ARKAKAPI

Bimonthly Cyber Security Magazine

03 January - February www.arkakapimag.com

The 10 Biggest Hacks of 2018

Mini Threat Intelligence

KRACK (Key Reinstallation Attack)

Daniel Bohannon Interview

The Role and Comparison of Operating Systems in Mass Surveillance

Key Exchange Problem in Public Key Cryptography and Keybase

Greetings!

Last month on 10 December it was the 70th anniversary of the proclamation of the Universal Declaration of Human Rights; the purest form of a saga humanity had been breaking its neck to write - from Spartacus to today.

Especially there are such two articles of this declaration that us, Arka Kapi Magazine, have embraced as one of its most significant norms.

The first one is about the privacy of one's life, as stated in Article 12:

"No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Every- one has the right to the protection of the law against such interference or attacks."

These are so important that states should secure this special occasion by mobilizing all legal means.

This is why we have been emphasizing anonymity since the first issue. Our another important red line lies on the opinion and expression freedom - the Article 19:

"Everyone has the right to freedom of opinion and expression; this right includes freedom to hold opinions without interference and to seek, receive and impart information and ideas through any media and regardless of frontiers."

Without the freedom of expression, it is needless to say that the freedom of opinion would have no meaning. Lacking the expression freedom, would we differ in any way from The Thinking Man statue of Rodin? That means that it should be among our main issues that an individual should not only be able to think, but also should be able to express these thoughts without being under pressure, while not being deprived of the right to spread it.

Another means of fundamental human rights is the right to internet access (also mentioned as right to broadband or freedom to connect). As humans, we all have the right to access internet and remain private online.

In the third issue, we are going to talk about some of the attack that threaten our individuality. You will find articles on the dangers and working principles of various systems and attacks, as well as some offline systems and more.

We aim so that you may predict where the attack may come from to protect yourself- and your rights!

Special thanks to Netsparker Ltd. for sponsoring this issue!

Ziyahan Albeniz - Cansu Topukçu
editor@arkakapimag.com

ARKAKAPI MAG

Cyber Security Magazine YEAR: 1 – NOV-DEC ISSUE: 2 Bimonthly - ISSN: 2645-906X www.arkakapimag.com

Editor in Chief: Ziyahan Albeniz • ziyahan@arkakapimag.com

Editorial Operations Manager: Cansu Topukçu • cansu@arkakapimag.com

Chief Business Officer: Oğuz Aydınyılmaz • oguz@arkakapimag.com

Publishing Coordinator: Şahin Solmaz • sahin@arkakapimag.com

Director of Web: Ömer Çıtak • omer@arkakapimag.com

Legal Advisor: Mehmet Pehlivan • mehmet@arkakapimag.com

Assistant research editor: Ayşenur Burak • nurayse47@gmail.com

Translators: Cansu Topukçu, Serdar Savaş, Emre İyidoğan, Enes Özen, Zekvan Arslan, Recep Kızılarslan, Hiva Rashvand

Social Media: twitter.com/arkakapimag instagram.com/arkakapimag facebook.com/arkakapimag

CONTENT

Ayşenur Burak • nurayse@gmail.com

Cyber Security Conferences

WORKSHOP ON SECURITY ISSUES IN CYBER-PHYSICAL SYSTEM (SECCPS)

January 3 - 5, 2019

Hangzhou Shujiang Hotel, China

Cyber attacks are not limited to computers only. Cyber Physical Systems (CPS) deals with the security of embedded systems, Internet of Things, SCADA Systems, Water Systems, and Smart-Grid Systems.

Info: *http://cloud.hdu.edu.cn/hase2019/SecCPS.html*

SANS BANGALORE 2019

January 7, 2019

Le Meridien Bangalore, India

You can register for current courses on Tactical Analytics, Network Penetration Test and ICS/SCADA Security Essentials.

Info: *https://www.sans.org/event/bangalore-january-2019*

REAL WORLD CRYPTO 2019

January 9 - January 11, 2019

San Jose, CA, USA

Real World Crypto 2019 will take place in San Jose Marriott, San Jose, USA on January 9-11, 2019. Real World Crypto 2019 is organized by the International Association for Cryptologic Research (IACR).

Info: *https://rwc.iacr.org/2019/*

SHIELDAFRICA 2019

January 21 - 24, 2019

School National Police De Côte D'ivoire, South Africa

SHIELDAFRICA
Abidjan 2019

It is an International Security and Defence exhibition that addresses the challenges of the African continent. The 2019 topic: "Border Protection and Control".

Info: *https://www.shieldafrica.com/*

CYBERTECH TEL AVIV 2019

January 28, 2019

Tel Aviv, Israel

Featured topics from across a wide range of sectors such as AI, IoT, FinTech, blockchain, privacy and GDPR, data protection, Identity Trust, cyber for the health, retail, smart mobility, communication, mobile, railways, aviation industries and more.

Info: *http://cybertechisrael.com/*

CYBERTECH
THE EVENT FOR THE CYBER INDUSTRY

DC4420 – 2019

February 26, 2019

The Phoenix, London, United Kingdom

DEF CON Local London chapter of the annual Defcon hacker convention.This event is free for all.

Info: *https://dc4420.org/about/*

HIMSS19

February 11 - 15, 2019

Orange County Convention Center, Orlando FL, United States

It is a worldwide conference that contributes to health information and technology.

HIMSS19
Global Conference & Exhibition
FEB 11–15, 2019 | ORLANDO

Info: *https://www.himssconference.org/*

CYBER INTELLIGENCE ASIA 2019

February 26-28, 2019

Bangkok, Thailand

The event brings together leading cybersecurity officials from across Asia-Pacific to present case studies of recent cyber crimes they have faced and how they responded to the threat/crime.

Info: https://www.asdevents.com/event_register.asp?id=19160

Bayram Gök • bayram@arkakapidergi.com

Cryptology in Industrial Revolution

In a way, cryptology has a history dependent on development of communication and transportation devices. The nineteenth century was a productive time period in which many communication tools we continued to use were developed. The traditional period in which the communication was provided through special messengers, mails and letters showed a great change in the 19th century. Significant governmental, commercial, political and military communications were mostly provided by special messengers. The message got delivered in a secure and confidential way provided by the messenger. While being transferred from point to point with with telegraph, radio and telephony, the message could have been captured while being transferred from point to point by getting the signals from the air or by connecting externally to the transport cables.

Those who are forty years and older will remember the periods when the telephone lines passed through the front of the windows as a wire cable, two pins connected to a telephone were inserted into the cable, the line would be listened, and the line would be usable. Fortunately, cables were taken underground, call logs were added to the bills and we got rid of this problem.

It was inevitable for cryptology not to change in the 19th century. New and powerful methods needed to be developed.

19th Century and New Communication Technologies

In 1835, the American painter Samuel Morse prepared the first telegraph system consisting of a simple electromagnet. Later, Morse and his assistant Vail developed the mechanism and created the Morse alphabet, which is still in use, consisting of dots and lines. The first telegraph line was Washington DC in 1843,z from Baltimore to Maryland.

By the 19th century, electricity was well known. In 1864, the Scottish theoretical physicist and mathematician James Clerk Maxwell published "*Maxwell A Dynamical Theory of the Electromagnetic Field*". With the four Maxwell Equations named after himself, Maxwell showed that electricity and magnetism are the same thing, they can transform into each other, the electric and magnetic fields move through the speed of light in space, and the light behaves like a wave. We owe everything from GSM phone system to fiber optic networks, from Wifi to 4G internet connections, from space to satellite communication.

Alexander Graham Bell and Charles Sumner Tainter went down in history as the first ones to make telephone calls on February 15, 1880. In 1891, the first automatic telephone call was made without the need of an operator. Then in 1892 the first long-distance telephone line in Chicago and New York was established.

Born in Bologna Italy, Marconi tried the first wireless telegraph in his house in 1894. Unable to find the necessary support in Italy, Marconi went to England in 1896. His work attracted great interest here. He set up radio stations on the Atlantic coast. On 18 January 1903, the US President Theodore Roosevelt was able to convey a message to King Edward VII of England on October 17, 1907, and a regular radio communications service was launched between Clifden Ireland and Glace Bay Canada.

Zimmermann Telegram

There is such an event I believe that if I examine the details of, the reader would understand the subject of this article easily. This event, known as the Zimmermann Telegram, needs to be studied with a multidisciplinary approach having Cryptology in it's heart. Let me tell you a little bit of history and some politics.

During the First World War, Arthur Zimmermann was the Minister of Foreign Affairs of the German Empire from 22 November 1916 until when he resigned on 6 August 1917. The effects of some of the events he triggered in that time still continue today.

From the beginning of the First World War, Germany had sought to block Britain by sinking British ships using submarines. The aim was to prevent Britain from providing logistics and replenishment from the US and other overseas colonies. Despite the 1258 passengers and 701 crew members, even the British cruise ship Lusitania was sunk by the German submarine off Ireland on May 7 after the setting off of the Atlantic Ocean from New York on May 1, 1915. 1198 people, including 124 US citizens, lost their lives but 761 were rescued. The United States, despite all of the casualties and the provocations of England, did not enter the war, and remained committed to the principles of Wilson and even offered peace without a winner to all sides.

Germany went further and decided to sink all ships, including US ships, in its allied waters. The US ambassador of Germany, John Von Bernstorff, issued this decision to the US government on January 31, 1917.

The German government feared that this new submarine war occured after the sinking of Lusitania would drive the US into the war and was looking for ways to keep the US busy and away from Europe.

On January 19, 1917, Zimmermann sent a telegram to the German Embassy in Washington to be forwarded to the German Ambassador of Mexico, Heinrich von Eckardt. In his message, Zimmermann talks about Germany's decision to start a new submarine war, and if the US enters the war, he wants the Mexican government to join the war alongside Germany. Mexico had lost its Texas, New Mexico and Arizona states as a result of the war with the US. Zimmermann promised Mexico an unlimited support by the German government and the retrieval of Texas, New Mexico and Arizona if they declared a war against United States. Mexico did not accept this proposal and refused to enter the war.

The password of the telegraph was broken by the intelligence unit called Room 40 and delivered to US President Woodrow Wilson. On February 28, 1917, President Wilson distributed the telegraphic text to the press. Germany, on the other hand, asserted that the telegraph was a fake, but on 29 March 1917, Zimmermann admitted that the telegram was real.

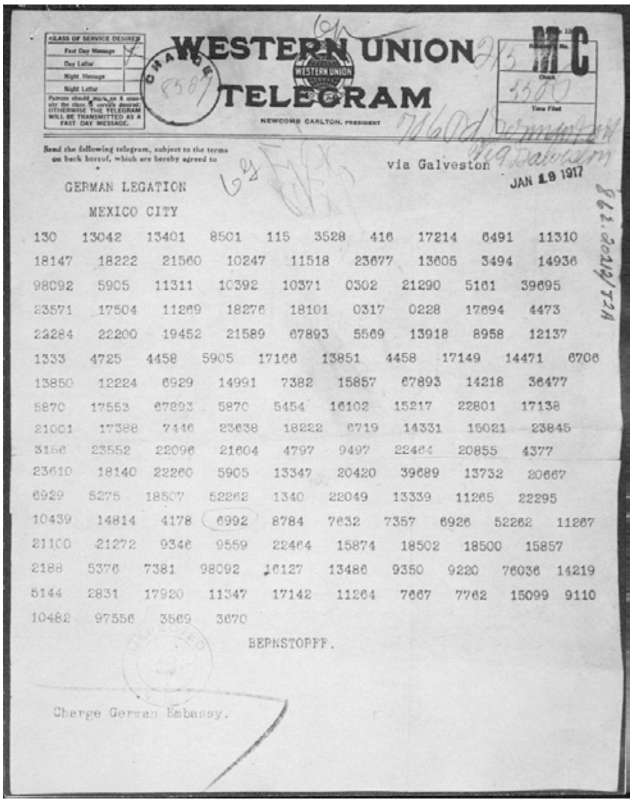
At the beginning of the war, the British destroyed the German submarine cables and cut off communications with the United States. Being left without diplomatic (red line) communication channels, the German Embassy was given permission to use the diplomatic lines of Washington by President Wilson, as long as they used it for peaceful purposes. Thanks to this treaty, Germany communicated with the ambassador of Washington via the US embassy in Germany. The line that emerged from the US embassy first reached a relay station in Denmark's Porthcurno town. In this relay station, the fortified line crossed the ocean and connected to the American continent.

Zimmermann believed that Americans would not be interested in the content of the telegraph, and trusted that his password could not be resolved. Though the US ambassador initially refused to forward the encrypted message, he then agreed. The Germans did not promise to use the line they would give to the US for peaceful purposes and they used the US diplomatic line against the US. Indeed, Americans were unaware of the content of the telegraph. But the British were listening and analyzing all the traffic at the Porthcurno relay station.

The British had already obtained the details of the German diplomatic code 13040 and the German military maritime code 0075. They partially solved the message within a day. However, the content of the telegram was reached in a moral and illegal way by listening to US diplomatic lines. Another problem was that the telegraph had to prove to the US that it was not a fake. They were supposed to make up a story, find a cover. Zimmermann's telegram was sent to the US with the code 0075.

The British knew that a telegraph would be sent to the German Embassy in the US from the US Embassy in Mexico. The British ambassador in Mexico obtained a copy of the message by bribing a trade telegraph official. While the message was being sent back to Mexico from the US, the Embassy of Mexico had been coded with code 13040, which was older than the code, since it did not have the code 0075. During the transmission, the telegraph had a new date. They solved the telegraph coded by 13040 and completed the missing parts of the message. The British could easily deliver this telegram to the US. It was not clear that they listened to the diplomatic line, but they showed that they solved the new code, 0075. Worst of all, the Germans could change the code 13040. Moreover, the US could verify this commercial telegraph with its own commercial telegraph records. The US believed and supported this story.

The Germans made another mistake and never thought the code could be broken. They asked the Mexican ambassador what copies of the telegraph he was doing and went on a treacherous hunt at the embassy. The British continued to resolve the message of the Germans. The same success as II. In the World War, Enigma continued to decode and decode. Turkey's peace talks in the British Lausanne also written to address the telegraph codes.



CLASS OF SERVICE DESIRED
Fast Day Message ✓
Day Letter
Night Message
Night Letter
Patrons should mark an X opposite the class of service desired; OTHERWISE THE TELEGRAM WILL BE TRANSMITTED AS A FAST DAY MESSAGE.

CHARGE 8589

WESTERN UNION TELEGRAM

NEWCOMB CARLTON, PRESIDENT

Receiver's No. 215
Check 3500
Time Filed

Send the following telegram, subject to the terms on back hereof, which are hereby agreed to

via Galveston

JAN 19 1917

GERMAN LEGATION

MEXICO CITY

```
130    13042  13401  8501   115    3528   416    17214  6491   11310
18147  18222  21560  10247  11518  23677  13605  3494   14936
98092  5905   11311  10392  10371  0302   21290  5161   39695
23571  17504  11269  18276  18101  0317   0228   17694  4473
22284  22200  19452  21589  67893  5569   13918  8958   12137
1333   4725   4458   5905   17166  13851  4458   17149  14471  6706
13850  12224  6929   14991  7382   15857  67893  14218  36477
5870   17553  67893  5870   5454   16102  15217  22801  17138
21001  17388  7446   23638  18222  6719   14331  15021  23845
3156   23552  22096  21604  4797   9497   22464  20855  4377
23610  18140  22260  5905   13347  20420  39689  13732  20667
6929   5275   18507  52262  1340   22049  13339  11265  22295
10439  14814  4178   6992   8784   7632   7357   6926   52262  11267
21100  21272  9346   9559   22464  15874  18502  18500  15857
2188   5376   7381   98092  16127  13486  9350   9220   76036  14219
5144   2831   17920  11347  17142  11264  7667   7762   15099  9110
10482  97556  3569   3670
```

BERNSTORFF.

INSPECTED

Charge German Embassy.

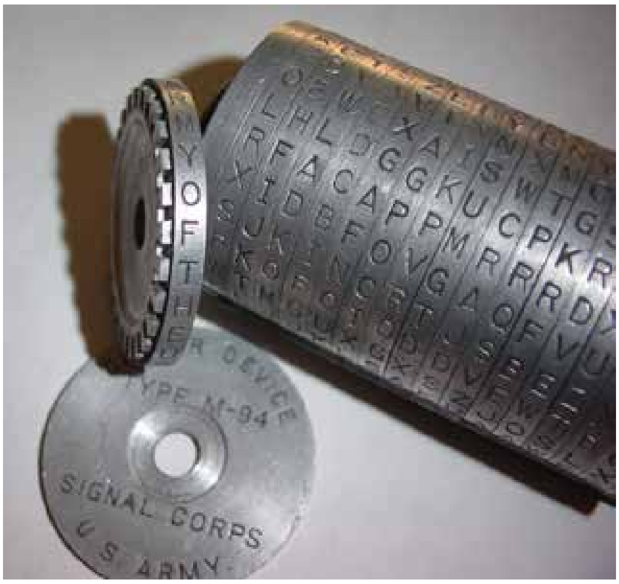
862.20212/82A

Figure 1 Copy of the Zimmerman Telegram sent to the Mexican Embassy

Following the entrance of the US into the war, Zimmermann concentrated on Russia which suffered from internal turmoil and converted to Republic regime on March 15, 1917 after the Tsar, Nicholas II of Russia left the throne. He allowed Lenin (Vladimir Ilyich Ulyanov) and his companions to pass through German territory by train to go from Switzerland to Russia. Lenin, too, was aware of Zimmermann's ideas, and this was the beginning of the October Revolution in Russia, whether this was a butterfly effect or a Room 40 effect.

