciphers and book ciphers (see [SOE Field Ciphers](#)). The system was used extensively during and after the Second World War by many intelligence organizations, sabotage and espionage units. The unbreakable encryption protects operatives and their contacts against decryption of their communications and disclosure of their identities. Such level of security cannot be guaranteed with other encryption systems during long-running operations because the opponent might have enough time to successfully decrypt the messages. The Soviets relied heavily on OTP's and OTT's during and after the Second World War for their armed forces and intelligence organizations, making much of their vital communications virtually impenetrable.

One-time tapes and one-time pads remained very popular for many decades, because of their absolute security, unequalled by any other crypto machine or algorithm. Today, digital versions of the one-time pad enable the storage of huge quantities of random key data, allowing secure encryption of large volumes of data. One-time encryption still is, and will continue to be, the only system that can offer absolute message security.

On the right you find images of various paper one-time pads. A miniature one-time pad booklet, together with a microdot reader and a special lens, was cleverly concealed in a toy truck that was brought into Canada by the young son of a foreign intelligence operative that entered the country to carry out espionage ([Canadian CSIS](#)). You can also see a TAPIR conversion table, used by East-German operatives ([SAS und Chiffrierdienst](#)). There's also a German one-time pad, used for official communications between Saigon and Berlin, that consists of a sealed folder with one hundred one-time pad worksheets, numbered 6500 to 6599. Each sheet contains random numbers and enough space to write down the message and perform the calculations ([NSA National Cryptologic Museum](#)). The last image is part of a one-time pad, used by Aleksandr Ogorodnik, a Soviet Foreign Ministry employee who committed espionage for the CIA ([Andrei Sineinikov \[Ru\]](#) ([English](#))).

A	E	I	N	R	TAPIR				
0	1	2	3	4	55	56	57	58	59
B	BE	C	CH	D	DE	F	G	GE	H
50	51	52	53	54	55	56	57	58	59
J	K	L	M	O	65	66	67	68	69
60	61	62	63	64	65	66	67	68	69
T	TE	U	UN	V	75	76	77	78	79
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99
0	1	2	3	4	5	6	7	8	9
00	11	22	33	44	55	66	77	88	99

Tapir conversion table © SAS und Chiffrierdienst



German one-time sheets. Image courtesy © NSA

24765	93659	55146	09380	18882	67898	69598
25341	88038	31282	39857	21708	51305	66499
65096	02819	74377	27960	20471	53361	18687
19226	31329	55134	83869	26588	24850	81322
01334	80225	37061	13995	88627	87293	53021
90865	91712	80927	18799	71311	57151	71976
98890	61224	59636	08076	65747	36834	49525
95428	50476	06584	38300	37155	75549	11968
43041	83175	29737	88523	76769	29465	47144
77230	19601	57378	51440	48030	63857	15846
32548	46508	71999	22399	86499	22365	91365
57311	83798	06280	74855	58916	46616	07784
10464	00582	88702	30607	80017	50120	76361
93610	38382	57828	27710	00047	00977	02927
53217	20255	20839	63759	74408	60213	32159
31617	14857	97505	25301	14258	36792	42161
52190	32626	07392	80180	32382	22804	82072
39585	92345	44974	09467	88114	50678	84634
44347	73284	49702	60171	56691	11969	37108
84448	37447	32088	01470	85391	00675	21874

Part of a CIA one-time pad used by Aleksandr Ogorodnik (TRIGON)
Source: KGB Archives

Click the images to enlarge them

Paper One-time pads ▲

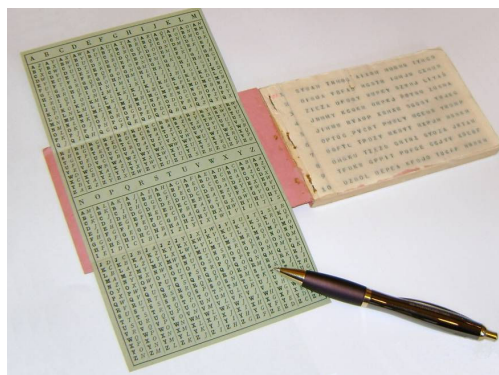
The use of pencil-and-paper one-time pads is limited because of the practical and logistical issues and the low message volume it can process. One-time pads were widely used by foreign service communicators until the 1980s, often in combination with code books. Such a code book contained all kinds of words or entire phrases, which were represented by a three or four figure code. For special names or expressions, not listed in the codebook, there were codes included that represent one letter that allowed the spelling of words. There was a book to encode, sorted by alphabet and/or category, and a book to decode, sorted by numbers. These books were valid for a long period of time and were not only to encode the message - which would be a poor encryption method by itself - but especially to reduce its length for transmission over commercial cable or telex.

Once the message was converted into numbers, the communicator enciphered these numbers with the one-time pad. Usually there was a set of two different pads, one for incoming and one for outgoing messages. Although a one-time pad normally has only two copies of a key, one for sender and one for receiver, some systems used more than two copies to address multiple receivers. The pads were like note blocks with random numbers on each small page, but with the edges sealed. One could only read the next pad by tearing off the previous pad. Each pad was used only once and destroyed immediately. This system enabled absolute secure communication. A good description of the use of one-time pads by the Canadian Foreign Service can be found on [Jerry Proc's website](#).

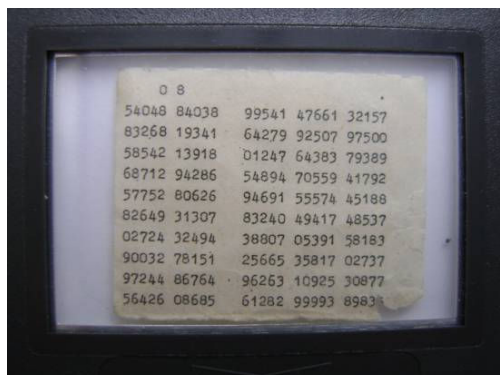
Intelligence agencies use one-time pads to communicate with their agents in the field, where security has absolute priority. With one-time pad, they don't have to carry compromising crypto systems or computer programs with them. They can carry a large number of one-time pad keys in very small booklets, on microfilm or even printed on clothing. These are easy to hide and to destroy. One way to send one-time pad encrypted messages to agents in the field is via [numbers stations](#). To do so, the message text is converted into digits prior to encryption.