The Jefferson Disc

As can be understood by its name, the Jefferson Disc was developed in 1795 by Thomas Jefferson, the third president of the United States, just as the Roman Emperor Julius Caesar developed the famous Caesar code. Hidden writing is such an indispensable need that the people who run states themselves have worked on this important issue as a developer. The Jefferson Disc was reinvented by french Étienne Bazeries, one hundred years later, independent of Thomas Jefferson. Bazeries was a talented writing analyst who was able to solve the Grand Chiffre developed by Antoine Rossignol for the 14th Louis. The US military revised the Jefferson Disc with the name M-94 and used it between 1923 and 1942. The breaking of Vigenère Encryption by Friedrich Kasiski in 1863 brought about a new search. The Jefferson Disc was the searched solution.



The Jefferson system consists of 36 numbered discs, all of which are arranged on an axis. Each disc has a hole in the center of the axle. 26 letters were processed around the disc. The order of letters on each disk is different. This makes each disk unique and each disk is given a distinctive number. The order of the disks on the axle is the key to encryption. Both the people encoding and decoding it must know and use the same sorting

Encrypting with Jefferson Disc

For convenience, let us suppose we use 15 disks. As shown in the table, each letter of the alphabet is written around the disk in a different order on each disk. Each disc has a distinctive number between 1 and 15.

Random ranking of Jefferson discs and letters adapted to Turkish

1. JNFĞLKIÜEOCTPABHÖRSUMGŞÇYVİDZ
2. KOCAGBUSNREMÖZĞLPÜŞVDIÇTJFHİY
3. ELGTYSBDNFUZIJÜCRİOMPKHÇÖŞVAĞ
4. YZEÖBOTLIİGÜVCMKAFDNRÇJŞSĞUHP
5. KÇITFCÖYŞPHLĞÜAEGOSUBİZVNMJRD
6. CÇSFHTEYVKZPİDAJRNIGLMOBĞUÖŞÜ
7. ÖRGİPYFÇEKJOBDÜMHIUĞZSCŞAVLTN
8. CŞMSÜAZFBEKDYĞPTGIOİVRNLÖJUHÇ
9. TLMOGHABDUÜÇVRFZİÖYĞCSKENŞIJP
10. KÇŞYGEMCTÖJRVOZLÜĞPIİNUASFBDH
11. IŞEOZUHBCKDÜRGİTVLÖĞFJPMÇSYNA
12. KCOHBÇİMNVRPYJTŞZDĞGALUIÜFSÖE
13. ÜŞĞMGNIVKİDPBHRÇYETLCSZUJOAFÖ
14. YCITÖAVZÜNĞHGEİMJŞBDUORFLKÇPS
15. İCIAGŞRTHZMPJYOÇDSLKFNEBVUÜĞÖ

The password key is designated by the order of the disks 7, 9, 11, 3, 5, 8, 6, 15, 10, 14, 2, 12, 1, 4, 13. The encoder places the discs on the axle in the agreed order.

Jefferson disks sorted by specified encryption scheme

7. ÖRGİPYFÇEKJOBDÜMHIUĞZSCŞAVLTN
9. TLMOGHABDUÜÇVRFZİÖYĞCSKENŞIJP
11. IŞEOZUHBCKDÜRGİTVLÖĞFJPMÇSYNA
3. ELGTYSBDNFUZIJÜCRİOMPKHÇÖŞVAĞ
5. KÇITFCÖYŞPHLĞÜAEGOSUBİZVNMJRD
8. CŞMSÜAZFBEKDYĞPTGIOİVRNLÖJUHÇ
6. CÇSFHTEYVKZPİDAJRNIGLMOBĞUÖŞÜ
15. İCIAGŞRTHZMPJYOÇDSLKFNEBVUÜĞÖ
10. KÇŞYGEMCTÖJRVOZLÜĞPIİNUASFBDH
14. YCITÖAVZÜNĞHGEİMJŞBDUORFLKÇPS
2. KOCAGBUSNREMÖZĞLPÜŞVDIÇTJFHİY
12. KCOHBÇİMNVRPYJTŞZDĞGALUIÜFSÖE
1. JNFĞLKIÜEOCTPABHÖRSUMGŞÇYVİDZ
4. YZEÖBOTLIİGÜVCMKAFDNRÇJŞSĞUHP
13. ÜŞĞMGNIVKİDPBHRÇYETLCSZUJOAFÖ

In order to encrypt his name, Thomas Jefferson turns the first number 7 disc on the axle into a T letter. Then dials the 2nd digit of the number 9 disc of the first number 1 until the letter H of the first disc is next to T. Repeats this operation for each letter to be encoded in other disks until the encryption is finished. When the process is finished, the discs will be aligned. Red painted letters indicate clear text encrypted.

The appearance of the Jefferson discs at the end of the encryption process:

7. ÖRGİPYFÇEK J OBDÜMHIUĞZSCŞAVL T
9. BDUÜÇVRFZİ Ö YĞCSKENŞIJPTLMOG H
11. UHBCKDÜRGİ T VLÖĞFJPMÇSYNAIŞE O
3. KHÇÖŞVAĞEL G TYSBDNFUZIJÜCRİO M
5. GOSUBİZVNM J RDKÇITFCÖYŞPHLĞÜ A
8. AZFBEKDYĞP T GIOİVRNLÖJUHÇCŞM S
6. NIGLMOBĞUÖ Ş ÜCÇSFHTEYVKZPİDA J
15. VUÜĞÖİCIAG Ş RTHZMPJYOÇDSLKFN E
10. DHKÇŞYGEMC T ÖJRVOZLÜĞPIİNUAS F
14. KÇPSYCITÖA V ZÜNĞHGEİMJŞBDUOR F
2. ÖZĞLPÜŞVDI Ç TJFHİYKOCAGBUSNR E
12. YJTŞZDĞGAL U IÜFSÖEKCOHBÇİMNV R
1. MGŞÇYVİDZJ N FĞLKIÜEOCTPABHÖR S
4. LIİGÜVCMKA F DNRÇJŞSĞUHPYZEÖB O
13. VKİDPBHRÇY E TLCSZUJOAFÖÜŞĞMG N

After this step, a predetermined offset value is counted upwards or backwards from the open text line. The reached line is transmitted to the other party as encrypted text. In this example, we determine the offset value of 17 backwards and generate the ***JÖTGJTŞŞTVÇUNFE*** code from the encrypted blue line and inform the other party.

Deciphering Encrypted Text with Jefferson Disk

When the ***JÖTGJTŞŞTVÇUNFE*** encrypted text reaches the operator, he/she places 7, 9, 11, 3, 5, 8, 6, 15, 10, 14, 2, 12, 1, 4, 13 numbered discs on the axle, again as it had been predetermined. As in the same encryption process, starting from the first letter of the encrypted text, the operator turns the disks until they get the ***JÖTGJTŞŞTVÇUNFE*** sequence. The specified offset value can reach up to 17 reverse sides, that is, the forward text of Thomas Jefferson. In fact, it may not be necessary to specify an offset value. This is because if there is an arbitrary row, except for the explicit text line, and the other side is declared, it is unlikely that there will be a meaningful line on the disks except the encrypted text. It will not be difficult for the decoder to notice the meaningful message when he/she examines the lines.

Hill Cipher

It was developed by Lester S. Hill in 1929 and is a linear algebra-based block encryption method. Unfortunately, Hill encryption is not widely available. Because the method required solving mathematical equations, it was difficult to use manually. Hill and his partner also built the mechanical encryption machine. But they failed to sell it. Hill encryption was an important development since it showed that encryption could be performed using math. Cryptography has become an important contribution to science.

The method used modular arithmetic and matrix. Each of the 26 letters in the English alphabet (A = 0, B = 1 B Z = 25) was paired with a number. Each letter was represented by the number calculated with Mod 26. Encryption was done by multiplying n-length (n vector) letter blocks with n x n dimensional square matrix. The **n x n matrix** was also considered as the encryption key. The decoding process had to be the inverse of the matrix. It was possible to adapt only to the alphabet of each language by changing the mode value.

In our example we will use mode 29 for the Turkish alphabet. Calculating the inverse of matrices is a complex task as their size increases. Moreover, it is more difficult to do this according to modular arithmetic. We will give our example through the 2-dimensional square matrix, which is easier. This is a selection that weakens the encryption key.

Let us briefly talk about the math tools we will use. More detailed

Modular Arithmetic

$\frac{A}{B} = Q\ ,R$; A is the dividend, B is the divisor, Q is the quotient and R is remainder. We are only interested in what the remainder is when we divide A by B. Therefore, we would have A mod B = R.

Modular Arithmetic Inversion

The modular inverse of A (mod C) is A^-1. (A*A^-1) 1 ≡ (mod C) or equivalent (A*A^-1) mod C=1. Only the numbers coprimeing C (numbers that share no prime factors with C) have a modular inverse (mod C).

If ax = 1 (mode m), x is the inverse of a. In practice, we are looking for the number of x, which provides mx + 1 (mod m) = 1. According to Mod29, the numbers 30, 59, 88, 117 … ∞ are numbers equal to 1 (mode 29). For example, for a = 8, 88 will be Mode 29 = 11.

Square Matrix

In linear algebra, the square matrix is a matrix whose number of rows and columns are equal. An **n x n** matrix is known as a square matrix of size n. Any two square matrices of same size can be used to perform matrix addition and multiplication.

2 x 2 Square Matrix Multiplication

Matrix multiplication is a binary process made in a pair of matrices and produces another matrix.

A = $[a\ b\ p\ q]$, B = $[x\ y]$ to AB = $[a\ b\ p\ q]\ [x\ y] = (\mathrm{a}x + by\ \ px + qy)$

The Inverse of 2 x 2 Square Matrix

If A = $[a\ b\ p\ q]$, $\mathrm{A}^{-1} = (\mathrm{a}q - bp)^{-1} * [q - b\ -pa]$

There is no inverse of a matrix such that det A= $(\mathrm{a}q - bp) = 0$

Encryption with Hill System

The number of 2x2 dimensional square matrix elements as HBCÇ (B = 1, H = 9, C = 2, Ç = 3) were determined as the key group. In order to express the key-letter and matrix relationship easily, we can visualize A = [H B C C]. If we replace the letters with Mod 29 number, we obtain the key matrix A = [9 1 2 3]. We use this matrix in the encryption process and its inverse in the decryption process, therefore we need to find the inverse of this matrix. We can't use this key if the matrix is not the inverse.

Let's calculate the inverse matrix

$A^{-1} = (9 * 3 - 1 * 2)^{-1} * [\ 3\,2827\ 9]$, determinant of matrix detA = $(9 * 3 - 1 * 2) = 25$; there is the inverse of the matrix because the determinant is not zero.

According to modular arithmetic, we apply inverse to determinant process.

For $= (9 * 3 - 1 * 2)^{-1} = 25^{-1} =$ ¿ $> 25 * x = 1 (\text{mod } 29)$ => **x=7**, $25 * 7 = 1 (\text{mod } 29)$ => $175 = 1$ (mod 29)

Continuing:

$A^{-1} = 7 * [3\,2627\ 9] = [\ 7 * 217 * 227 * 15\ 7 * 5] =$ ¿ [21 22 15 5] Since the inverse of the matrix exists, the key be can used. We can continue the encryption process.

The key we set, HBCC is written as one (H B C Ç) → (9123)

If the text to be encrypted is ATEŞ, (A T), (EŞ) → (0 23), (5 22) is written as two pieces (block size becomes 2 because we use 2x2 square matrix).

Let's do the multiplication with key matrix.

(9123) (023) ≡ (23 11) (mod 29) , (9123) (522) ≡ (918) (mod 29)

(23 11), (918) → (*T İ*), (*H Ö*)

Our encrypted text is TİHÖ.

Decoding with the Hill System

Field code decoder converts TİHÖ encoded textz letters to matrices:

TİHÖ → (*Tİ*), (*HÖ*) → (23 11), (918)

It multiplies these matrices with the previously solved inverse matrix of the key.

(2122155) (23 11) ≡ (023) (mod 29) and (2122155) (918) ≡ (522) (mod 29)

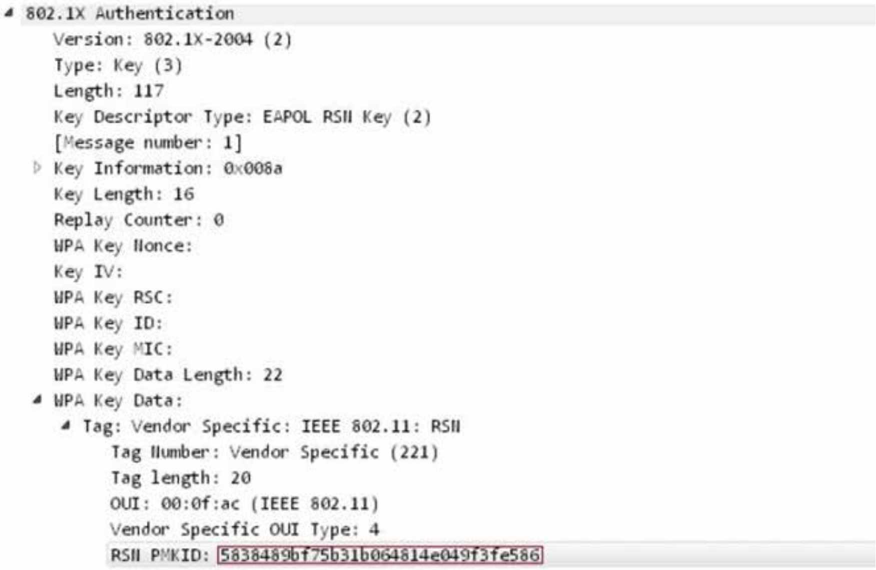
(023), (522) → (*AT*), (*EŞ*) → ATEŞ

The 10 Biggest Hacks of 2018

As we leave another year behind, we can step back and take a look at the major hacks that took place in 2018. Listed below are the top ten major hacks of 2018 in no particular order. With the beginning of the new year, we're hoping to see fewer of the hacks which affected thousands of users.

Attack On WPA/WPA2 - *November 25, 2018*

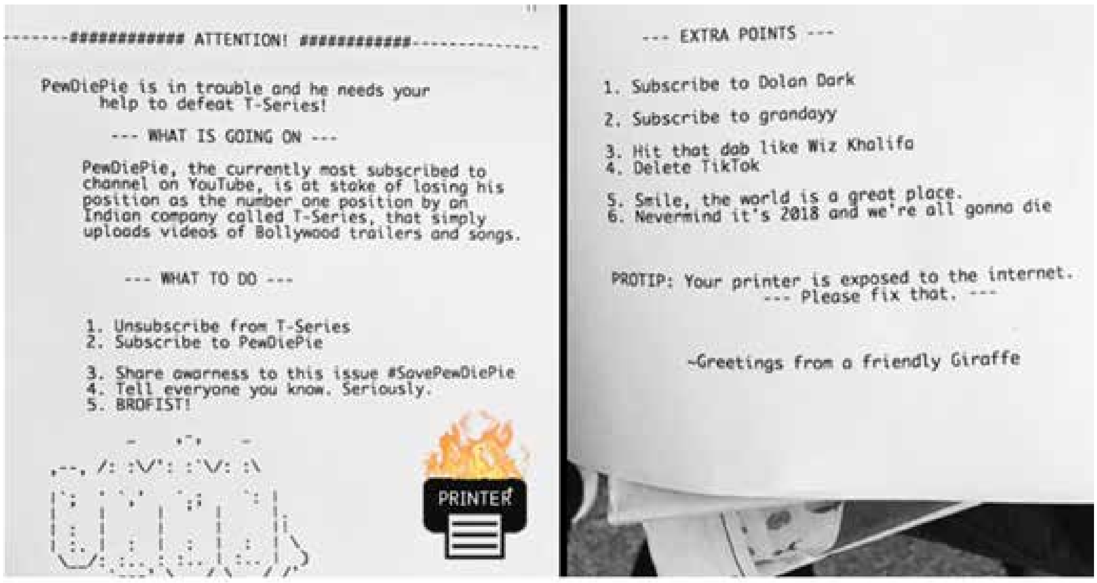
The commonly used WPA/WPA2 encryption methods on WiFi networks were targeted in this attack. It has been long known that these encryption methods are the safest. However, with the implementation of this hack, it's clear that nothing is safe in the world of technology. The internet communication of users can be hacked through their Wi-Fi networks and the activity of users can be spied on. The attackers recover the Pre-shared Key login passwords to obtain user data.



```
◢ 802.1X Authentication
    Version: 802.1X-2004 (2)
    Type: Key (3)
    Length: 117
    Key Descriptor Type: EAPOL RSN Key (2)
    [Message number: 1]
  ▷ Key Information: 0x008a
    Key Length: 16
    Replay Counter: 0
    WPA Key Nonce:
    Key IV:
    WPA Key RSC:
    WPA Key ID:
    WPA Key MIC:
    WPA Key Data Length: 22
  ◢ WPA Key Data:
    ◢ Tag: Vendor Specific: IEEE 802.11: RSN
        Tag Number: Vendor Specific (221)
        Tag length: 20
        OUI: 00:0f:ac (IEEE 802.11)
        Vendor Specific OUI Type: 4
        RSN PMKID: 5838489bf75b31b064814e049f3fe586
```

50,000 Printers Hacked to Promote PewDiePie YouTube Channel - *November 30, 2018*

The most-subscribed Youtube channel challenge between Youtubers is perhaps the most active internet battle of our era. A hacker hijacked more than 50,000 printers worldwide to print out a flyer asking people to subscribe to PewDiePie's Youtube channel. The flyer also asked to unsubscribe from T-Series' channel. Will the wireless printing feature be the beginning of a new form of advertising?



```
-------############# ATTENTION! ############--------------

PewDiePie is in trouble and he needs your
     help to defeat T-Series!

     --- WHAT IS GOING ON ---

   PewDiePie, the currently most subscribed to
   channel on YouTube, is at stake of losing his
   position as the number one position by an
   Indian company called T-Series, that simply
   uploads videos of Bollywood trailers and songs.

       --- WHAT TO DO ---

   1. Unsubscribe from T-Series
   2. Subscribe to PewDiePie

   3. Share awarness to this issue #SavePewDiePie
   4. Tell everyone you know. Seriously.
   5. BROFIST!
```

```
      --- EXTRA POINTS ---

  1. Subscribe to Dolon Dark

  2. Subscribe to grandayy

  3. Hit that dab like Wiz Khalifa
  4. Delete TikTok

  5. Smile, the world is a great place.
  6. Nevermind it's 2018 and we're all gonna die


  PROTIP: Your printer is exposed to the internet.
                --- Please fix that. ---


         ~Greetings from a friendly Giraffe
```

New Facebook Bug Exposed 6.8 Million Users Photos to Third-Party Apps - *December 14, 2018*

2018 was definitely not Facebook's year. In mid-December, just as the year was coming to an end, a new programming bug was discovered on Facebook. Nearly 1500 third-party apps acquired access to the photos of nearly 7 million users. Crossing fingers for a better year for Facebook and its users.