A good example is the [TAPIR procedure](#), used by the former East Germany intelligence, and published on the [SAS und Chiffrierdienst](#) website. With TAPIR, the plain text is converted into figures by a table, similar to the straddling checkerboard, prior to encryption with one-time pad. The most frequent letters have one digit, other letters, commonly used bigrams, figures and signs have two digits. Next, the digits are encrypted by subtracting the key from the plain text numbers. The TAPIR table suppresses peaks of the digit frequency distribution and creates fractionation. In a ciphered text one never knows if a digit is a complete letter, or a left or right part of a letter or figure. WR 80 is a carriage return. Bu 81 (Buchstaben) and Zi 82 (Ziffern) are used to switch between letters (yellow) and figures (green). ZWR 83 is a space. Code 84 is used as prefix for four-digit or five-digit codes, replacing long words or sentences, obtained from a codebook. You can [view an example codebook here](#). Note that the odd sequence of numbers that represent the words or expressions on the codebook are composed carefully in such a way that errors in the code values are easily detected. On the SAS und Chiffrierdienst website you can also find a good description by East-German intelligence (Stasi) of the [one-time pad procedures](#) for numbers messages, used by CIA agents who operated in the former DDR). One-time Pad booklets and TAPIR table courtesy [SAS und Chiffrierdienst](#), copyrighted). More details about the use of paper one-time pads are found at [manual one-time pad page](#)

Below, on the left, a one-time pad booklet with Vigenere table from a Western agent, seized by the East-German MfS (Ministerium für Staatssicherheit or Stasi). The second image is a one-time pad sheet (preserved in a 35 mm slide frame) from an East-German agent, found by the West-German BfV (Bundesamt für Verfassungsschutz, the federal domestic intelligence). The right-most image is a one-time pad of a West agent, found by the MfS (also preserved in a 35 mm slide frame). The pad itself is only about 15 mm or 0.6 inch wide (thus even smaller than depicted) and virtually impossible to read with the naked eye! I even had difficulties to photograph it clearly. Such miniature one-time pads were used by illegal agents, operating in foreign countries, and were hidden inside innocent looking household items like cigarette lighters, fake batteries or ashtrays. You can click the images to enlarge them. However, to read the small pad you will need to click and zoom in once more in your browser after enlarging (Detlev Freisleben collection).



Letters-only one-time pad booklet with Vigenere table



A standard 250 digits one-time pad (HVA-Stasi)



Miniature pad

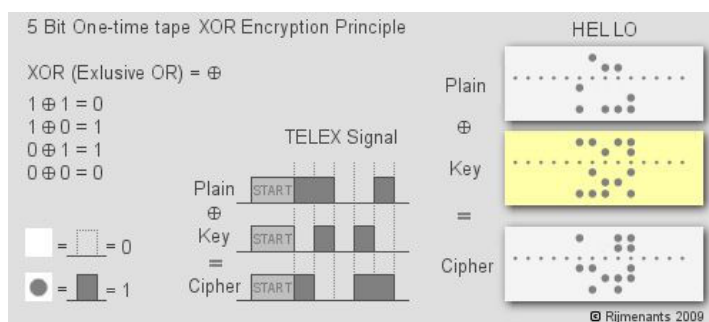
All images on this page are copyrighted. More about copyrights and the use of images [on this page](#).

One-time pad based Crypto Machines ▲

Until the 1980s, one-time-tapes were widely used to secure Telex communications. The Telex machines used Vernam's original one-time-tape (OTT) principle. The system was simple but solid. It required two identical reels of punched paper tape with truly random five-bit values, the so-called one-time tapes. These were distributed beforehand to both sender and receiver. Usually, the message was prepared (punched) in plain onto paper tape. Next, the message was transmitted on a Telex machine with the help of a tape reader, and one copy of the secret one-time tape ran synchronously with the message tape on a second tape reader. Before exiting the machine, the five-bit signals of both tape readers were mixed by performing an Exclusive OR (XOR) function, thus scrambling the output. On the other end of the line, the scrambled signal entered the receiving machine and was mixed, again by XOR, with the second copy of the secret one-time tape. Finally, the resulting readable five-bit signal was printed or perforated on the receiving machine.

A unique advantage of the punched paper tape keys was that copying them quickly was virtually impossible. The long tapes (which were sealed in plastic before use) were on a reel and printed with serial numbers and other markings on the side. To unwind the tape, copy it and rewind it again with a perfectly aligned print was very unlikely and such one-time tapes were therefore more secure than other keys sheets that were copied quickly by taking a photo or writing them over by hand.

A famous example of one-time pad's security is the [Washington/Moscow hotline](#) with the [ETCRRM II](#), a standard commercial one-time tape mixer for Telex. Although simple and cheap, it provided absolute security and unbreakable communications between Washington and the Kremlin, without disclosing any secret crypto technology. Some other cipher machines that used the principle of one-time pad are the American [TELEKRYPTON](#), [SIGSALY](#) (noise as one-time pad), [B-2 PYTHON](#) and SIGTOT, the British [BID-590 NOREEN](#) and [5-UCO](#), the Canadian [ROCKEX](#), the Dutch [ECOLEX series](#), the Swiss Hagelin CD-57 RT, CX-52 RT and T-55 with a superencipherment option, the German Siemens [T-37-ICA](#) and [M-190](#), the East German [T-304 LEGUAN](#), the Czech [SD1](#), the Russian [M-100 SMARAGD](#) and [M-105 N AGAT](#) and the Polish [T-352/T-353 DUDEK](#). There were also many teletype or ciphering device configurations in combination with a tape reader, for one-time tape encryption or superencipherment. The image below explains one-time tape encryption for Telex (TTY Murray).



Teletype signal one-time tape encryption

One-time tape Teletype Hotline ▲

Below are three images of the famous Washington-Moscow hotline, encrypted with one-time tapes. The Hotline became operational in 1963 and was a full duplex teleprinter (Telex) circuit. Although the Hotline always was shown as a red telephone in movies and popular culture, the option of a speech link was turned down immediately as it was believed that spontaneous verbal communications could lead to miscommunications, misperceptions, incorrect translation or unwise spontaneous remarks, which are serious disadvantages in times of crisis. Nevertheless, the red phone myth lived a long life.

The real hot line was a direct cable link, routed from Washington over London, Copenhagen, Stockholm and Helsinki to Moscow. It was a double link with commercial teleprinters, one link with a Teletype Corp Model 28 ASR teleprinter with English characters and the other link with East German T-63 teleprinters with Cyrillic character. The links were encrypted with one-time tapes by means of four ETCRRM's (Electronic Teleprinter Cryptographic Regenerative Repeater Mixer). The one-time tape encryption provided unbreakable encryption, absolute security and privacy. Although a highly secure system, the unclassified standard teleprinters and ETCRRM's were sold by commercial firms and therefore did not disclose any secret crypto technology to the opponent. More info at Jerry Proc's [Washington/Moscow hotline](#) and on [Top Level Communications](#).



The ETCRRM
Image © NSA



The East-German T-63, used on the US-USSR hotline,
together with the ETCRRM one-time tape mixer.
Image courtesy National Security Agency. © NSA.



Inside the Washington-Moscow hotline room.
Four black ETCRRM's are placed in the middle.
Image courtesy National Security Agency. © NSA.

Hotline images with kind permission of the National Security Agency, copyright NSA (click to enlarge)

Usability of One-time pad ▲

One-time pad encryption is only possible if both sender and receiver are in possession of the same key. Therefore, the keys must be exchanged physically and securely beforehand, by both parties personally or through a trusted courier. This means that the secure communications are expected and planned within a specific time frame. Enough key material must be available for all required communications until a new exchange of keys is possible. Depending upon the situation, a large volume of keys could be required for a short time period, or little key material could be sufficient for a very long time period, up to several years. One-time pads are more suitable for the latter.

Procedures are required to ensure that the key material is always accounted for, and tamper proof systems must ensure detection of unauthorized access or copying of the keys. One-time pads are especially interesting in circumstances where long-term security is essential. One-time pad is not suitable for encryption of information that is no longer important after a short time period that exceeds the time to cryptanalyse normal encryption algorithms. Although usable for such communications, the efforts for secure distribution of keys will not be in proportion to the required security, and a normal crypto algorithm will be more suitable.

Although one-time pad is the only perfect cipher, it has two major disadvantages. The first problem is the generation of a large quantity of random numbers or letters. For absolutely true randomness we cannot create these keys by simple mechanical devices or computer algorithms like a computer RND function or stream ciphers. The second problem is the key distribution. Since each key can only be used once and has to have the same length as the message, we will need a large number of different keys, physically distributed to both sender and receiver.

Of course, it would be useless to send the one-time pads to the receiver by encrypting them with AES, IDEA or another strong algorithm. This would lower the unbreakable security of the pads to the security level of the algorithm, used to encrypt the pads. Therefore, the key distribution creates enormous logistical and security problems if one-time pad is used on large scale. The costs of secure production, distribution, custody and destruction of one-time pad keys are only affordable by government departments such as military, intelligence services and embassies. However, these problems are not an issue when one-time pads are used on a small scale.

Another disadvantage is that one-time encryption doesn't provide message authentication and integrity. Of course, you know that the sender is authentic, because he has the appropriate key and only he can produce a decipherable ciphertext, but you cannot verify if the message is corrupted, either by transmission errors or by an adversary. A solution is to use a hash algorithm on the plaintext and send the hash output value, encrypted along with the message, to the recipient (a hash value is a unique fixed-length value, derived from a message). Only the person who has the proper one-time pad is able to correctly encrypt the message and corresponding hash. An adversary cannot predict the effect of his manipulations on the plaintext, nor on the hash value. Upon reception, the message is deciphered and its content checked by comparing the received hash value with a hash that is created from the received message. Unfortunately, a computer is required to calculate the hash value, making this method of authentication impossible for a purely manual encryption.

One-time Pads in Today's World ▲

In the computer era, modern computer algorithms such as symmetric block ciphers and asymmetric public key algorithms replaced one-time pads because of practical considerations and solution to key distribution problems. However, although the current crypto algorithms are secure, they could become useless because of new hardware developments, a mathematical breakthrough such as a faster de-factoring of primes, new types of attacks or new quantum computing solutions that speed up brute force attack. Modern crypto algorithms provide practical security and privacy, essential to our economy and everyday life. However, sometimes we need ever lasting absolute security and privacy, and that's only possible with one-time encryption.

Some experts argue that the distribution of large quantities of one-time pads or keys is impractical. This was indeed the case in the era of paper tapes on reels and paper pads. However, today's electronics, such as PC cards with hardware random noise generators, are capable of generating large numbers of truly random keys, and current one-time encryption software can process large quantities of data at high speed. Current data storage technology such as USB sticks, DVD's, external hard disks or solid-state drives enable the physical transport of enormous quantities of truly random keys. Companies like [Mils Electronic](#) and [IDQ](#) offer quality one-time-key systems and hardware true random generators.

Actual sensitive communications are often limited to a small number of users. In such cases, one-on-one communications with the associated key distribution, possibly in configuration with a star topology, is no longer a practical problem, especially considering the security benefits. By using a so-called sneaker net (transferring data on removable media by physically couriers), you can reach a throughput (amount of data per unit time) of one-time keys that is greater than what a network can process on encrypted data. In other words, it could take a few hours to drive a terabyte of key material, stored on an external drive, by car to someone, but it will take days or even weeks to consume that amount of keys on a broadband network. A terabyte sized key can easily encrypt you e-mail traffic for a year, including attachments (many Internet providers won't even allow this amount of traffic). Therefore, one-time key encryption is still well-suited in specific circumstances where absolute security is preferred above practical considerations, regardless the cost of secure physical transport of keys by couriers.

Nonetheless, the pencil and paper one-time pad still is a very practical method of encryption where the sender and/or receiver must do all work by hand, without the aid of a computer. Given the many vulnerabilities, frequently found in standard computer systems, widespread viruses, spyware, worms and Trojan Horses, this is an important security benefit if you don't have a physically separate or trusted secure computer at your disposal. You could call it the poor man's one-time pad, but it works perfectly. A perfect crypto algorithm on your computer is useless if spyware logs your keystrokes or captures the plain text and sends it to someone else. A very special type of manual one-time pad encryption can be applied with [Visual Cryptography](#). More about one-time encryption in today's world is found in the paper [Is One-time Pad History?](#)

One-time Pad with Numbers ▲

There are many different ways to apply one-time pads. All of them are absolutely secure if the rules of one-time pad are followed. We can apply one-time pad with numbers or letters. In our first example, we will demonstrate the use of numbers. This is the most flexible system that allows many variations. Usually, encryption is performed by subtracting the random one-time pad key from the plaintext and decryption by adding the ciphertext and key together. Enciphering by addition and deciphering by subtraction works just as good, as long as sender and receiver agree upon using the opposite calculations. However, before we can perform the calculations with the plaintext and key we need to convert the text into digits. There are various ways to do this. A most basic method is to assign a two-digit value to each letter (eg. A=01, B=02 and so on through Z=26).

A popular and more economic way to convert text into digits is a so-called straddling checkerboard. Note that this text-to-digit conversion itself is by no means secure and must be followed by an encryption! A straddling checkerboard converts the most frequently used letters into one-digit values and the other letters into two-digit values. This results in a ciphertext that is

considerably smaller than the basic A=01/Z=26 systems. Various checkerboards exist with different character sets and symbols, optimized for different languages.