LibSSH Flaw Allows Hackers to Take Over Servers Without Password - *October 16, 2018*

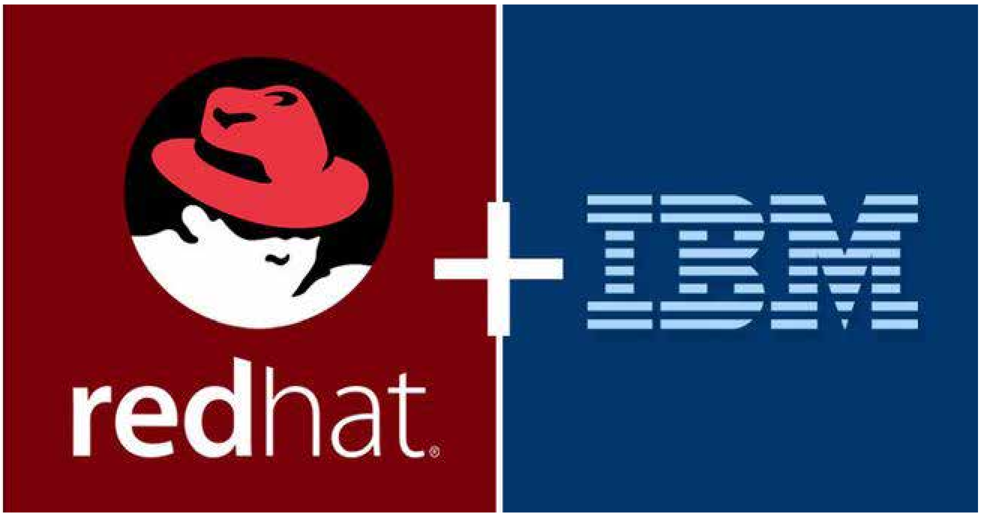
A severe vulnerability has been hiding for the past four years, only to be discovered in 2018. Libssh, SSH implementation library was found vulnerable to a hack that gave access and administrator privileges to servers without the necessity of authentication.

Someone Hijacked MEGA Chrome Extension to Steal Users' Passwords - *September 04, 2018*

In September, the importance of a course of 24 hours has proved it's vitality in the hacker's world. The Chrome extension for the MEGA.nz cloud storage service had been compromised and replaced with a malicious version. Due to the auto-update feature of extensions, in a matter of 24 hours millions of users' credentials and credit card information for popular websites like Amazon, Microsoft, Github, and Google, as well as private keys for users' cryptocurrency wallet were stolen.

IBM Buys "Red Hat" Open-Source Software Company for $34 Billion - *October 28, 2018*

Maybe not the hack news you expect, but a huge legal hack took place in 2018. IBM confirmed that it would be acquiring the open source Linux firm **Red Hat** for $34 billion. Once again we saw the importance and growth of an open source firm.



Unprivileged Linux Users With UID > INT_MAX Can Execute Any Command - *December 06, 2018*

Ever wondered what importance your operating system's UID had? Well, according to a new vulnerability, any low-privileged user with a UID greater than 2147483647 can execute all systemctl commands. They don't even need any authorization. Yeah, go check your UID!

Unpatched VirtualBox Zero-Day Vulnerability and Exploit Released Online - *November 07, 2018*

Not all virtual machines are bulletproof. A zero-day vulnerability in VirtualBox has been disclosed by a security researcher. The vulnerability occurs due to memory corruption issues and affects Intel PRO / 1000 MT Desktop (82540EM) network card (E1000) when the network mode is set to NAT (Network Address Translation). A malicious program can escape the virtual machine and affect the host machine. Don't trust the VMs!

Just Answering A Video Call Could Compromise Your WhatsApp Account - *October 09, 2018*

Google Project Zero security researcher found a vulnerability in WhatsApp messenger that could allow hackers to remotely take full control of your WhatsApp just by video calling you over the messaging app.

Intel CPU Vulnerability Exploits Hyper-Threading to Steal Encrypted Data - *November 03, 2018*

Just because you encrypt data doesn't mean it's safe and sound. A new vulnerability on Intel CPUs allows attackers to steal sensitive protected data like passwords and cryptographic keys from other processes running in the same CPU core with multi-threading enabled.

Back to the Old Tech: OFFLINE COMMUNICATION NETWORKS AND SECURITY

The place of internet in today's everyday lives is so important that it does not need to be explained. Most of our lives dependently continue on the existence of internet and the consequences of this will seem to increase in the future. Under tranquil and peaceful circumstances we do not actually notice the vulnerabilities of the internet technology. Although it seems as an uncontrollable and unbreakable structure, the internet actually has a pretty fragile structure. We have already noticed it's inspectability with local censorships and with the global studies done by NSA. However, we did not actually figure out how fragile it's structure is. The reason of this may be that since the widespread of internet we did not face a great war or disaster. In fact, there does not always need to be a great war or disaster. For instance a government can stop the access to internet if they wish, for how long they want, in the whole country.

In this article, I am going to talk about the weak points of the internet technology and what we will experience in a possible disaster environment. As a solution to these, I am going to explain the concept of my project which I have been running some tests on and plan to finish in 2019, additionally mentioning the security side of this concept.

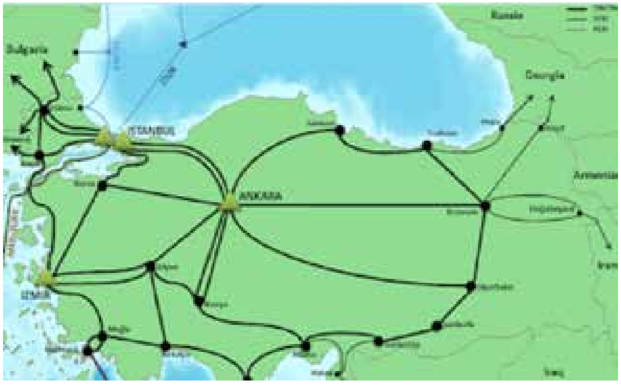
Physical Structure of Internet

There is something basic taught in network: the OSI model. There are 7 layers in this model:

1) Physical Layer
2) Data link Layer
3) Network Layer
4) Transport Layer
5) Session Layer
6) Presentation Layer
7) Application Layer

Security researchers can expertise on these layers specifically. But one of these layers, the "physical layer", is usually neglected. For example when telling the story of how a user connects to a website, we start by telling that the user types in the site, sending a query to DNS. However, there is a crucial point skipped here: how are the packets a user from Turkey sends to a server in America transported in physical world? The answer is simple: via cables! (can be transported via satellites but that is excluded for now)

For instance the image below shows the internet exit cables of Turkey:



http://www.burakavci.com.tr/2015/12/turkiye-backbone-kafos.html

So, what is the fragility of this cable technology? Of course, it being open to physical damages. In case of a great war or disaster, for example, if the exterior and interior cables of Turkey are damaged, a big majority of the Turkish people will be left without internet.

Physical damages are actually not the only problem. Each line going abroad has a capacity. In 2007, Estonia was exposed to cyber attacks by Russia Estonia, almost filling up it's external line capacity, thus rendering Estonia unable to provide internet connection to the outer world.

Server-Client Structure

We all know the server-client structure. Let's roughly cover it again. For example, *arkakapidergi.com* broadcasts over a server. Users who want to read the contents connect here, yet if someone sets the server *arkakapidergi.com* is on on fire, these contents would become inaccessible. Thanks to today's technologies like cloud, this would not be as easy. However, at the end of the day, most things actually work in a centralized structure. Twitter or Whatsapp's server damaged in a battle would cause their users to be deprived from them.

The P2P (peer-to-peer) technology developed as an alternative to server-client structure offers a solution to such problems. Anyway, since it also works internet-based, someone cutting the cable with scissors throws this complex technology away.

Theoretical Concept of Offline Communication Network

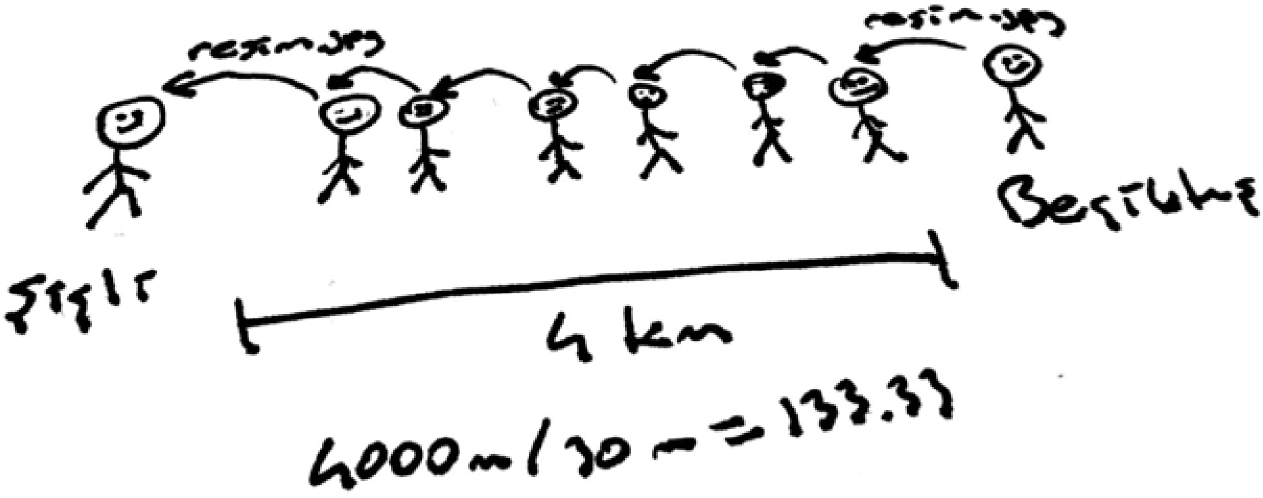
Internet actually is not required for two phones to communicate. We can share messages and files over the older Wifi and Bluetooth protocols, but the other person should be close to us. We can't send a file from Beşiktaş to Şili over Wifi. So, what would happen if there were intermediaries that could carry our file over Wifi Beşiktaş to Şili? You can think of this as a Chinese whispers game. The file you send will reach the target at Şişli transported over Wifi by intermediaries. It can be shown with a sketch like this (all drawings are from my project draft notebook) :

The distance between Beşiktaş and Şişli is 4 km. Although the range of telephone Wifi differ from device to device, this corresponds to approximately 30 meters according to my online researches. For this reason, in order to transfer a file from Beşiktaş to Şişli, 133 people standing at 30 meters intervals are required. Considering the crowdedness of Istanbul, the number 133 is pretty reasonable. After understanding that this is practically possible, let us take a look at the details. The sketch above is actually not realistic nor detailed. For instance, what if it is midnight and there are not 133 people between Beşiktaş and Şişli? Or more importantly if there is, how these people will know who the message should be transferred to? I have two ideas at this point:

1) Full Mesh Network

For example, I would like to send a message to a friend who is living in Şişli. What I need to do is to broadcast the message and the recipient name to people near me. People within 30 meters diameter will see this broadcast message but since they are not the receiver, they will keep the message in their own devices. Later on, to transport the message to the receiver, when these people are traveling throughout the day, they will continue broadcasting this message sent from the user Utku. Other users will again keep this message in their devices since they too are not the receiver. This message will go around a lot of people's device until it reaches the real receiver.

Advantages: Messages can be transported between users without having the need for an extra structure

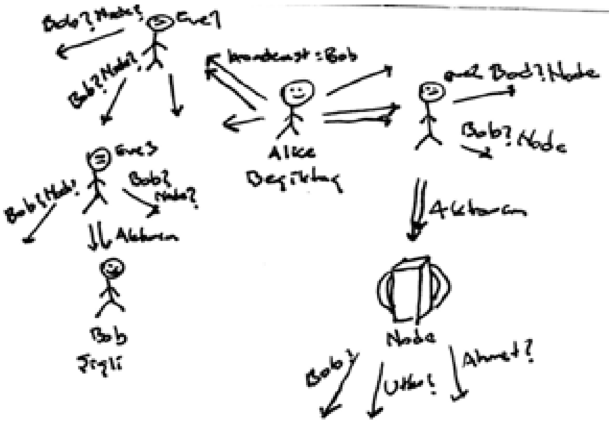


Disadvantages: There are too much. Stored and broadcast messages will take up too much space in the devices as the number of users increase. Besides, since the feedback mechanism will be problematic, broadcasts will continue constantly. For example, the message somehow reached the user at Şişli, but another user in Avcılar may continue broadcasting the message without knowing it did.

2) Semi Mesh Network

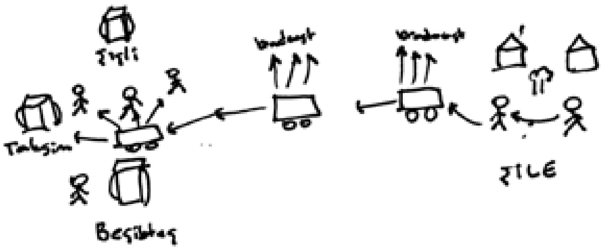
This concept has a structure similar to blockchain. Here, broadcast is done not only by the users, but also by nodes located at several points in Istanbul. Following the same example: I want to send a message to a friend in Şişli. The first thing I will do is again to broadcast the message and the recipient. Users within 30 meters will take this message and since they are not the receiver, they will keep it in their devices. In order to save the users from entering the loop of endless broadcasting, there is a factor, the nodes. The user will continue to broadcast the message until finding the receiver or a node. Besides, if a node is encountered, the message will be transmitted and broadcast will be ended. Let us explain with this sketch:

Alice in Beşiktaş, broadcasts the message she wants to send to Bob who is in Şişli. The broadcast message reaches to users named Eve1 and Eve2, which broadcasts this message again. User Eve2 transfers the message to a node she is close to and ends the broadcasting process. And the node is constantly broadcasting the messages it received for users Bob, Utku and Ahmet. On the other hand, Eve3 gets the broadcast message of Eve1, and since her broadcast found it's recipient Bob, she ends her own process. Therefore, Bob has received his message via Eve3. Ok, so what if Eve3 was not in between? Bob could grab the message only by going near the node. You can think of these nodes as a post office. The users can sometimes go and check if there is a new message for them. Do not think of these nodes as big base stations. They can be small computers some volunteers dedicate for this job. As an example, a café owner can turn an unused computer into a node. So, users visiting this café can receive if there is a message there waiting for them.



Process-Speeding Postmen

Alright, so how would a person living in a village in Şile send a message to Beşiktaş over this system? The message will never reach the recipient since the number of people in between is limited. This is where the volunteer postman carriages get involved. For instance, a carriage will collect all broadcast messages from a village and set off to Istanbul and will transfer the messages to a lot of nodes by travelling all over Istanbul. Therefore the target users of the villagers will be able to read their messages when they come to these nodes.



Security Side

Message Confidentiality and Security

When explaining the concept, I did not mention the security side. In this concept, since everyone can carry anyone's message, confidentiality is actually very much violated. It is of course not logical this way. However, it is possible to ensure message security and confidentiality. Here, we will focus on 3 points:

1) When a message to Utku is broadcast, user Ziyahan should not be able to end the transmission by saying "I am Utku" (Authentication)

2) The message should not be read by those other than the recipient (End-to-end encryption)

3) The contents of the message should not be changed during transmission (Integrity)

Before starting to talk about the details, I need to mention that cryptography is a science branch which requires special education and that I have no high education upon it, I study it as a hobby. So, the methods I will describe below may not be the best ones.

In order to overcome the first and second problems, we can use asymmetric and symmetric encryption algorithms together as a hybrid. Here, a handshake model similar to that used when connecting to HTTPS websites can be run.

First of all, the users should download the application and login while connected to the internet. When they login, the application server gives them a public-private key pair. Apart from this, it also sends digital signature data encrypted by the server's private key, proving that these keys are personal. At the same time it sends the server's public key to the user's devices so they can authenticate this digital signature.

Let us say that the internet connection is lost and user Bob wants to send a message to user Alice. The handshake will look like this:

Bob: I have a message to Alice, are you Alice?

Alice: Yes, I am Alice.

Alice: Here, take my public key encrypted by the server's private key (digital signature)

Bob decrypts Alice's digital signature with the server public key found in his device. Result: "the person I encounter is really Alice and I have Alice's public key"

Bob: Here, take the message I want to send to you

Handshake ends and Alice receives the message. But there is a puzzling detail here.

If it is Bob who directly sent the message to Alice, there is no problem. He already gained Alice's public key during the handshake, he can encrypt and send with it**.** With this he can encrypt and send the message (Actually he will encrypt the symmetric key using hybrid encryption but I am going to pass that as it will branch out). If Bob is the person carrying this message, then how the first person to send this message will have Alice's public key and encrypt the message with it?

This situation is one of the weakest points of offline communication network. I have a solution for this in my mind: when registering to the application, users should download everyone's public key to their devices. That is to say, if there are 1000 people registered to the system, there will be 1000 public key information. Thereby, for example when the user Utku wants to send a message to user Alice, he will encrypt the message with Alice's public key and then broadcast this message. Bob will take this message to transfer it to Alice but he will not be able to see the contents since he doesn't have Alice's private key. Apart from this, since he does not have Alice's digital signature, Bob will not be able to introduce himself to user Utku as Alice. Later, Bob will make a handshake with Alice and transfer her the message and Alice will decrypt and read the message with her own private key.

If you noticed,the key system is dependent on the users downloading the application on internet existence. So when the internet is gone, will someone not be able to obtain and use this application? This person will never be able to prove oneself since he/she could not grab a digital signature from the application server. This means that he/she can not make use of sending private messages yet there is no reason not to be able to broadcast public messages.

Public channels will be designed at the nodes for this type of users. For instance, if a new user broadcasts "I'm hungry" publicly, if there is a node close to him/her, this message will be broadcast through that node's public channel. If there is not a nearby node, the surrounding users will take this message and transfer it when they get close to a node.

Other Security Problems

Security problems are not restricted only to message contents. Another significant security problem will be DoS (Denial of Service) attacks. Here, a user can fill the capacity of people around by broadcasting hundreds of messages a second. But with control mechanisms put into the application, this can be overcome up to a point.

Apart from this, jamming attacks will also cause trouble. Long story short, we can say that this project contains lots of problems that need to be pondered on before being realized.

Similar Works

There are projects available which make it possible for users to intercommunicate and send files offline, e.g. Firechat, hypelabs.io. When I first saw these projects I became sad that it was already done. However, when examining further I realized that these projects do not support a distributed structure like I described above. That is to say that you can send and receive messages with people around you, but you cannot send a message to a user far away over other users.

Apart from this, I came across studies tried in some universities in America. But never encountered a distributed structure that can be realized at a city-size.

WAF Bypassing Methods Part 2: Tricks and Indirect Access

This article is a follow-up to my article titled "WAF Bypassing Methods" in our first issue. In my previous writing, I've described the ways to learn the IP address for bypassing cloud-based WAFs. In this article, as well as discussing the methods that can be used to bypass WAFs, I will talk about several ways to normalize outgoing traffic over the WAF and make sure that this traffic is not blocked. I would like to briefly summarize the definition of WAFs because this text is a continuation.

What is WAF?

A Web Application Firewall (WAF) aims to secure the web applications. It essentially monitors the traffic between the user and the application. It can generate reports, and manipulate the traffic or the application if needed. It tries to prevent potential attacks that come up after such manipulations.

WAFs can be designed in many ways. However, there are two sorts of WAFs regarding the protection model. One of these WAFs is based on rules. These WAFs work based on the rules that are assigned to themselves. Thanks to these rules, the requests in a blacklist are blocked while the requests in a whitelist are allowed. On the other hand, signature-based WAFs block the requests that fit the formats (signatures) while allowing the rest. WAFs generally use these two approaches together.

How to Bypass WAFs?

WAFs usually work based on signatures, and they don't know the situation of the background application. Thus, a command may not be blocked by the WAF when the command is executed by the application running in the background. Some WAFs have strict filters (1). Therefore, different types of character codes can overpass the WAFs. Likewise, the comment lines and text blocks that are allowed by WAFs can be misused because signature-based WAFs don't know the backend applications. I will list the bypass methods in two categories: rule-based and Signature-Based WAFs.

After recurring attacks, devices running in the background like SIEM and UTM can detect your attacks, and let you be faced with a WAF that shall not be passed and is limited. To prevent such situation you can change the attack frequency or type. This rule set sometimes might be more limited is not on the old set and might have been defined independently than the old rules. This new rule set might allow a method that was not allowed by the old rule set. So, it is better to try both sets in such cases. WAFs are used to secure the web applications as is evident from its name. If the application may returns to you in a different way, then you might not need to go to the WAF. If, for example, you can send the responses of the requests to a different point rather than showing them through the web interface (outgoing routing), you won't need to go to the WAF for the responses and will be able to bypass the WAF.

Rule-based WAFs

Rule-based WAFs put some characters in order to replace with another character set or to delete thanks to some specific rules. In this way, they will clean the potential attacks in the requests made to the application.

Character Encoding

Some basic rule-based WAFs may not decode a character encoding method that the background application can understand.

This method can be used if a kind of character encoding is supported by the application and is not blocked by the WAF. Sometimes, some specific characters in a character encoding might be prevented. In such cases, it is possible to combine multiple encoding techniques. These are some character encodings: Base64, Hex, URL Encoding, HTML Entity Encoding, CSS Hex Encoding, Unicode, UTF-8 and other variations.

(%00, &#xHH;,\x22,&quot,\XXXXXX..)

HTTP Headers

Some WAFs allow a request without checking if it contains some certain headers and if these headers have one of the trusted IP addresses. Some of them are listed below:

X-Originating-IP: 127.0.0.1

X-Forwarded-For: 127.0.0.1

X-Remote-IP: 127.0.0.1

X-Remote-Addr: 127.0.0.1

Some WAFs do filtering based on the Content-Type and Host headers. Such WAFs filter some contents using the Content-Type header or check if the Host header has a correct domain. It is possible to bypass WAFs by changing or deleting these headers in a request.

Double Encoding

Double encoding can be used against WAFs that try to delete words using basic filters. Some WAFs will change the unpermitted words just for once. Therefore, you can make the first word get deleted and bypass the simple filters by writing a word in the middle of the second one. Against such attacks, WAFs that delete the word recursively have been developed. (SELselectECT)

Size Changing

Some rule-based WAFs cannot detect the words while checking them once the type size has changed. It is possible to bypass such WAFs by trying upper/lower case combinations. (SeLEcT)

Changing Path

Some WAFs check only the path the request was made to. It is possible to change the path parameter or add a random path variable against such WAFs. If the domain/path combination that a request was made to do not match the rules on a WAF, the WAF might not do filtering. Meanwhile, changing the characters at the end of a path may help us to bypass WAFs. Even though there are different characters at the end of the path, there are many ways which are accepted as a request to the same source from the server. WAFs may not process these characters. So, the WAFs that do filtering to the right path can be evaded.

(/path/vulnerable.php/randomvalue?rest

/path/vulnerable.php;randomvariable=randomvalue?rest

/path/vulnerable.php/rest/, /path/vulnerable.php/rest\)

Hiding Parameters

Some parameters that were added to the requests might be not used and removed by web applications. The requests that involve these parameters will be accepted and run by the web application, with the WAF not filtering them. (PHP deletes the + character at the beginning of the request, ASP deletes the % character at the beginning of the request.)

Adding Comments

This method should be helpful if signature-based WAFs cannot interpret the comment lines correctly. Signature-based WAFs have difficulties with understanding the nested comment lines just like rule-based WAFs have problems about double encoding. This requires them to simulate the application, and this is very difficult as they have many applications that must be protected. Hence, the attack text can be hidden within the comments.

(?parameter=1+un/**/ion+sel/**/ect+4,7,9--)

Signature-Based WAFs

rule-based WAFs could be conquered because they have simple rules. However, signature-based WAFs checks the text if it has a known signature (attack types). In this case, we benefit from the advantage of this vulnerability, which is that a WAF is not aware of the application running, not to be caught by the signature. So, the request must be detected and be converted to a malicious code by the application but not be identified by the WAF. WAFs sometimes can check the responses. At this stage, it is possible to change the response using another way. This changed request must be normal to the WAF. However, it must be understandable by you at the same time.

Function Usage

This method aims to hide the request into the function that is used by the web server or application. The server/application will process the functions once we send the attack, that is prevented by a WAF, hiding it into these functions. As a result of the processed functions, the text that will be used in the attack will be re-generated by the server/application. These functions must run before the parameters we send are not processed. Otherwise, they would not be effective. WAFs might not filter alternatives of these functions. We can bypass WAFs by trying these alternatives.

(substring() -> mid(), substr(), benchmark() -> sleep(), ascii() -> hex(), bin()**)

Logical "and/or" Usage

The logical operators like AND/OR that are interpreted by the server can be used during some attacks such as SQL injection. Because of these operators, the request will be combined and run on the server side. The request running on the server side evade the WAFs because of the usage of the text equivalents of "&" or "|" characters instead of themselves. Changing the request by using "more than, less than, and not equal" signs can be used against the WAFs that check the "equal" sign.

(/?value=2+OR+0x42=0x42, /?value=1+and+ascii(lower(mid((select+parola+from+users+limit+1,1),1,1)))=74)

Exploitable WAF Functions

Some WAFs attempt to provide extra security by adding different functions. We can detect those functions and use them against WAFs. For example, the query below will create a malicious request for a WAF that replaces "*" character with space.

http://www.site.com/index.php?page=-15+uni*on+sel*ect+1,2,3,4

HTTP Parameter Pollution

This attack is related to how the web server interprets the sequential HTTP parameters. Some web servers would combine the two variables if they were defined twice. In that way, we can prepare the attack by defining the variable more than one.

(/?id=1;select+1&id=5,7+from+users+where+id=5--)

HTTP Parameter Distribution

This attack arises from WAFs misinterpreting different parameters. When a comment line and one more parameter is added between two parameters, a WAF will consider the parameters separately and do filtering accordingly. Thus, the web server doesn't see the comment line and run the request. Then, the attack is blocked by the WAF.

(/?a=1+union/*&b=*/select+1,pass/*&c=*/from+users)

Using Wildcards

Some wildcards might be interpreted differently by the web server or the application running in the background side. These characters can be used to establish an attack. For instance, (*) character completes a text, and (?) character completes the missing characters.

(ls -> /???/?s)

Decomposition

Sometimes, when you try an SQL Injection, WAFs begin to catch your attacks and begin to be a problem for you. If the SQL parameter is so long, it can be blocked by the WAF. This method applies to an SQL injection, and it uses the character that is used to split the columns in the queries. After usage of this character, some WAFs think that the query ended, corrupted or is incorrect. Thanks to this character, it is possible to write longer queries, and a WAF doesn't block them even if the application accepts them.

Concatenation

Some applications use different characters to mark the text as "String" and to combine them while some of the applications use the + character to make a text combination ("." Perl and PHP; ".." Lua). If these characters were not combined, then they can be combined with the texts, and WAF can be bypassed. Another way to combine the text is the literal concatenation. This method is used to combine the alternate texts. If a text comes one after another, some programming languages and applications may assume that those texts are the same. So, WAF can be overcome this way.

(echo 't'e's't)

Buffer Overflow

Some of WAFs are very sensitive to the long texts and special characters. These WAFs can crash once they encounter such a situation. For slowing a WAF down or crashing it, long and complex texts and parameters that contain special characters can be tried. After that, it is liable to access the server without any WAF limitation.

(?page=-15+and+(select 1)=(Select 0xCC[..(1000 adet "C")..])+/*!uNIOn*/+/*SeLECt*/+1,2,3,4…..)

(?page=null%0A/**//*!50000%55nIOn*//*yoyu*/ll/**/%0A/*!%53eLEct*/%0A/*nnaa*/+1,2,3,4….)

Coding Language

Some parameters can face with the simple rules in especially XSS attacks. When we send these parameters in a Javascript context, we would be able to evade the WAFs. The difference between this method and using functions is to write the attack within the language syntax instead of using the new text that was generated after the function. So, we focus on assimilating the attack on another language rather than to code the text with functions.

({ background-url: "javascript:alert(1)"; } , { text-size: "expression(alert('XSS'))"; }..)

And the rest?

The methods defined in this article have arose since the WAF filters are defined so strictly, or that they cannot simulate the applications running in the background. Also, WAFs can be bypassed by generating commands that the background application will accept, and the WAFs cannot detect. If you would want to extend the methods, you can process the data (JSON, XML, YAML, etc.) and receive feedback differently (Blind SQLi via DNS queries). You have to know the WAF's specifications to bypass or secure it. Which characters can it block? Which characters does it change and what kind of request does it wait for? Such information will be helpful for you once you encounter a WAF.

In the next article, I will show you how to bypass WAFs' limitations for automated bots. So, you will be able to evade WAFs when scanning a website or downloading it automatically.

(1) They are not applicable/flex. They are the mechanisms that can do parameter/keyword-based filtering. For example, that the first % character is filtered and the %% characters are not blocked while they are side by side.

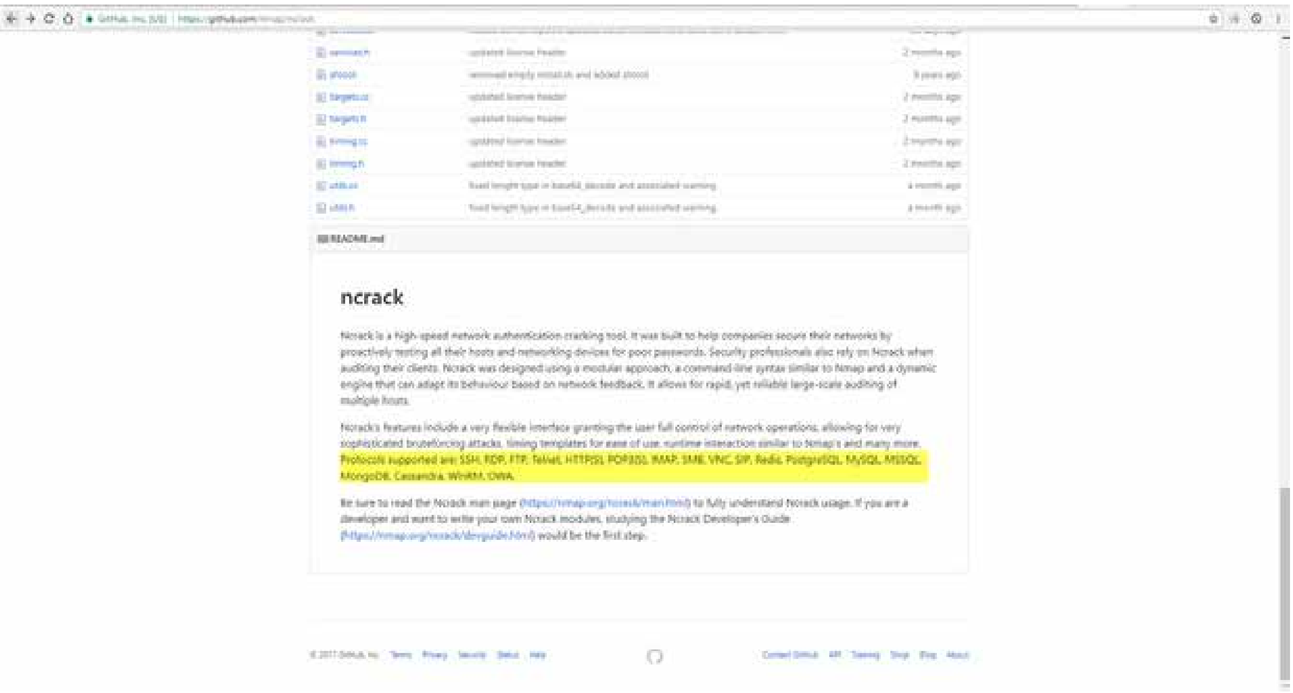
MINI THREAT INTELLIGENCE

The importance of identification and prevention of the cyber attacks against institutions is taken into a high consideration by cyber security centers in today's world. However, a low interaction honeypot system that is easily detectable becomes nothing more than a useless system that just consumes resources in these institutions. But a honeypot system that's easy to install and manage can be invaluable if it's integrated to the SIEM product and can be used as a mini cyber threat intelligence service for the institution.

This homemade mini threat intelligence service could easily discover how the institutions are hacked over simple passwords of each service and protocol. It can also find the countries in which the attackers reside. Just like in my previous studies, I had to make sure that the system wouldn't be easily discoverable by attackers and be manageable enough to avoid being time-consuming.