The first row of the checkerboards contains the most frequent characters with some blanks between them. The following rows (as many as there were blanks in the top row) contain the remaining letters. These following rows are designated by the digits above the blanks in the top row. Checkerboards are memorized by the top row letters, which can depend on the language it is optimized for. Some example mnemonics are "AT-ONE-SIR" and "ESTONIA---" (English), "DEIN--STAR" and "DES--TIRAN" (German), "SENORITA--" and "ENDIOSAR--" (Spanish), "RADIO-NET-" (Dutch) or "ZA---OWIES" (Polish). Such word combinations are easily composed with an anagram generator. More blanks in the top row gives more additional rows and thus more characters. There's no need to keep this table secret or scramble the order of the digits or letters because one-time encryption follows. More examples of checkerboards are found [on this page](#).

In our example we use a basic checkerboard with the "AT-ONE-SIR" mnemonic, optimized for English.

```

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

```

The top row letters are converted into the one-digit values right above them. All other letters are converted into two-digit values by taking the row header and the column header. To convert figures, we use "FIG" before and after the digits and write out each digit three times to exclude errors.

Let us convert the text "PLEASE CONTACT ME AT 1200H." with the checkerboard

```

Plaintext : P L E A S E   C O N T A C T   M E   A T   [fig] 1 2 0 0 [fig] H .
conversion: 60 28 5 0 7 5   21 3 4 1 0 21 1   29 5 0 1   69 111 222 000 000 69 25 68

```

To encrypt the message, we complete the last group with zero's and write the one-time pad key underneath the plaintext. Since we use digits, the key are plaintext must be calculate the ciphertext by modulo 10. This modulo 10 is essential to the security of the encryption! Therefore, we subtract the key without borrowing (e.g. $3 - 7 = 13 - 7 = 6$, and don't borrow 10 from the digit's next-left neighbor).

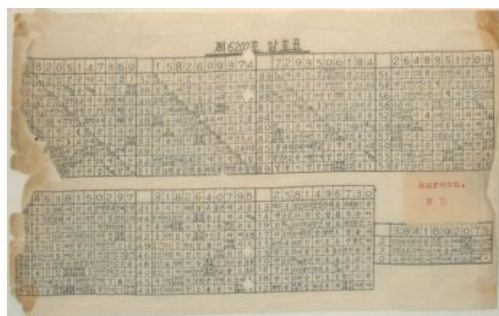
```

Plaintext :      60285 07521 34102 11295 01691 11222 00000 06925 68000
OTP-Key    : (-) 50418 55297 01164 98769 26107 85944 36228 44985 25485
-----
Ciphertext:      10877 52334 33048 23536 85594 36388 74882 62040 43625

```

To decrypt the message, we add the ciphertext and one-time pad key together without carry (e.g. $5 + 7 = 2$ and not 12, and don't carry 10 to next-left digit). Next, we re-convert the digits back into text. It's easy to separate the one-digit values from the two-digit values. If a digit combination starts with row number 2 or 6, it is a two-digit code and another digit follows. In all other cases it's a one-digit code.

Sometimes a codebook or codesheet is used to reduce ciphertext length and transmission time. Such codebook can contain all kinds of words and/or small phrases about message handling and operational, technical or tactical expressions. A codebook system does not always require a large book with thousands of expressions. Even a single code table can contain enough practical information to reduce the message length enormously. Below images of a Korean code table sheet, the instructions on how to convert the table content into digits and how to calculate the ciphertext . Images © Detlev Freisleben Archive ([click to enlarge](#))



Code sheet to convert expressions and words into digits



Instructions

	3	8	4	1	6	9	2	0	7	5
2	것	한	문	한	을	을	은	에	하	한
7	것	한	문	을	에	의	을	에	하	한
0	는	하	는	에	의	의	이	에	하	고

Detail of the text to digit table

A Practical Example with Numbers 🗨️

As a little exercise we will decipher a recording of an actual [numbers station](#) (see important note below). You can open or download (right-click and Save Target As...) the sound file below. The broadcast starts with a repeated call sign melody and the receiver's call sign "39715", followed by six tones and the actual message. All message groups are spoken twice to ensure correct reception. Write down the message groups once (skip the call sign). Once you have the complete message, write the given one-time pad key underneath it. Add message and key together, digit by digit, from left to right, without carry (e.g. $6 + 9 = 5$ and not 15). Finally, convert the digits back into text with the help of the "AT-ONE-SIR" straddling checkerboard as shown in the previous section. Make sure to separate one-digit and two-digit characters correctly.

This little exercise shows exactly how secret agents can receive messages in an absolutely secure manner, with only one-time pads, a small short-wave receiver and pencil and paper.

 [Numbers Station Message](#) (1724 Kb)

The one-time pad key to decipher this message:

```
66153 77185 10800 54937 48159 83271 12892 07132 34987 53954 23074
```

Important Note: Although we use a recording from an actual numbers station (Lincolnshire Poacher, E3 Voice), the one-time pad key is fictitious and reverse-calculated (key = plaintext - ciphertext) so that a readable but fictitious message is obtained when using this key. In reality, we don't know which key was used, whether we must add or subtract and there is no way to decipher the original message. In fact, since a one-time pad key is truly random, one can calculate any plaintext from a given ciphertext, as long as you use the 'right' wrong key. That's exactly why one-time pad is unbreakable.

One-time Pad with Letters

We can also encrypt the plaintext with a one-time key that consists of random letter only. This is done with the help of a Vigenere table. To encrypt a letter, we take the plaintext letter in the column header and the key letter in the row header. The crossing of those two letters is the ciphertext. In the first letter of our example, the crossing between the plaintext T and key X is ciphertext Q. To decrypt a letter, we take the key letter in the row header and find the ciphertext letter in that row. The plaintext letter is the column header above the ciphertext letter. In our example, we take the X row, find the Q in that row and see the plain T on top of the Q. To make it easier to remember, we can consider the horizontal column header of the square as plaintext, the vertical row header as key and the square field as ciphertext.

An example text:

```
Plaintext :  T H I S   I S   S E C R E T
OTP-Key   :  X V H E   U W   N O P G D Z
-----
Ciphertext: Q C P W   C O   F S R X H S

In groups :   QCPWC OFSRX HS
```

The Vigenere square (tabula recta):

```

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

There are several practical solution to the Vigenere table. Click the links to view a [bigram table \(txt file\)](#), [triads table \(txt file\)](#), [Vigenere Table Card](#) and [its ASCII version](#), [Vigenere Disk](#) and [Vigenere Slider](#). All images can be saved by right-clicking and then printed and cut out.

Another way to calculate letter one-time pads without a Vigenere table, although more elaborate, is to perform a modulo 26 calculation. We assign each letter a numerical value (eg. A=0, B=1 C=3 and so on through Z=25). Note that we start with A=0 and not A=1 to enable the use of modulo 26. Text and key values are added together (this time with carry!), with modulo 26: if a value is more than 25, we subtract 26 from that value. Finally, we convert the result back into letters. To decipher the message, we convert the ciphertext and one-time pad key into numerical values and subtract one-time pad key values from ciphertext values, again modulo 26(if a value is less than 0 we add 26 to that value).