After an intensive thought process, I modified several open-source services and made a new system that saves all the failed login attempts. I had to analyze the protocols used by hacker tools like ncrack and THC-Hydra. By the end of the day I decided to use ssh, ftp, http and postgresql protocols. I then installed two Ubuntu operating systems on the virtual system XenServer on a Mini-PC.

On one of the Ubuntu systems I installed a tool called loginmon which stores the failed attempts. On the other Ubuntu, I installed Splunk Community edition to analyze the loginmon logs.



GitHub, Inc. [US] https://github.com/nmap/ncrack

services.h	updated license header	2 months ago
shtool	removed empty install.sh and added shtool	9 years ago
targets.cc	updated license header	2 months ago
targets.h	updated license header	2 months ago
timing.cc	updated license header	2 months ago
timing.h	updated license header	2 months ago
utils.cc	fixed length type in base64_decode and associated warning	a month ago
utils.h	fixed length type in base64_decode and associated warning	a month ago

README.md

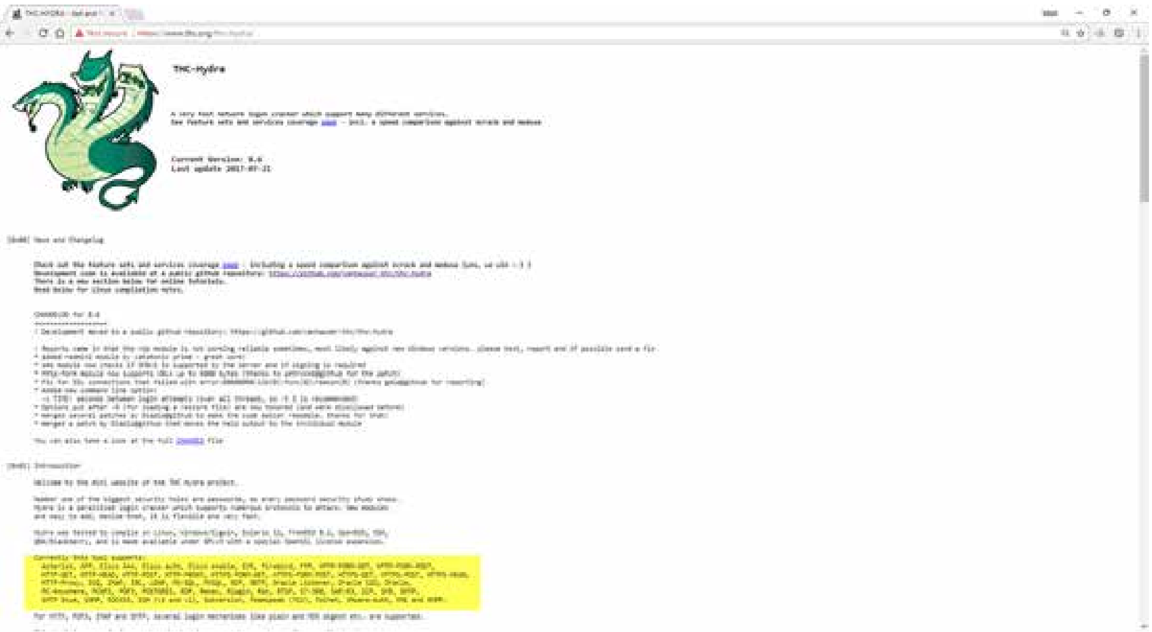
ncrack

Ncrack is a high-speed network authentication cracking tool. It was built to help companies secure their networks by proactively testing all their hosts and networking devices for poor passwords. Security professionals also rely on Ncrack when auditing their clients. Ncrack was designed using a modular approach, a command-line syntax similar to Nmap and a dynamic engine that can adapt its behaviour based on network feedback. It allows for rapid, yet reliable large-scale auditing of multiple hosts.

Ncrack's features include a very flexible interface granting the user full control of network operations, allowing for very sophisticated bruteforcing attacks, timing templates for ease of use, runtime interaction similar to Nmap's and many more. Protocols supported are: SSH, RDP, FTP, Telnet, HTTP(S), POP3(S), IMAP, SMB, VNC, SIP, Redis, PostgreSQL, MySQL, MSSQL, MongoDB, Cassandra, WinRM, OWA.

Be sure to read the Ncrack man page (https://nmap.org/ncrack/man.html) to fully understand Ncrack usage. If you are a developer and want to write your own Ncrack modules, studying the Ncrack Developer's Guide (https://nmap.org/ncrack/devguide.html) would be the first step.

© 2017 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub API Training Shop Blog About

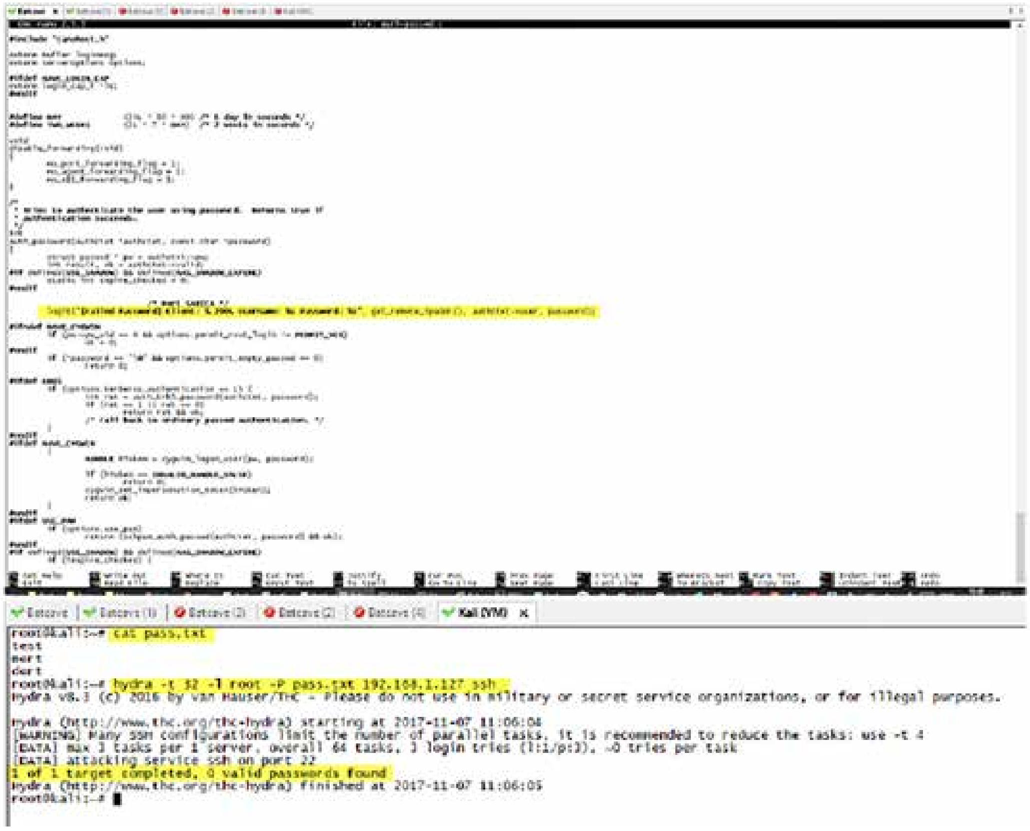


THC-Hydra

Current Version: 8.6
Last update 2017-07-21

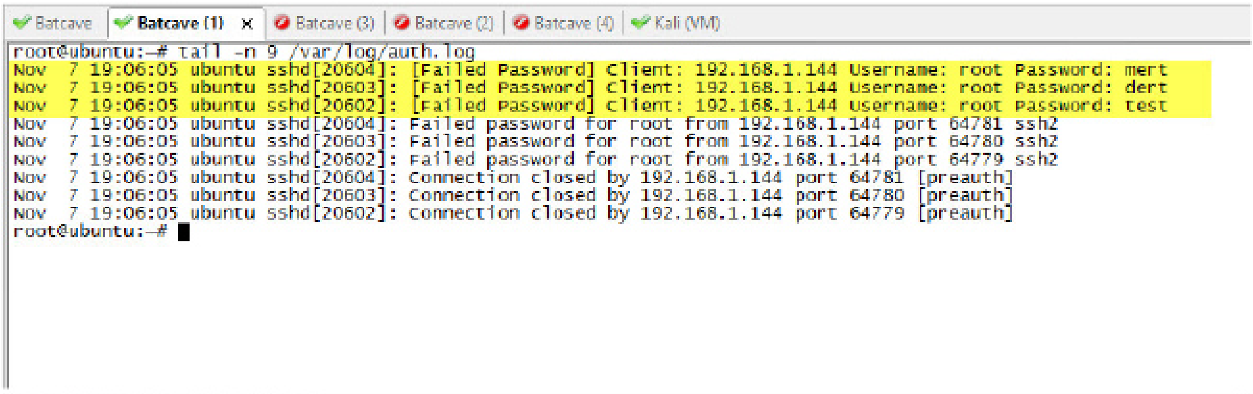
One of the biggest problems I faced was that the passwords used by the server components that made a connection to these protocols (like sshd, postgresql, vsftpd) and their failed login attempts would be stored in clear-text. Therefore I tweaked the source codes of each of the server components.

I downloaded the openssh-7.2p2 package and edited its auth-passwd.c file. Then I downloaded postgres-10.0 package and edited the src/backend/libpq/auth.c file. And finally I downloaded vsftpd-3.0.3 package and edited the pre-login.c and logging.c files.

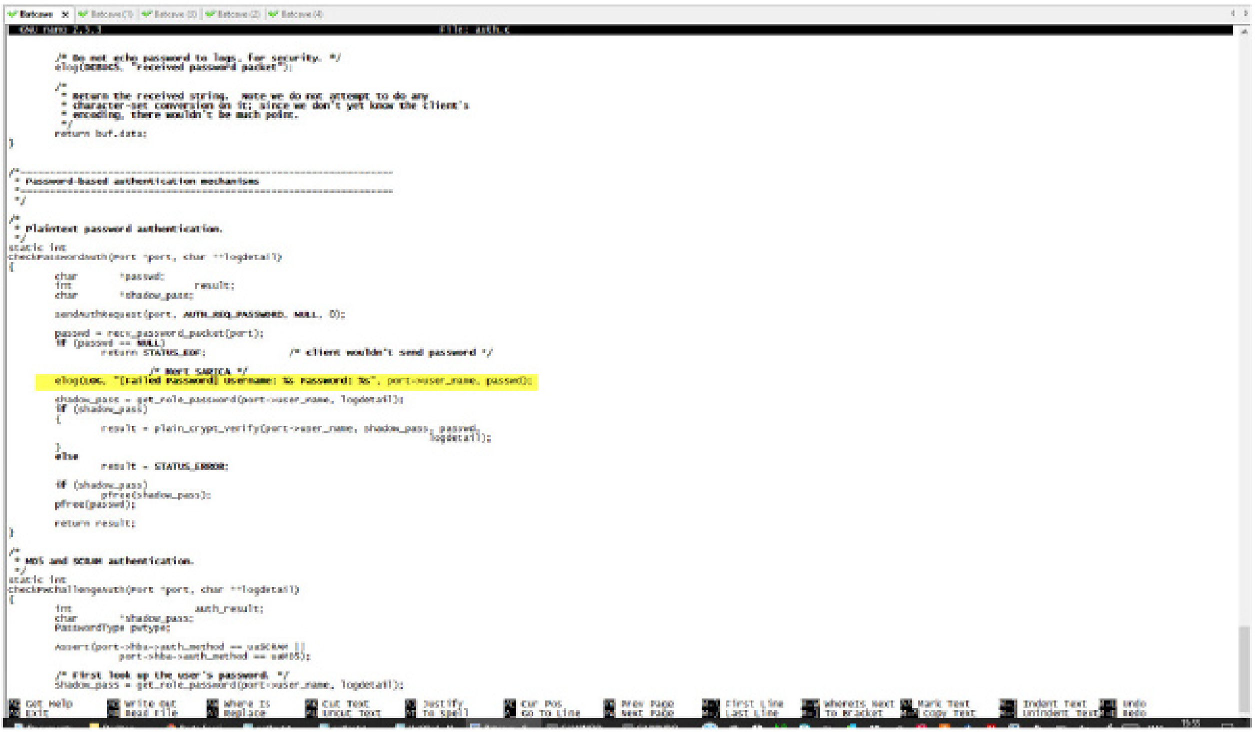


```
  GNU nano 2.5.3                      File: auth-passwd.c

#include "canohost.h"

extern Buffer loginmsg;
extern ServerOptions options;

#ifdef HAVE_LOGIN_CAP
extern login_cap_t *lc;
#endif

#define DAY             (24L * 60 * 60) /* 1 day in seconds */
#define TWO_WEEKS       (2L * 7 * DAY)  /* 2 weeks in seconds */

void
disable_forwarding(void)
{
        no_port_forwarding_flag = 1;
        no_agent_forwarding_flag = 1;
        no_x11_forwarding_flag = 1;
}

/*
 * Tries to authenticate the user using password.  Returns true if
 * authentication succeeds.
 */
int
auth_password(Authctxt *authctxt, const char *password)
{
        struct passwd * pw = authctxt->pw;
        int result, ok = authctxt->valid;
#if defined(USE_SHADOW) && defined(HAS_SHADOW_EXPIRE)
        static int expire_checked = 0;
#endif

                        /* Mert SARICA */
        logit("[Failed Password] Client: %.200s Username: %s Password: %s", get_remote_ipaddr(), authctxt->user, password);
#ifndef HAVE_CYGWIN
        if (pw->pw_uid == 0 && options.permit_root_login != PERMIT_YES)
                ok = 0;
#endif
        if (*password == '\0' && options.permit_empty_passwd == 0)
                return 0;

#ifdef KRB5
        if (options.kerberos_authentication == 1) {
                int ret = auth_krb5_password(authctxt, password);
                if (ret == 1 || ret == 0)
                        return ret && ok;
                /* Fall back to ordinary passwd authentication. */
        }
#endif
#ifdef HAVE_CYGWIN
        {
                HANDLE hToken = cygwin_logon_user(pw, password);

                if (hToken == INVALID_HANDLE_VALUE)
                        return 0;
                cygwin_set_impersonation_token(hToken);
                return ok;
        }
#endif
#ifdef USE_PAM
        if (options.use_pam)
                return (sshpam_auth_passwd(authctxt, password) && ok);
#endif
#if defined(USE_SHADOW) && defined(HAS_SHADOW_EXPIRE)
        if (!expire_checked) {

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    ^Y Prev Page  M-\ First Line  M-W WhereIs Next  ^^ Mark Text   M-} Indent Text    M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line ^V Next Page  M-/ Last Line   M-] To Bracket    M-6 Copy Text   M-{ Unindent Text  M-E Redo
```

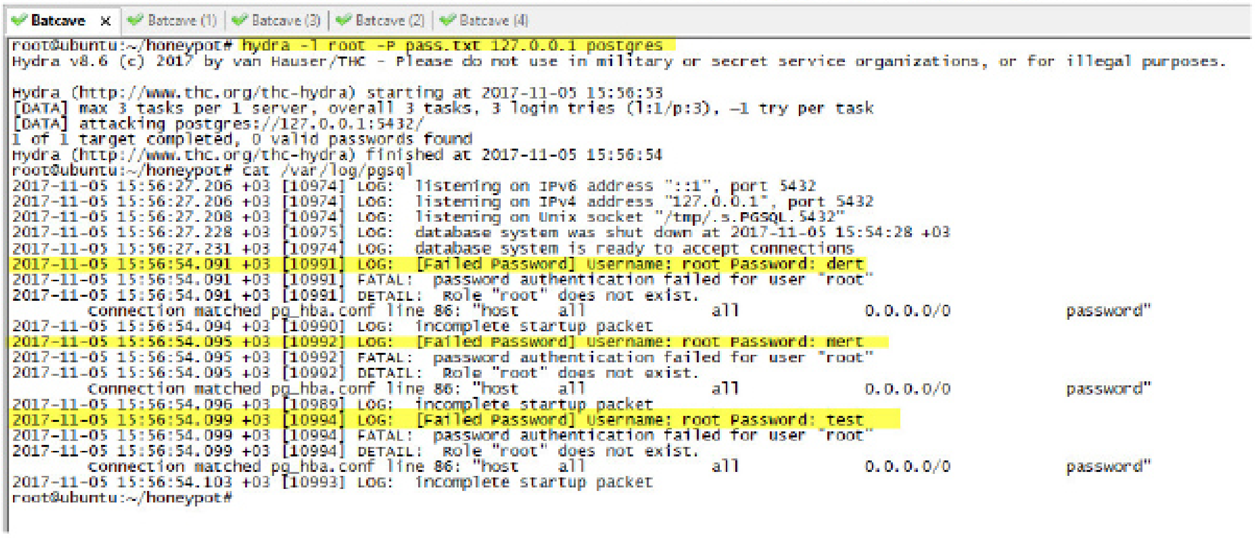
```
root@kali:~# cat pass.txt
test
mert
dert
root@kali:~# hydra -t 32 -l root -P pass.txt 192.168.1.127 ssh
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-11-07 11:06:04
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 3 tasks per 1 server, overall 64 tasks, 3 login tries (l:1/p:3), ~0 tries per task
[DATA] attacking service ssh on port 22
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-11-07 11:06:05
root@kali:~# 
```