```
Plaintext :   T H I S   I S   S E C R E T
              19 07 08 18 08 18 18 04 02 17 04 19
OTP-Key:     X V H E U W N O P G D Z
            +23 21 07 04 20 22 13 14 15 06 03 25
-----
Result:      42 28 15 22 28 40 31 18 17 23 07 44
Mod 26 =     16 02 15 22 02 14 05 18 17 23 07 18
-----
Ciphertext:  Q C P W   C O   F S R X H S

In groups :   QCPWC OFSRX HS
```

You can use a little help table to make the calculations easier. To encrypt by addition we take for example T(19) + X(23). The total of 42 in the conversion table represents the letter Q which is the encryption result. To decrypt by subtracting we take Q(16) - X(23). If the result would give a negative value (which is the case here) we take the greater equivalent of Q(16), which is (42) in the conversion table. We can now find the deciphered letter with Q(42) - X(23) = T(19)

MODULO 26 HELP TABLE

```

  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
+-----+
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 --
```

There are many more variations of one-time pad but they all use the same principle that a random value is added and afterwards subtracted or the other way around. For more information on how to use one-time pad, please visit the [How to use manual one-time pads](#) page.

Secret Splitting ▲

There is a special way to use one-time pad where the key is not to be destroyed. When information should be available only when two people agree to reveal that information, we can use secret splitting. The secret information is encrypted with a single one-time pad whereupon the original plaintext is destroyed. One user receives the encrypted message and the other user the key. In fact, it doesn't matter who gets which, since both pieces of information can be seen as equal, encrypted parts of the original information. The split parts are both called keys. Both these keys are useless without each other. This is called secret splitting. One could encrypt for example the combination to a safe and give the split ciphertext to two different individuals. Only when they both agree upon opening the safe, will it be possible to decipher the combination to the safe. You could even split information into three or more pieces by using two or more keys.

In this little example Charlie splits his secret safe combination 21 46 03 88. A random key is subtracted digit by digit, without carry, from the combination numbers. Alice and Bob both receive one piece of the information from Charlie. It's mathematically impossible for both Alice and Bob to retrieve the combination numbers unless they share their keys. This is done by simply adding the keys (without carry).

```
Charlie's Combination:  21 46 03 88
Random one-time key    - 25 01 77 61
                      -----
                      06 45 36 27
```

Alice's key = 25017761

Bob's key = 06453627

Of course, we could also use secure splitting on text to encrypt passwords and such. Just convert the text into numbers (e.g., A=01, B=02 and so on through Z=26). To split the secret into more parts, just add a one-time key for each of the new persons. For three persons you must subtract two keys (without carry) from the plaintext to obtain the ciphertext (e.g. $2 - 4 - 9 = 9$. Because $2 - 4 = 12 - 4 = 8$ and $8 - 9 = 18 - 9 = 9$). Instead of keeping your secret password in an envelop, you could split it and give the shares to different persons, of which at least one is trusted. One person could never act on his own and approval of a second person is always required. When granddad, old and sick, splits the secret combination from the safe that contains his money and gives each of his children one part, they can only get their hands on his money if they all agree (not that this will make him live longer).

However, since this system is unbreakable, all information is lost if one of the shares goes missing. There's no way back if a share is lost or destroyed by accident! It might be useful to have one extra copy of your share somewhere on a secure location.

More about Secret Splitting [on this page](#).

About Modular Arithmetic ▲

Modular arithmetic has interesting properties that play an important role in cryptography and it is essential to the security of one-time pad encryption. The result of an encryption process could reveal information about the key or the plaintext. Such information might either point to possible solutions or enable the codebreaker to discard some wrong assumptions. The codebreaker will use this information as a lever to break open the encrypted message. By using modular arithmetic on the result of a calculation we can obscure the values that were used to calculate that result.

Modular arithmetic works similarly to counting hours, but on a decimal clock. If the hand of our clock is at 7 and we add 4 by advancing clockwise, we pass the 0 and arrive at 1. Likewise, when the clock shows 2 and we subtract 4, advancing anticlockwise, we arrive at 8. In mathematics, modulo x is the remainder after the division of a positive number by x . Some examples: 16 modulo 12 = 4 because 16 divided by 12 is 1 and this leaves a remainder of 4. Also, 16 modulo 10 is 6 because 16 divided by 10 is 1 and thus leaves a remainder of 6.



Modular arithmetic is very valuable to cryptography because the result value reveals absolutely no information about the two values that were added or subtracted. If the result of a modulo 10 addition is 4, we have no idea whether this is the result of $0 + 4$, $1 + 3$, $2 + 2$, $3 + 1$, $4 + 0$, $5 + 9$, $6 + 8$, $7 + 7$, $8 + 6$ or $9 + 5$. The value 4 is the result of an equation with two unknowns, which is impossible to solve.

The modulus should have the same value as the number of different elements that need to be calculated, with 0 designated to the first element. Thus, for bits (0 or 1) we use modulo 2 and for bytes (8-bit values between 0 and 255) we use modulo 256 (in Boolean arithmetic, we call this an XOR operation). For digits (0 through 9) we use modulo 10. In practice, modulo 10 is easy to perform by adding without carry and subtracting without borrowing, which basically means discarding all but the most-right digit of the result. For our one-time pad encryption with numbers, it could not be easier.

Performing modulo calculations on letters needs some additional explanation. One would tend to assign the numbers 1 through 26 to the letters and then apply modulo 26. However, because the result of a modulo calculation can be zero ($26 \bmod 26 = 0$), the first element should always be regarded as zero. Thus, we must assign the values 0 through 25 to the letters A through Z and then use modulo 26 (that's also why we used 0 through 11 on our clock, and not 1 through 12). Modulo 26 is a bit more complicated to calculate than modulo 10, but the Vigenere Square is a practical way to perform modulo 26.

We will explain the need for modular arithmetic with some small examples: with normal addition, the ciphertext result 0 can only mean that both key and plaintext have the value 0. A ciphertext result of 1 means that the two unknowns can only be $0 + 1$ or $1 + 0$. With result 2, the unknowns can only be $0 + 2$, $1 + 1$ or $2 + 0$. Thus, for some ciphertext result values we can either immediately determine the unknowns or we can see which unknowns of the equation could be possible or impossible. Suppose we combine the letter X (23) with the truly randomly selected key Z (25). With modulo 26, the result would be 22 (W) because $(23 + 25) \bmod 26 = 22$. This value does not reveal anything about plaintext or key. However, with a normal calculation the result will be 48. Although both plain letter and truly random key are unknown, we can draw some important conclusions: the total of 48 is only possible with combinations X (23) + Z (25), Y (24) + Y (24) or Z (25) + X (23). By merely looking at the ciphertext, we can discard all letters A through W as possible candidates for both plaintext and key. This is an important clue for the codebreaker. Of course, there are various ways to screw up encryption by not applying modular arithmetic, but they will all create a biased ciphertext instead of a random ciphertext. To codebreakers, bias is as valuable as gold.

Is One-time pad Unbreakable? ▲

If all rules of one-time pad are followed? Yes! When a truly random key is combined with a plaintext, the result is a truly random ciphertext. To find key or plaintext, an adversary only has the random ciphertext at his disposal. This is an equation with two unknowns, which is mathematically unsolvable. Also, since each key digit or letter is truly random, there is no mathematical or logical relation whatsoever between the individual ciphertext characters. The modulo 10 (for one time pad digits) or modulo 26 (for one-time pad letters) also ensures that the ciphertext does not reveal any information about the two unknowns in the equation.

If someone had infinite computational power he could go through all possible keys (a brute force attack). He would find out that applying the key XVHEU on ciphertext QJKES would produce the (correct) word TODAY. Unfortunately, he would also find out that the key FJRAB would produce the word LATER, and even worse, DFPAB would produce the word NEVER. He has no idea which key is the right one. In fact, you can produce any desired word or phrase from any one-time pad -encrypted message, as long as you use the 'right' wrong key. There is no way to verify if a solution is the right one. Therefore, the one-time pad system is proven completely secure.

The enciphered word:

```
Plain text:  T O D A Y
OTP-Key:    + X V H E U
            -----
```

Ciphertext: = Q J K E S

The deciphered word, with one correct and two wrong one-time pad keys:

```
Ciphertext:  Q J K E S      Q J K E S      Q J K E S
OTP-Key:    - X V H E U    - F J R A B    - D F P A B
-----
Plain text: = T O D A Y    = L A T E R    = N E V E R
```

The one-time pad encryption scheme itself is mathematically unbreakable. Therefore, the attacker will focus on breaking the key instead of the ciphertext. That's why a truly random key is essential. If the key is generated by a deterministic algorithm the attacker could find a method to predict the output of the key generator. If for instance a crypto algorithm is used to generate a random key, the security of the one-time pad is lowered to the security of the used algorithm and is no longer mathematically unbreakable. If a one-time pad key, even truly random, is used more than once, simple cryptanalysis can recover the key.

Indeed, although the ciphertext result of a truly random key is a truly random ciphertext, using the same key twice will result in a relation between the two ciphertexts and consequently also between the two keys. The different ciphertext messages are no longer truly random and it's possible to recover both plaintexts by heuristic analysis. Another unacceptable risk of using one-time pad keys more than once is the known-plaintext attack. If the plaintext version of a one-time pad encrypted version is known, it is of course no problem to calculate the key. This means that if the content of one message is known, all messages that are encrypted with the same key are also compromised.

Breaking a Reused One-time pad ▲

Using a one-time pad more than once will always compromise the one-time pad and all ciphertext, enciphered with that one-time pad. To exploit reused one-time pads we can use a heuristic method of trial and error. This simple method enables the complete, or at least partial, deciphering of all messages. This can even be done with pencil and paper, although it is a slow and cumbersome process. The principle is as follows: a crib, which is a presumed piece in the first plaintext, is used to reverse-calculate a piece of the key. This presumed key is then applied at the same position on the second ciphertext. If the presumed crib was correct then this will reveal a readable part of the second ciphertext and provide clues to expand the cribs. In the following example we will demonstrate the breaking of two messages, only with the aid of pencil and paper.

We have two completely different ciphertext messages, "A" and "B". They are both enciphered with the same one-time pad, but we have no knowledge of that key. Let us begin with assuming that the letters are converted into digits by assigning them the values A=01 through Z=26, that the enciphering is performed by subtracting the key from the plaintext without borrowing ($5 - 8 = 15 - 8 = 7$) and that deciphering is performed by adding ciphertext and key together without carry ($7 + 6 = 3$ and not 13). This is a standard and unbreakable application of one-time pad, if only they had never used that one-time pad twice! The reason I use the basic A=01 to Z=26 is to make it easier to see the separate letters. The described heuristic analysis works also with a straddling checkerboard (one-digit and two-digit conversions).

```
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
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
```

Ciphertext A: 69842 23475 84252 16490 45441 18956 51010 4

Ciphertext B: 55841 41281 75131 05995 61489 69256 61

First, we must search for a crib. A crib is an assumed piece of plaintext that corresponds to a given ciphertext. These can be commonly used words, parts of words, or frequently used trigrams or bigrams. Some examples of frequent trigrams in the English language are "THE", "AND", "ING", "HER" and "HAT". Frequent bigrams are "TH", "AN", "TO", "HE", "OF" and "IN". Of course, a crib should be as long as possible. If you know who sent the message and what he might be talking about you could try out complete words.

In our example, we don't have any presumed words, so we'll have to use some other group of letters. Let's try the crib "THE", which is the most frequently used trigram in the English language. Now, in this example we only have one small piece of ciphertext. In real life, you might have a few hundred digits at your disposal for testing, which makes a successful crib more likely.

We align the letters "THE" with every position of ciphertext "A" and subtract the ciphertext from the crib. The result is the assumed one-time key. In heuristic terms, this is our trial. To test it, we add the assumed key to ciphertext "B" to recover plaintext "B". Unfortunately, as shown underneath the first "THE" of the example, we get our heuristic error. We continue to try out all positions. For the sake of simplicity, I only show three example positions of the crib. Our trial and error will show us that the 9th character position (17th digit) provides a possible correct plaintext "B", the trigram "OCU".

```

CHECKING "THE" CRIB

Crib on A          T H E          T H E          T H E
                  20 08 05          20 08 05          20 08 05
Ciphertext A      -69 84 22 34 75 84 25 21 64 90 45 44 11 89 56 51 01 04
-----
Presumed Key       46 86 71          66 18 60          41 52 54
Ciphertext B      +55 84 14 12 81 75 13 10 59 95 61 48 96 92 56 61
-----
Presumed Plain B   20 90 83          15 03 21          33 08 15
                  T ?? ??          O C U           ?? H O
                  (impossible)      (possible)      (impossible)
```

There are a few, but not too many, solutions to complete this "OCU" piece of plaintext, and we'll have to try them all out. So, let's try out the obvious "DOCUMENT". This assumption has to pass our trial and error again. Therefore, here below, we use "DOCUMENT" as a crib for plaintext "B" at exactly the same place. We subtract ciphertext B from the assumed plaintext "DOCUMENT" to again recover a new portion of the presumed key. Our presumed key is now already expanded to 16 digits.

We add this presumed key to ciphertext "A" to hopefully recover something readable and indeed, "OTHESTAT" could well be a correct solution, thus confirming the used crib. Can we make this crib any longer? "THE STAT" could be part of "THE STATUS", "THE STATION" or "THE STATIC", and "O THE" might be expandable to "TO THE", as "TO" is a popular bigram that ends with the letter O. Again we must test these solutions by recovering the related assumed key and try that key out on the other ciphertext. If correct, this will again reveal another little readable piece of plaintext. Remember we started only with the assumption that there could be a "THE" in one messages and already end up with "DOCUMENT" and "TO THE STAT..." after only two heuristic steps!