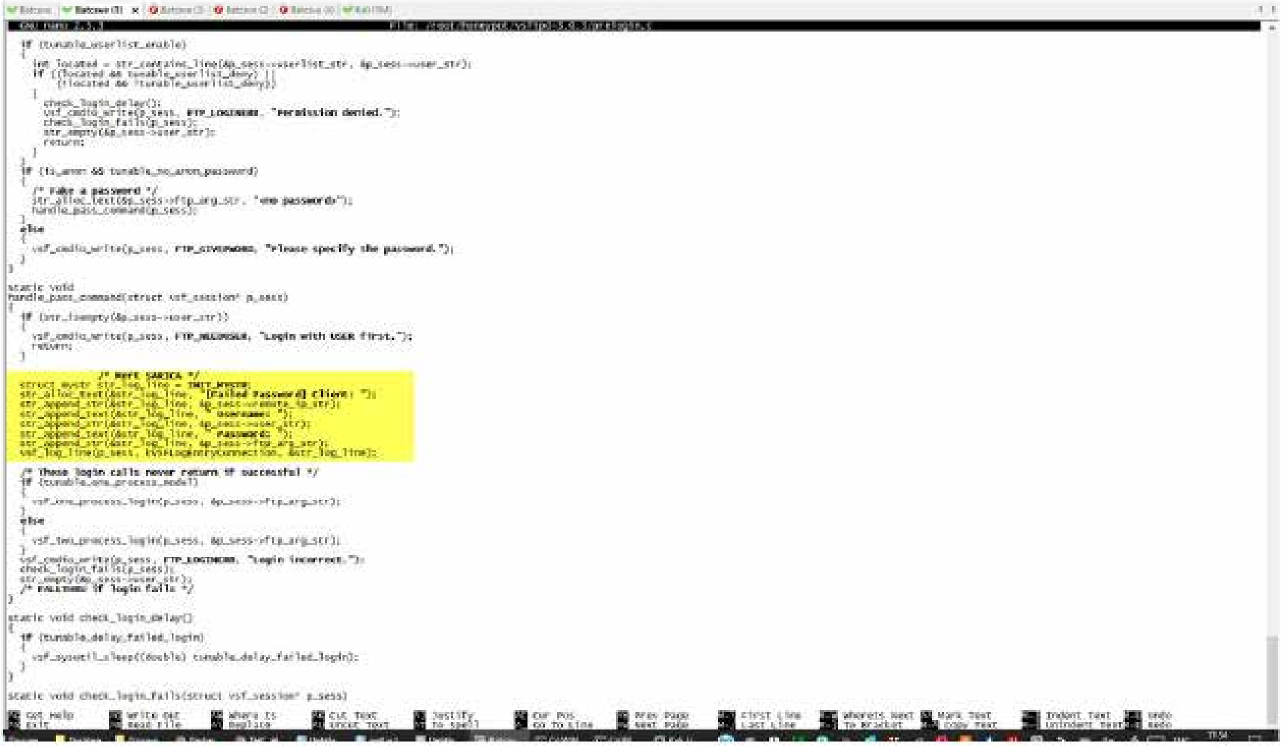
```
root@ubuntu:~# tail -n 9 /var/log/auth.log
Nov  7 19:06:05 ubuntu sshd[20604]: [Failed Password] Client: 192.168.1.144 Username: root Password: mert
Nov  7 19:06:05 ubuntu sshd[20603]: [Failed Password] Client: 192.168.1.144 Username: root Password: dert
Nov  7 19:06:05 ubuntu sshd[20602]: [Failed Password] Client: 192.168.1.144 Username: root Password: test
Nov  7 19:06:05 ubuntu sshd[20604]: Failed password for root from 192.168.1.144 port 64781 ssh2
Nov  7 19:06:05 ubuntu sshd[20603]: Failed password for root from 192.168.1.144 port 64780 ssh2
Nov  7 19:06:05 ubuntu sshd[20602]: Failed password for root from 192.168.1.144 port 64779 ssh2
Nov  7 19:06:05 ubuntu sshd[20604]: Connection closed by 192.168.1.144 port 64781 [preauth]
Nov  7 19:06:05 ubuntu sshd[20603]: Connection closed by 192.168.1.144 port 64780 [preauth]
Nov  7 19:06:05 ubuntu sshd[20602]: Connection closed by 192.168.1.144 port 64779 [preauth]
root@ubuntu:~# 
```



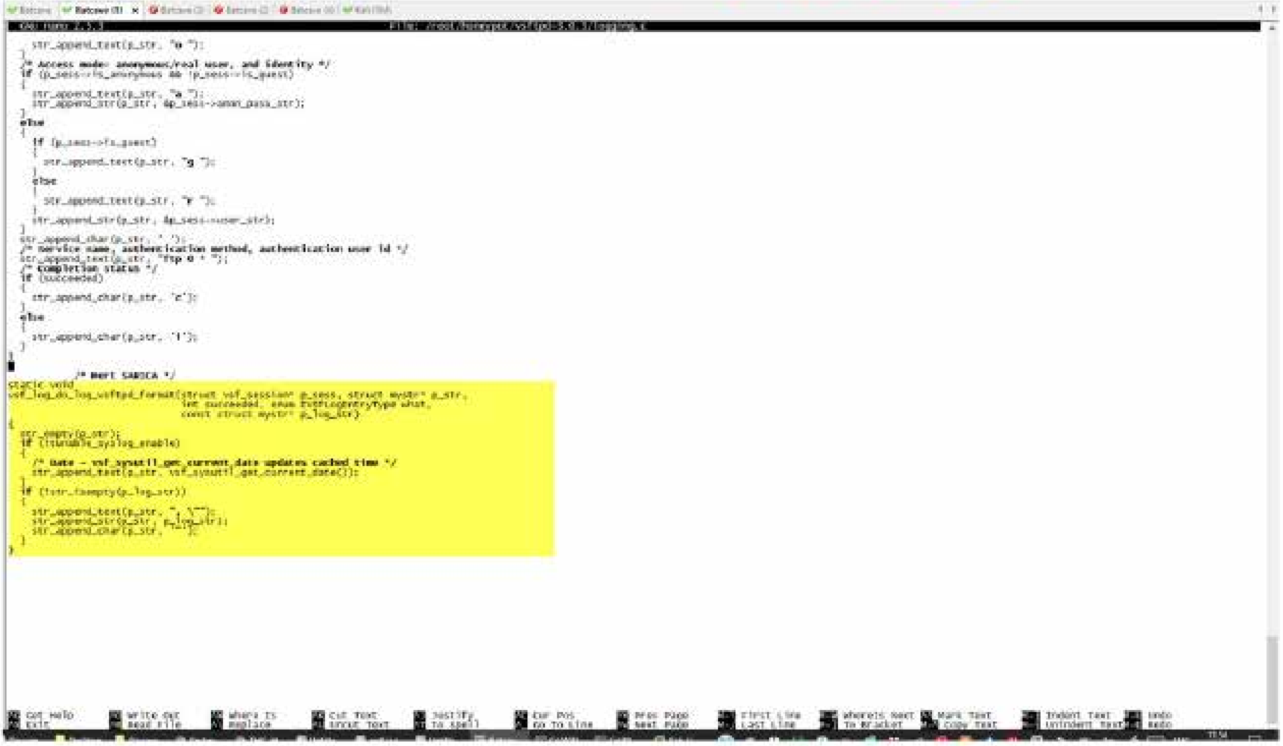
```
  GNU nano 2.5.3                                          File: auth.c

        /* Do not echo password to logs, for security. */
        elog(DEBUG5, "received password packet");

        /*
         * Return the received string.  Note we do not attempt to do any
         * character-set conversion on it; since we don't yet know the client's
         * encoding, there wouldn't be much point.
         */
        return buf.data;
}


/*----------------------------------------------------------------
 * Password-based authentication mechanisms
 *----------------------------------------------------------------
 */

/*
 * Plaintext password authentication.
 */
static int
CheckPasswordAuth(Port *port, char **logdetail)
{
        char       *passwd;
        int                     result;
        char       *shadow_pass;

        sendAuthRequest(port, AUTH_REQ_PASSWORD, NULL, 0);

        passwd = recv_password_packet(port);
        if (passwd == NULL)
                return STATUS_EOF;              /* client wouldn't send password */

                    /* MERT SARICA */
        elog(LOG, "[Failed Password] Username: %s Password: %s", port->user_name, passwd);

        shadow_pass = get_role_password(port->user_name, logdetail);
        if (shadow_pass)
        {
                result = plain_crypt_verify(port->user_name, shadow_pass, passwd,
                                                                        logdetail);
        }
        else
                result = STATUS_ERROR;

        if (shadow_pass)
                pfree(shadow_pass);
        pfree(passwd);

        return result;
}

/*
 * MD5 and SCRAM authentication.
 */
static int
CheckMD5ChallengeAuth(Port *port, char **logdetail)
{
        int                     auth_result;
        char       *shadow_pass;
        PasswordType pwtype;

        Assert(port->hba->auth_method == uaSCRAM ||
                   port->hba->auth_method == uaMD5);

        /* First look up the user's password. */
        shadow_pass = get_role_password(port->user_name, logdetail);
```



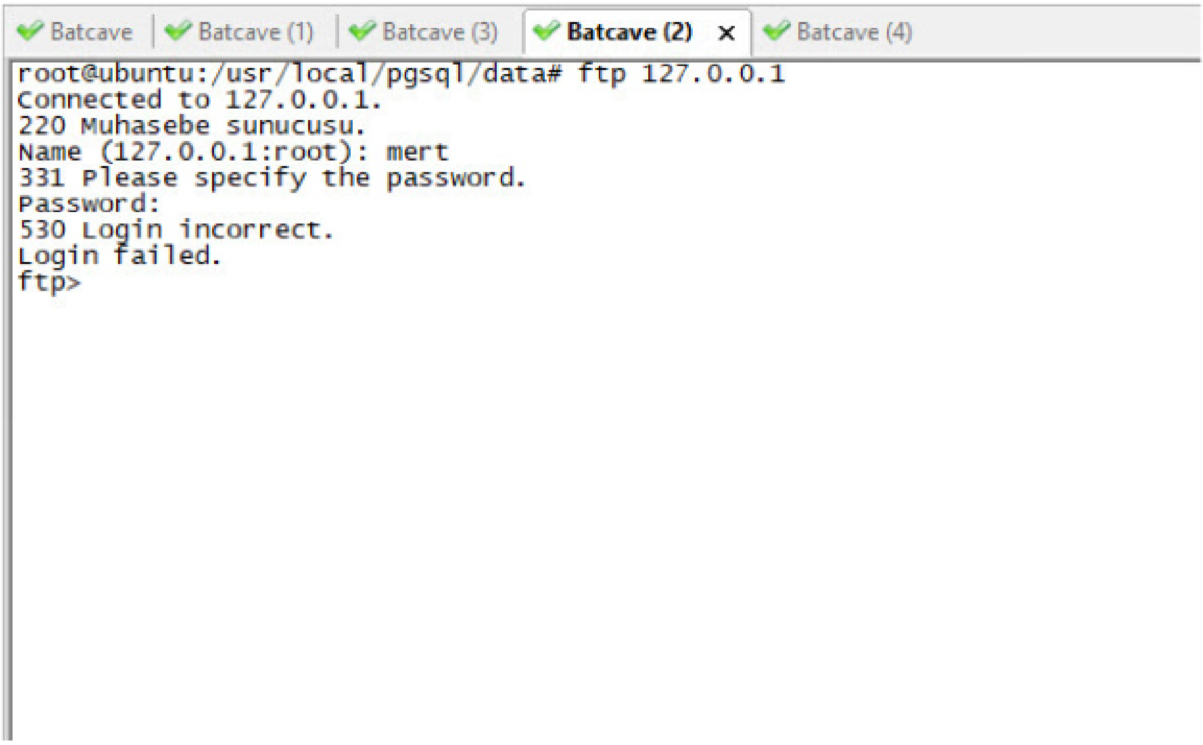
```
root@ubuntu:~/honeypot# hydra -l root -P pass.txt 127.0.0.1 postgres
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-11-05 15:56:53
[DATA] max 3 tasks per 1 server, overall 3 tasks, 3 login tries (l:1/p:3), ~1 try per task
[DATA] attacking postgres://127.0.0.1:5432/
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-11-05 15:56:54
root@ubuntu:~/honeypot# cat /var/log/pgsql
2017-11-05 15:56:27.206 +03 [10974] LOG:  listening on IPv6 address "::1", port 5432
2017-11-05 15:56:27.206 +03 [10974] LOG:  listening on IPv4 address "127.0.0.1", port 5432
2017-11-05 15:56:27.208 +03 [10974] LOG:  listening on Unix socket "/tmp/.s.PGSQL.5432"
2017-11-05 15:56:27.228 +03 [10975] LOG:  database system was shut down at 2017-11-05 15:54:28 +03
2017-11-05 15:56:27.231 +03 [10974] LOG:  database system is ready to accept connections
2017-11-05 15:56:54.091 +03 [10991] LOG:  [Failed Password] Username: root Password: dert
2017-11-05 15:56:54.091 +03 [10991] FATAL:  password authentication failed for user "root"
2017-11-05 15:56:54.091 +03 [10991] DETAIL:  Role "root" does not exist.
        Connection matched pg_hba.conf line 86: "host    all             all             0.0.0.0/0               password"
2017-11-05 15:56:54.094 +03 [10990] LOG:  incomplete startup packet
2017-11-05 15:56:54.095 +03 [10992] LOG:  [Failed Password] Username: root Password: mert
2017-11-05 15:56:54.095 +03 [10992] FATAL:  password authentication failed for user "root"
2017-11-05 15:56:54.095 +03 [10992] DETAIL:  Role "root" does not exist.
        Connection matched pg_hba.conf line 86: "host    all             all             0.0.0.0/0               password"
2017-11-05 15:56:54.096 +03 [10989] LOG:  incomplete startup packet
2017-11-05 15:56:54.099 +03 [10994] LOG:  [Failed Password] Username: root Password: test
2017-11-05 15:56:54.099 +03 [10994] FATAL:  password authentication failed for user "root"
2017-11-05 15:56:54.099 +03 [10994] DETAIL:  Role "root" does not exist.
        Connection matched pg_hba.conf line 86: "host    all             all             0.0.0.0/0               password"
2017-11-05 15:56:54.103 +03 [10993] LOG:  incomplete startup packet
root@ubuntu:~/honeypot#
```



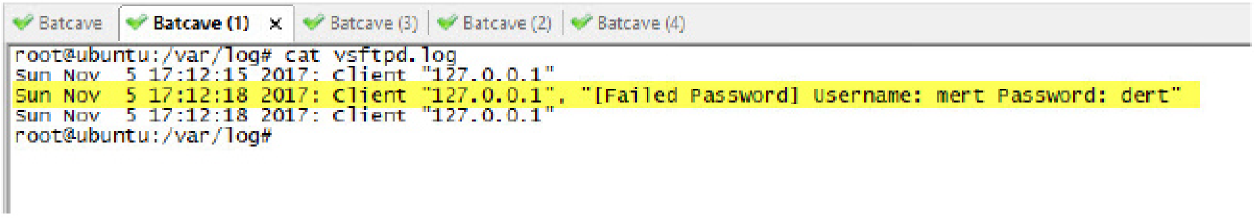
```
  GNU nano 2.5.3                    File: /root/honeypot/vsftpd-3.0.3/prelogin.c

  if (tunable_userlist_enable)
  {
    int located = str_contains_line(&p_sess->userlist_str, &p_sess->user_str);
    if ((located && tunable_userlist_deny) ||
        (!located && !tunable_userlist_deny))
    {
      check_login_delay();
      vsf_cmdio_write(p_sess, FTP_LOGINERR, "Permission denied.");
      check_login_fails(p_sess);
      str_empty(&p_sess->user_str);
      return;
    }
  }
  if (is_anon && tunable_no_anon_password)
  {
    /* Fake a password */
    str_alloc_text(&p_sess->ftp_arg_str, "<no password>");
    handle_pass_command(p_sess);
  }
  else
  {
    vsf_cmdio_write(p_sess, FTP_GIVEPWORD, "Please specify the password.");
  }
}

static void
handle_pass_command(struct vsf_session* p_sess)
{
  if (str_isempty(&p_sess->user_str))
  {
    vsf_cmdio_write(p_sess, FTP_NEEDUSER, "Login with USER first.");
    return;
  }

                /* Mert SARICA */
  struct mystr str_log_line = INIT_MYSTR;
  str_alloc_text(&str_log_line, "[Failed Password] Client: ");
  str_append_str(&str_log_line, &p_sess->remote_ip_str);
  str_append_text(&str_log_line, ", Username: ");
  str_append_str(&str_log_line, &p_sess->user_str);
  str_append_text(&str_log_line, ", Password: ");
  str_append_str(&str_log_line, &p_sess->ftp_arg_str);
  vsf_log_line(p_sess, kVSFLogEntryConnection, &str_log_line);

  /* These login calls never return if successful */
  if (tunable_one_process_model)
  {
    vsf_one_process_login(p_sess, &p_sess->ftp_arg_str);
  }
  else
  {
    vsf_two_process_login(p_sess, &p_sess->ftp_arg_str);
  }
  vsf_cmdio_write(p_sess, FTP_LOGINERR, "Login incorrect.");
  check_login_fails(p_sess);
  str_empty(&p_sess->user_str);
  /* FALLTHRU if login fails */
}

static void check_login_delay()
{
  if (tunable_delay_failed_login)
  {
    vsf_sysutil_sleep((double) tunable_delay_failed_login);
  }
}

static void check_login_fails(struct vsf_session* p_sess)
```



```
  GNU nano 2.5.3                    File: /root/honeypot/vsftpd-3.0.3/logging.c

    str_append_text(p_str, "o ");
  }
  /* Access mode: anonymous/real user, and identity */
  if (p_sess->is_anonymous && !p_sess->is_guest)
  {
    str_append_text(p_str, "a ");
    str_append_str(p_str, &p_sess->anon_pass_str);
  }
  else
  {
    if (p_sess->is_guest)
    {
      str_append_text(p_str, "g ");
    }
    else
    {
      str_append_text(p_str, "r ");
    }
    str_append_str(p_str, &p_sess->user_str);
  }
  str_append_char(p_str, ' ');
  /* Service name, authentication method, authentication user id */
  str_append_text(p_str, "ftp 0 * ");
  /* Completion status */
  if (succeeded)
  {
    str_append_char(p_str, 'c');
  }
  else
  {
    str_append_char(p_str, 'i');
  }
}

            /* Mert SARICA */
static void
vsf_log_do_log_vsftpd_format(struct vsf_session* p_sess, struct mystr* p_str,
                             int succeeded, enum EVSFLogEntryType what,
                             const struct mystr* p_log_str)
{
  str_empty(p_str);
  if (!tunable_syslog_enable)
  {
    /* Date - vsf_sysutil_get_current_date updates cached time */
    str_append_text(p_str, vsf_sysutil_get_current_date());
  }
  if (!str_isempty(p_log_str))
  {
    str_append_text(p_str, ", \"");
    str_append_str(p_str, p_log_str);
    str_append_char(p_str, '"');
  }
}
```