```

CHECKING "DOCUMENT" CRIB

Crib on B          D O C U M E N T
                  04 15 03 21 13 05 14 20
Ciphertext B      -55 84 14 12 81 75 13 10 59 95 61 48 96 92 56 61
-----
```



```

Presumed Key          94 66 18 60 75 19 22 74
Ciphertext A    +69 84 22 34 75 84 25 21 64 90 45 44 11 89 56 51 01 04
-----
Presumed Plain A          15 20 08 05 19 20 01 20
                  . . . O T H E S T A T . . .

```

This process is repeated over and over. Some new cribs will prove to be dead end and others will result in readable words or parts of words (trigrams or bigrams). More plaintext means better assumptions and the puzzle will become easier and easier. Thanks to the two ciphertexts, you can verify the solutions of one plaintext with its counterpart ciphertext, over and over again, until the deciphering is completed.

Finally, we'll give the solution, just to verify the results of our trial and error:

```

                        THE ORINIGAL MESSAGES

Plaintext A      R E T U R N T O T H E S T A T I O N
                  18 05 20 21 18 14 20 15 20 08 05 19 20 01 20 09 15 14
KEY              -59 21 08 97 43 30 05 94 66 18 60 75 19 22 74 58 14 10
-----
Ciphertext A      69 84 22 34 75 84 25 21 64 90 45 44 11 89 56 51 01 04