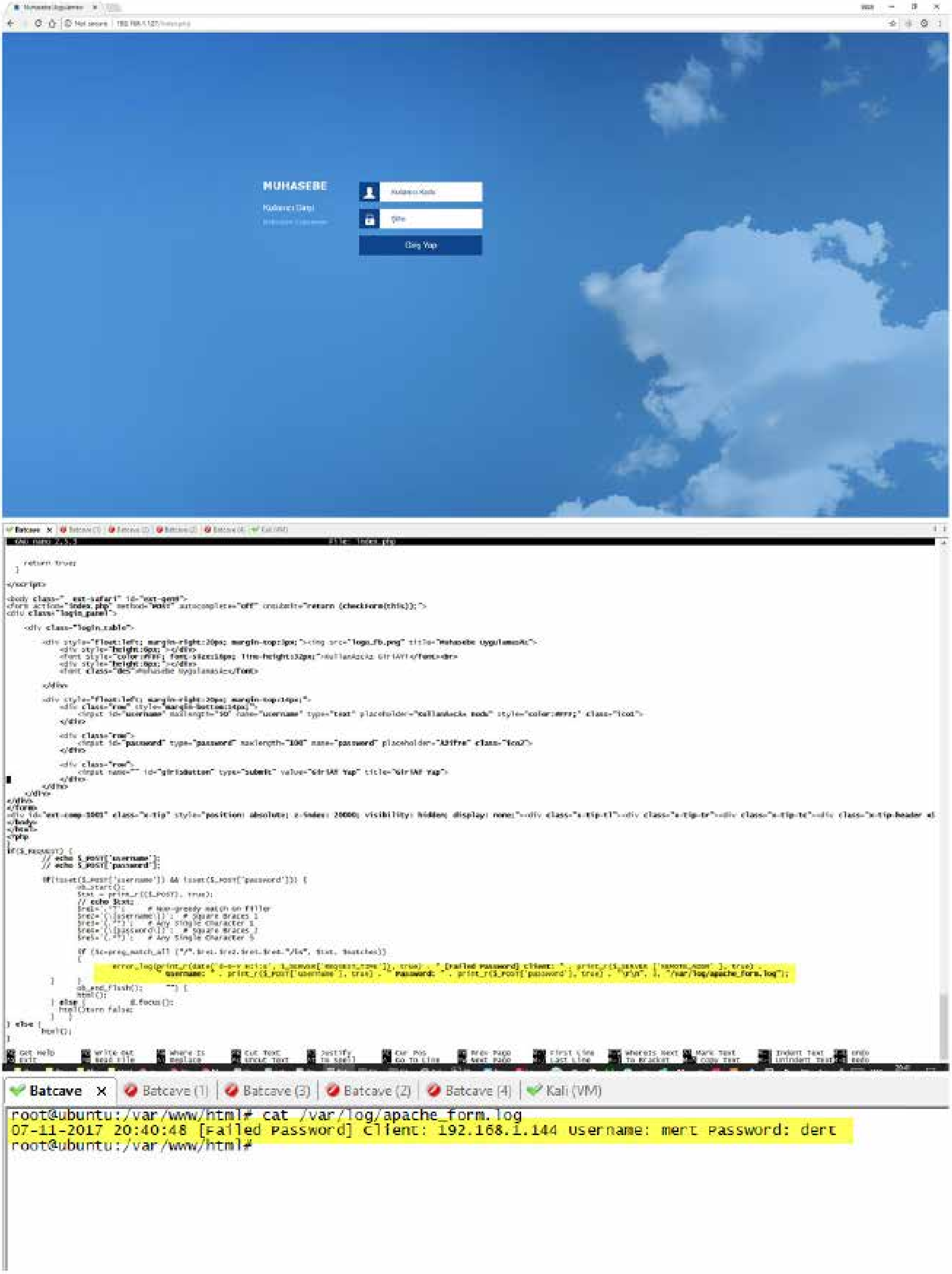


```
root@ubuntu:/usr/local/pgsql/data# ftp 127.0.0.1
Connected to 127.0.0.1.
220 Muhasebe sunucusu.
Name (127.0.0.1:root): mert
331 Please specify the password.
Password:
530 Login incorrect.
Login failed.
ftp>
```



```
root@ubuntu:/var/log# cat vsftpd.log
Sun Nov  5 17:12:15 2017: Client "127.0.0.1"
Sun Nov  5 17:12:18 2017: Client "127.0.0.1", "[Failed Password] Username: mert Password: dert"
Sun Nov  5 17:12:18 2017: Client "127.0.0.1"
root@ubuntu:/var/log#
```

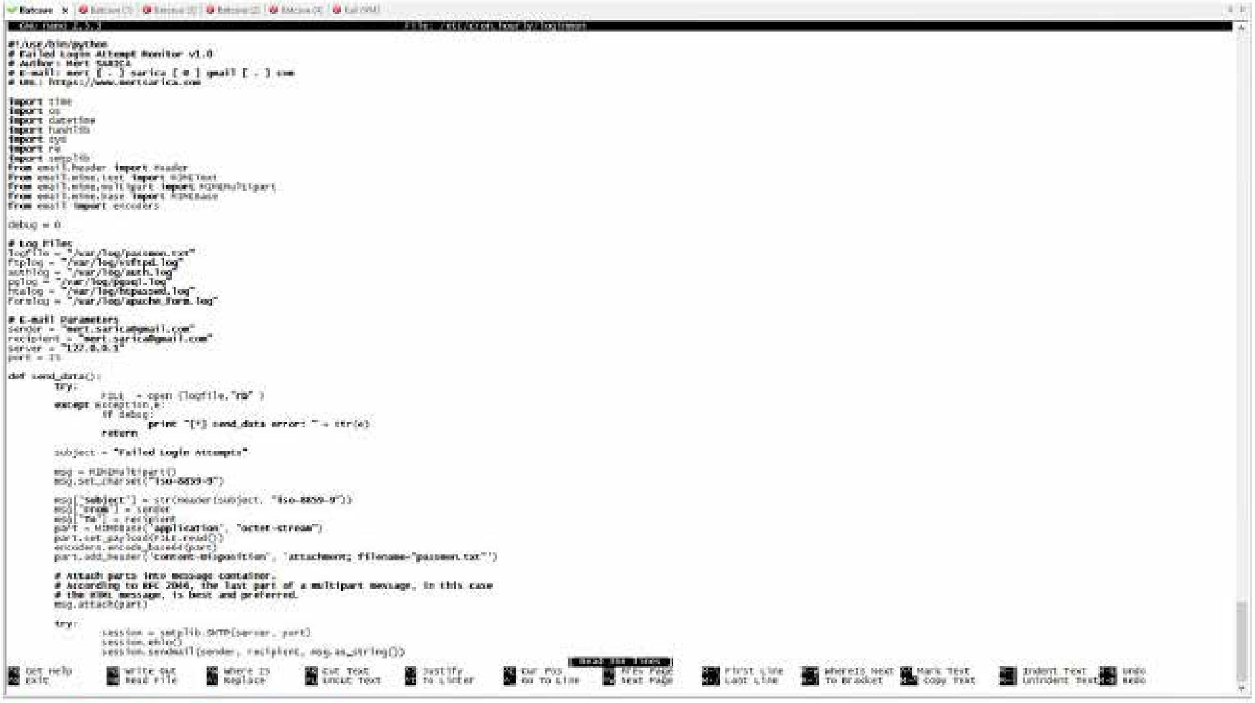
As the last step, I enabled the index.php file on the main folder of the web server to store the failed login attempts.



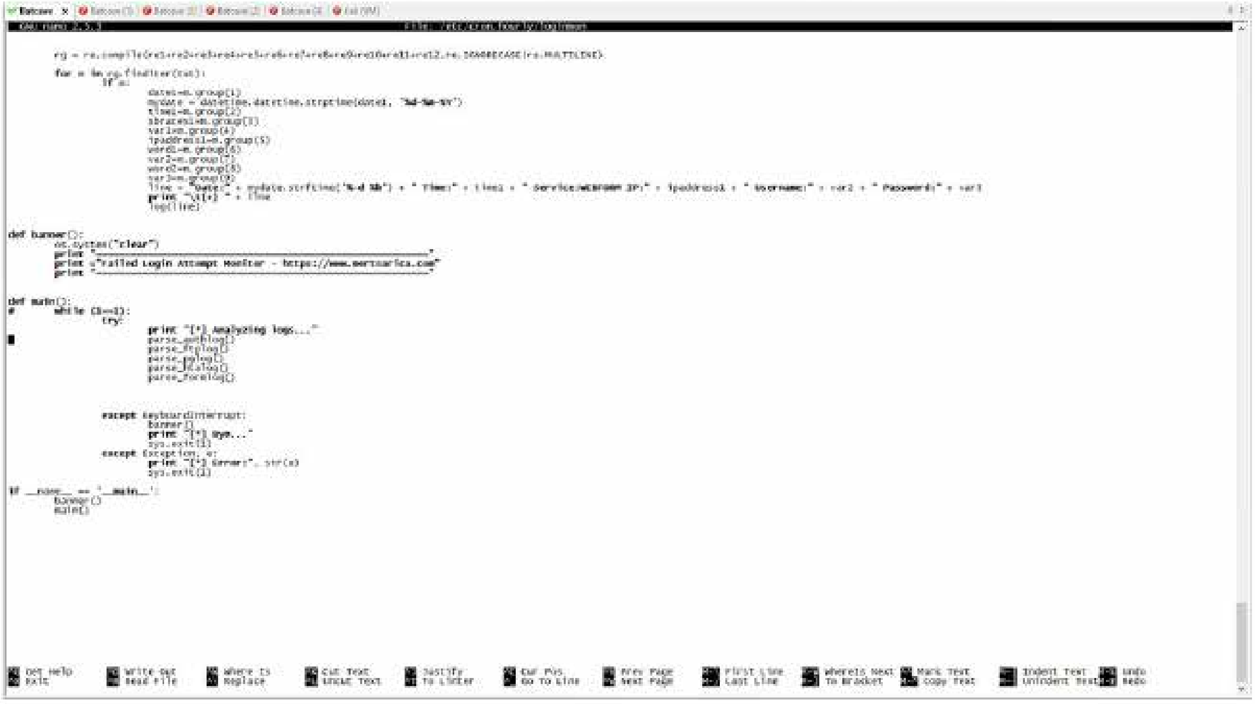
```
      return true;
  }
</script>
<body class=" ext-safari" id="ext-gen9">
<form action="index.php" method="POST" autocomplete="off" onsubmit="return (checkform(this));">
<div class="login_panel">

    <div class="login_table">

        <div style="float:left; margin-right:20px; margin-top:1px;"><img src="logo_fb.png" title="Muhasebe Uygulaması">
            <div style="height:8px;"></div>
            <font style="color:#FFF; font-size:16px; line-height:32px;">Kullanıcı Girişi</font><br>
            <div style="height:8px;"></div>
            <font class="des">Muhasebe Uygulaması</font>

        </div>

        <div style="float:left; margin-right:20px; margin-top:14px;">
            <div class="row" style="margin-bottom:14px;">
                <input id="username" maxlength="50" name="username" type="text" placeholder="Kullanıcı Kodu" style="color:#FFF;" class="ico1">
            </div>

            <div class="row">
                <input id="password" type="password" maxlength="100" name="password" placeholder="Şifre" class="ico2">
            </div>

            <div class="row">
                <input name="" id="girisbutton" type="submit" value="Giriş Yap" title="Giriş Yap">
            </div>
        </div>
    </div>
</div>
</form>
<div id="ext-comp-1001" class="x-tip" style="position: absolute; z-index: 20000; visibility: hidden; display: none;"><div class="x-tip-tl"><div class="x-tip-tr"><div class="x-tip-tc"><div class="x-tip-header x
</body>
</html>
<?php
}
if($_REQUEST) {
        // echo $_POST['username'];
        // echo $_POST['password'];

        if(isset($_POST['username']) && isset($_POST['password'])) {
                ob_start();
                $txt = print_r(($_POST), true);
                // echo $txt;
                $re1='.*?';     # Non-greedy match on filler
                $re2='(\\[username\\])';   # Square Braces 1
                $re3='(.*?)';   # Any Single Character 1
                $re4='(\\[password\\])';   # Square Braces 2
                $re5='(.*?)';   # Any Single Character 5

                if ($c=preg_match_all ("/".$re1.$re2.$re3.$re4."/is", $txt, $matches))
                {
                        error_log(print_r(date('d-m-Y H:i:s', $_SERVER['REQUEST_TIME']), true) . " [Failed Password] Client: " . print_r($_SERVER ['REMOTE_ADDR'], true) .
                                " Username: " . print_r($_POST['username'], true) . " Password: " . print_r($_POST['password'], true) . "\r\n", 3, "/var/log/apache_form.log");
                }
        }
                ob_end_flush();     ""); {
                html();
        } else {        d.focus();
          html()turn false;
        }   }
} else {
        html();
}
```

```
root@ubuntu:/var/www/html# cat /var/log/apache_form.log
07-11-2017 20:40:48 [Failed Password] Client: 192.168.1.144 Username: mert Password: dert
root@ubuntu:/var/www/html#
```