Plaintext B      D E L I V E R D O C U M E N T S
                  04 05 12 09 22 05 18 04 15 03 21 13 05 14 20 19
KEY              -59 21 08 97 43 30 05 94 66 18 60 75 19 22 74 58
-----
Ciphertext B      55 84 14 12 81 75 13 10 59 95 61 48 96 92 56 61

```

Little fragments like, for example, "FORMA" is easily expanded to "INFORMATION", gaining 6 additional letters as a crib. "RANSP" is most likely "TRANSPORT" or, with some luck, "TRANSPORTATION", providing 9 additional letters, a quite large crib. Sometimes, the already recovered text provides clues about the words that precede or follow them, or will help to get ideas for words on other places in the message. It's a slow and tedious process, but the patchwork will gradually grow. Slow, cumbersome and tedious pays off in this line of work. This method is also usable when the text is converted into digits with a straddling checkerboard or any other text-to-digit conversion systems.

Of course, this example is short and simple. In reality, there could be all kinds of complications that require many more trials. What system is used to convert text into digits? What language is used? Did they use abbreviations or slang? Are there words available as cribs or do we need to piece together trigrams or even bigrams until we have a word to get launched? Does the message contain actual words or are there only codes from a codebook? Is the one-time pad reused completely or only partially, and do they start at the same position in both messages? All these problems can slow down the heuristic process and require a vast number of trials, with associated dead ends and errors, before the job is done. Success is not guaranteed, but in most cases, the reuse of one-time pads will result in a successful deciphering. This is certainly the case with today's computer power, enabling fast heuristic testing.

History has shown many examples of negligent use of one-time pad, the VENONA project being the most notorious. This is a fine example of how important it is to follow the basic rules of one-time pad. Soviet Intelligence historically always relied heavily on one-time pad encryption, with good reason and success. Soviet communications have always proved extremely secure. However, during the Second World War, the Soviets had to create and distribute enormous quantities of one-time pad keys. Time pressure and tactical circumstances lead in some cases to the distribution of more than two copies of certain keys. In the early 1940s, the United States and Great Britain analyzed and stored enormous quantities of encrypted messages, intercepted during the war.

American codebreakers discovered by cryptanalysis that a very small portion of the tens of thousands of KGB and GRU messages between Moscow and Washington were enciphered with reused one-time pads. The messages were encoded with codebooks prior to enciphering with one-time pad, making the task even immensely harder for the codebreakers. Finding out which key was reused on what message, the reconstruction of the codebooks and recovering the plaintext were enormous challenges that took years. Eventually they managed to reconstruct more than 3,000 KGB and GRU messages, just because of a distribution error by the Soviets. VENONA was crucial in solving many spy cases. Although VENONA is often mistakenly referred to as the project that broke Soviet one-time pads, they never actually broke one-time pad, but exploited implementation mistakes as described above.

Make no mistake! It will never be possible to break one-time pad if properly applied. This example only shows how to exploit the most deadly of all mistakes: reusing a one-time pad.

Random Numbers ▲

The use of a truly random key, as long as the plaintext, is an essential part of the one-time pad. Since the one-time algorithm itself is mathematically secure, the codebreaker cannot retrieve the plaintext by examining the ciphertext. Therefore, he will try to retrieve the key. If the random values for the one-time key are not truly random but generated by a deterministic mechanism or algorithm it could be possible to predict the key. Thus, selecting a good random number generator is the most important part of the system.

In the pre-electronic era, true random was generated mechanically or electro-mechanically. Some of the most curious devices were developed to produce random values. Today, there are several options to generate truly random numbers. Hardware Random Number Generators (RNG's) are based on the unpredictability of physical events. Some semiconductors such as Zener diodes produce electrical noise in certain conditions. The amplitude of the noise is sampled at fixed time intervals and translated into binary zeros and ones.

Another unpredictable source is the tolerance of electronic component properties and their behavior under changing electrical and temperature conditions. Some examples are ring oscillators that operate at a very high frequency, the drift, caused by resistors, capacitors and other components in oscillators or time drift of computer hardware. Photons, single light particles, are another perfect source of randomness. In such systems, a single photon is sent through a filter, and its state is measured. The quality of such randomness sources can be verified with statistical tests to detect failure of the system.

Even when hardware-based true random generators are used, it will be necessary in some cases to improve their properties, for instance to prevent unequal distribution of zero's or one's in a sequence. One simple way to improve or whiten a single bit output is to sample two consecutive bits. The value sequence 01 would result in an output bit 0 and the value sequence 10 would give output 1. The repetitive values 00 and 11 are discarded. Some hardware RNG's are the [Mills Generator](#) with a combination of several ring oscillators, the [Quantis QRNG](#), based on the unpredictable state of photons, the CPU clock jitter based [ComScir PCQNG](#) generator, and the [VIA Nano processor](#) with its integrated dual quantum RNG's.

Another option is the manual generation of numbers. Of course, this time consuming method is only possible for small volumes of keys or key pads. Nevertheless, it's possible to produce truly random numbers. You could use five ten-sided dice (see image right). With each throw, you have a new five-digit group. Such dice are available in toy stores or you could make them yourself ([dice template](#)).

Never ever simply use normal six-sided dice by adding the values of two dice. This method is statistically unsuitable to produce values from 0 to 9 and thus absolutely insecure (the total of 7 will occur about 6 times more often than the values 2 or 12). Instead, use one black and one white die and assign a value to each of the 36 combinations, taking in account the order/color of the dice (see table below). This way, each combination has a .0277 probability (1 on 36). We can produce three series of values between 0 and 9. The remaining 6 combinations (with a black 6) are simply disregarded, which doesn't affect the probability of the other combinations.

TRUE RANDOM 0 TO 9 WITH BLACK AND WHITE DICE

BW	BW	BW	BW	BW
11 = 0	21 = 6	31 = 2	41 = 8	51 = 4
12 = 1	22 = 7	32 = 3	42 = 9	52 = 5
13 = 2	23 = 8	33 = 4	43 = 0	53 = 6
14 = 3	24 = 9	34 = 5	44 = 1	54 = 7
15 = 4	25 = 0	35 = 6	45 = 2	55 = 8
16 = 5	26 = 1	36 = 7	46 = 3	56 = 9



THROWS WITH BLACK 6 ARE DISCARDED

You could also assign the letters A through Z and numbers 0 through 9 to all 36 dice combinations, again taking in account the order/color as in the table above. This way, you can create one-time pads that contain both letters and numbers. Such one-time pads can be used in combination with a Vigenere square, similar to the one described above, but with a 36 x 36 grid where each row contains the complete alphabet, followed by all digits. This will also produce a ciphertext with both letters and numbers. An advantage is that your plaintext can contain figures.

You can also use lotto balls. However, after extracting a number, that ball must always be mixed again with the other balls before extracting the next ball. If random bit values are required you can use one or more coins that are flipped, with one side representing the zero's and the other side the one's. With 8 coins you could compose an 8 bit value (byte) in one throw. Many other manual systems can be devised, as long as statistical randomness is assured. These simple but effective and secure methods are suitable for small one-time pads or small keys that are used to protect passwords (see [Secret Splitting](#)).

Another alternative is the use of a software based generator. However, software random number generators will never provide absolute security because of their deterministic nature. Crypto secure pseudo-random number generators (CSPRNG's) produce a random output that is determined by a key or seed. A large (unlimited) amount of random values is derived from a seed or key with a limited size, and seed and output are related to each other. In fact, you're no longer using one-time encryption, but an encryption with a small sized key. Brute forcing the seed by trying out all possible seeds, or analysis of the output or parts of the output could compromise the generator.

There are techniques to improve the output of CSPRNG's. Using a truly random and very large seed is essential. This could be done by accurate time or movement measurements of human interaction with the computer, for instance mouse movements, or by measuring the drift of computer processes time (note that a normal computer RND function is totally insecure). Another technique to drastically improve a CSPRNG is to combine the generator output with multiple other generators, the so-called "whitening". This will make analysis of the output much more difficult because each generator output obscures information about the other generator outputs. In the end, however, only one time pad encryption, based on truly random keys, is really unbreakable. More information about the secure generation of randomness is found in the IETF [RFC 1750 Randomness Recommendations for Security](#).

There's also the issue of secure computers to process, store or print the truly random numbers. Even the use of a hardware generator with truly random output, necessary for absolute security, is useless if the computer itself is not absolutely secure. Unfortunately, there's no such thing as a secure personal computer. The only absolutely secure computer is a physically separated computer, with restricted input/output peripherals, never connected to a network and securely stored with controlled access. Any other computer configuration will never guarantee absolute security. Cryptographic software is only secure on a stand-alone computer or dedicated crypto equipment.

Although a good CSPRNG theoretically never achieves Shannon's perfect secrecy, it can be useful to generate pads that are not theoretically but practically secure. You can [download Numbers 8.3](#), a small program that generates and prints random series of numbers or letters in various formats. Note that this software does not create unbreakable one-time pads unless external true random is loaded into the program.

More on One-time Pad ▲

- [Numbers Stations](#)
- [Secret Splitting](#)
- [Cuban Agent Communications](#) 📄 Paper on Cuban numbers stations, Cuban agents in the U.S. and the errors they made
- [Guide to Secure Communications with the One-time Pad Cipher](#) 📄 how to use one-time pads and set up secure communications with them
- [Spies and Numbers - Here to Stay](#) 📄 The use of one-time pads in espionage
- [Is One-time Pad History?](#) 📄 Usability of one-time pad in today's world
- [TEMPEST](#) How a one-time tape mixer lead to suppression of spurious signals
- [Operation Tinker Bell](#) Decipher one-time pad messages during the hunt for a KGB defector
- [The Numbers Relay Page](#) an online method to post numbers messages.

Off-Site ▲

- [One-Time Letter Pads and One-Time Figure Pad](#) on Jerry Proc's [Cryptomachines website](#)
- [A list of East German one-time pad systems](#) on the [SAS und Chiffrierdienst](#) website (in German)
- [One-time pad on the Crypto Museum website](#)
- [David Kahn's The Codebreakers](#) standard work on cryptography (see [my book reviews](#))
- [Steven Bellovin's paper](#) on Frank Miller's 1882 invention of one-time pad
- **NEW** [BAPCO's Use of One Time Pad During WW2](#) my blog on one of the earliest official records of one-time pad encryption by a commercial firm.
- **NEW** [ANC's DTMF payphone messages](#) How the African National Congress sent encrypted messages in the fight against apartheid

One-time Pad Tool ▲

You can download the freeware CT-46 one-time pad tool. This small program is a tool to exercise one-time pad encryption. It uses the CT-46 conversion table to convert text into digits. The software includes a help section with instructions on how to perform one-time pad encryption with pencil and paper.

Runs on Windows™ 98/ME/2000/XP/Vista/Win7 and with WINE on Linux or Parallels Desktop on MAC.

📦 [CT-46 OTP v1.0.1 Full Install Including run-time files \(Zip 1426 Kb\)](#)

📦 [CT-46 OTP v1.0.1 Program exe only, without run-time files \(Zip 31 Kb\)](#)

Please check the readme file before installation.