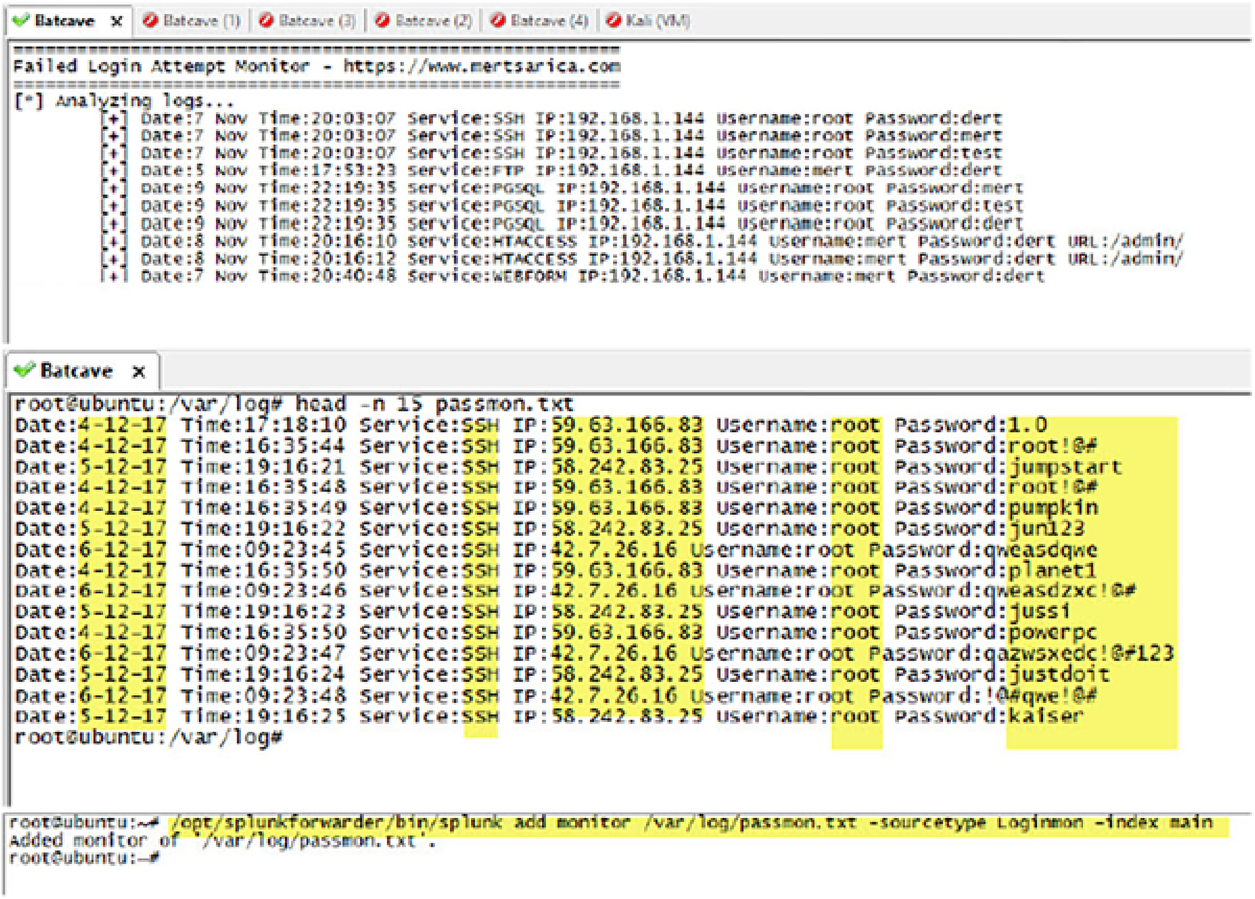
I configured the loginmon Python tool I built to send an hourly report of the failed passwords, usernames, ip addresses, and dates stored in clear text in /var/log folder to Splunk.



```
  GNU nano 2.5.3                                File: /etc/cron.hourly/loginmon

#!/usr/bin/python
# Failed Login Attempt Monitor v1.0
# Author: Mert SARICA
# E-mail: mert [ . ] sarica [ @ ] gmail [ . ] com
# URL: https://www.mertsarica.com

import time
import os
import datetime
import hashlib
import sys
import re
import smtplib
from email.header import Header
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders

debug = 0

# Log Files
logfile = "/var/log/passmon.txt"
ftplog = "/var/log/vsftpd.log"
authlog = "/var/log/auth.log"
pglog = "/var/log/pgsql.log"
htalog = "/var/log/htpasswd.log"
formlog = "/var/log/apache_form.log"

# E-mail Parameters
sender = "mert.sarica@gmail.com"
recipient = "mert.sarica@gmail.com"
server = "127.0.0.1"
port = 25

def send_data():
        try:
                FILE  = open (logfile,"rb" )
        except Exception,e:
                if debug:
                        print "[*] send_data error: " + str(e)
                return

        subject = "Failed Login Attempts"

        msg = MIMEMultipart()
        msg.set_charset("iso-8859-9")

        msg['Subject'] = str(Header(subject, "iso-8859-9"))
        msg['From'] = sender
        msg['To'] = recipient
        part = MIMEBase('application', "octet-stream")
        part.set_payload(FILE.read())
        encoders.encode_base64(part)
        part.add_header('Content-Disposition', 'attachment; filename="passmon.txt"')

        # Attach parts into message container.
        # According to RFC 2046, the last part of a multipart message, in this case
        # the HTML message, is best and preferred.
        msg.attach(part)

        try:
                session = smtplib.SMTP(server, port)
                session.ehlo()
                session.sendmail(sender, recipient, msg.as_string())
```



```
  GNU nano 2.5.3                                File: /etc/cron.hourly/loginmon

        rg = re.compile(re1+re2+re3+re4+re5+re6+re7+re8+re9+re10+re11+re12,re.IGNORECASE|re.MULTILINE)

        for m in rg.finditer(txt):
                if m:
                        date1=m.group(1)
                        mydate = datetime.datetime.strptime(date1, "%d-%m-%Y")
                        time1=m.group(2)
                        sbraces1=m.group(3)
                        var1=m.group(4)
                        ipaddress1=m.group(5)
                        word1=m.group(6)
                        var2=m.group(7)
                        word2=m.group(8)
                        var3=m.group(9)
                        line = "Date:" + mydate.strftime('%-d %b') + " Time:" + time1 + " Service:WEBFORM IP:" + ipaddress1 + " Username:" + var2 + " Password:" + var3
                        print "\t[+] " + line
                        log(line)

def banner():
        os.system("clear")
        print "_______________________________________________________"
        print u"Failed Login Attempt Monitor - https://www.mertsarica.com"
        print "_______________________________________________________"

def main():
#       while (1==1):
                try:
                        print "[*] Analyzing logs..."
                        parse_authlog()
                        parse_ftplog()
                        parse_pglog()
                        parse_htalog()
                        parse_formlog()


                except KeyboardInterrupt:
                        banner()
                        print "[*] Bye..."
                        sys.exit(1)
                except Exception, e:
                        print "[*] Error:", str(e)
                        sys.exit(1)

if __name__ == '__main__':
        banner()
        main()
```



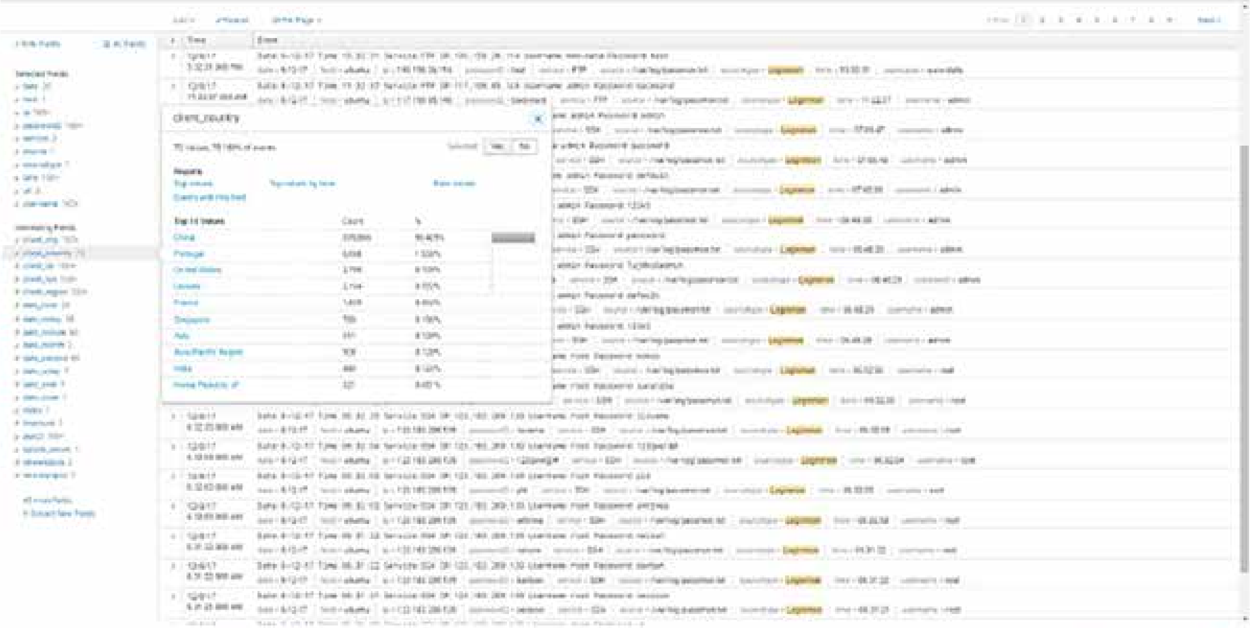
```
==========================================================
Failed Login Attempt Monitor - https://www.mertsarica.com
==========================================================
[*] Analyzing logs...
        [+] Date:7 Nov Time:20:03:07 Service:SSH IP:192.168.1.144 Username:root Password:dert
        [+] Date:7 Nov Time:20:03:07 Service:SSH IP:192.168.1.144 Username:root Password:mert
        [+] Date:7 Nov Time:20:03:07 Service:SSH IP:192.168.1.144 Username:root Password:test
        [+] Date:5 Nov Time:17:53:23 Service:FTP IP:192.168.1.144 Username:mert Password:dert
        [+] Date:9 Nov Time:22:19:35 Service:PGSQL IP:192.168.1.144 Username:root Password:mert
        [+] Date:9 Nov Time:22:19:35 Service:PGSQL IP:192.168.1.144 Username:root Password:test
        [+] Date:9 Nov Time:22:19:35 Service:PGSQL IP:192.168.1.144 Username:root Password:dert
        [+] Date:8 Nov Time:20:16:10 Service:HTACCESS IP:192.168.1.144 Username:mert Password:dert URL:/admin/
        [+] Date:8 Nov Time:20:16:12 Service:HTACCESS IP:192.168.1.144 Username:mert Password:dert URL:/admin/
        [+] Date:7 Nov Time:20:40:48 Service:WEBFORM IP:192.168.1.144 Username:mert Password:dert
```

```
root@ubuntu:/var/log# head -n 15 passmon.txt
Date:4-12-17 Time:17:18:10 Service:SSH IP:59.63.166.83 Username:root Password:1.0
Date:4-12-17 Time:16:35:44 Service:SSH IP:59.63.166.83 Username:root Password:root!@#
Date:5-12-17 Time:19:16:21 Service:SSH IP:58.242.83.25 Username:root Password:jumpstart
Date:4-12-17 Time:16:35:48 Service:SSH IP:59.63.166.83 Username:root Password:root!@#
Date:4-12-17 Time:16:35:49 Service:SSH IP:59.63.166.83 Username:root Password:pumpkin
Date:5-12-17 Time:19:16:22 Service:SSH IP:58.242.83.25 Username:root Password:jun123
Date:6-12-17 Time:09:23:45 Service:SSH IP:42.7.26.16 Username:root Password:qweasdqwe
Date:4-12-17 Time:16:35:50 Service:SSH IP:59.63.166.83 Username:root Password:planet1
Date:6-12-17 Time:09:23:46 Service:SSH IP:42.7.26.16 Username:root Password:qweasdzxc!@#
Date:5-12-17 Time:19:16:23 Service:SSH IP:58.242.83.25 Username:root Password:jussi
Date:4-12-17 Time:16:35:50 Service:SSH IP:59.63.166.83 Username:root Password:powerpc
Date:6-12-17 Time:09:23:47 Service:SSH IP:42.7.26.16 Username:root Password:qazwsxedc!@#123
Date:5-12-17 Time:19:16:24 Service:SSH IP:58.242.83.25 Username:root Password:justdoit
Date:6-12-17 Time:09:23:48 Service:SSH IP:42.7.26.16 Username:root Password:!@#qwe!@#
Date:5-12-17 Time:19:16:25 Service:SSH IP:58.242.83.25 Username:root Password:kaiser
root@ubuntu:/var/log#
```

```
root@ubuntu:~# /opt/splunkforwarder/bin/splunk add monitor /var/log/passmon.txt -sourcetype Loginmon -index main
Added monitor of '/var/log/passmon.txt'.
root@ubuntu:~#
```

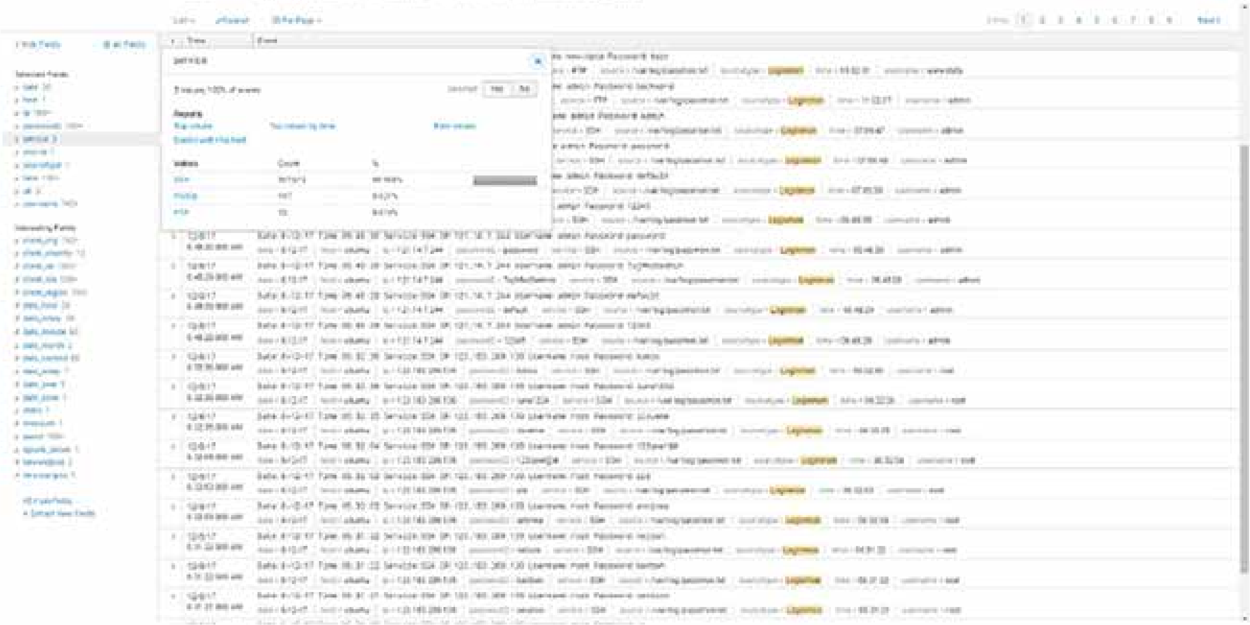
By the end of a month, I analyzed the 500,000 logs gathered on Splunk and discovered that:

- The most common attack type was the dictionary attack.
- Most of the attacks were rooted from China.
- The most targeted service was SSHD.
- The most attempted username was root.
- And the most attempted password was 1234.



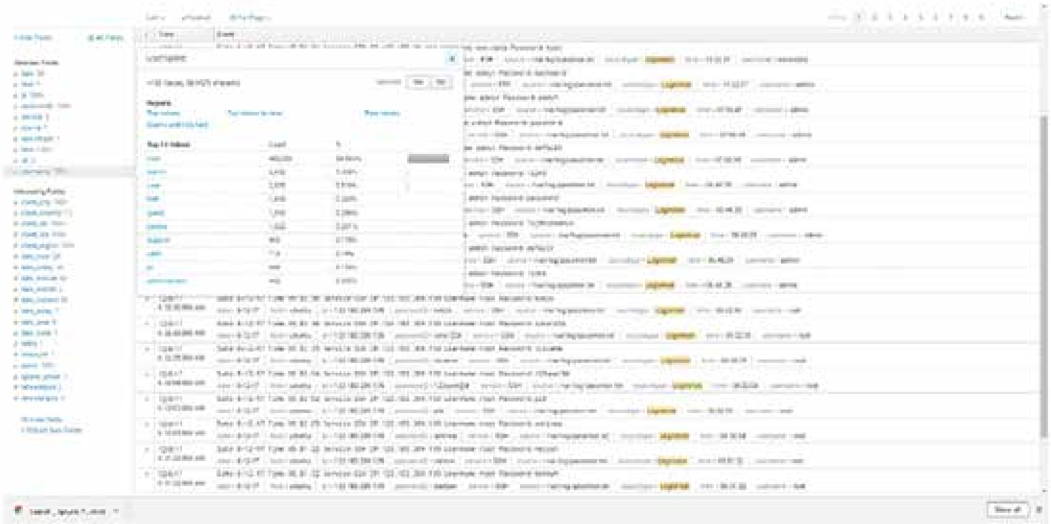
client_country

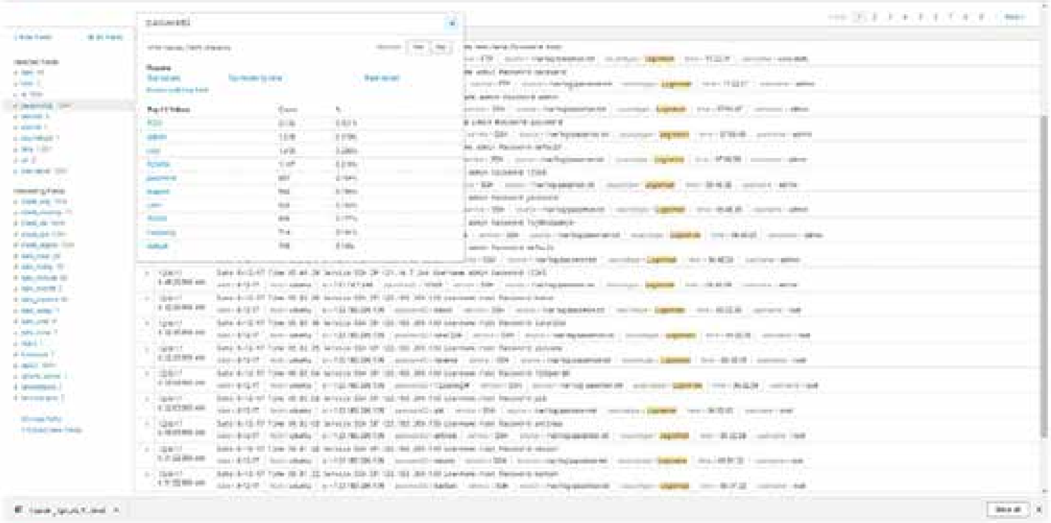
Top 10 Values	Count	%
China	[illegible]	[illegible]
Portugal	[illegible]	[illegible]
United States	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
France	[illegible]	[illegible]
Singapore	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
Asia/Pacific Region	[illegible]	[illegible]
India	[illegible]	[illegible]
Korea, Republic of	[illegible]	[illegible]

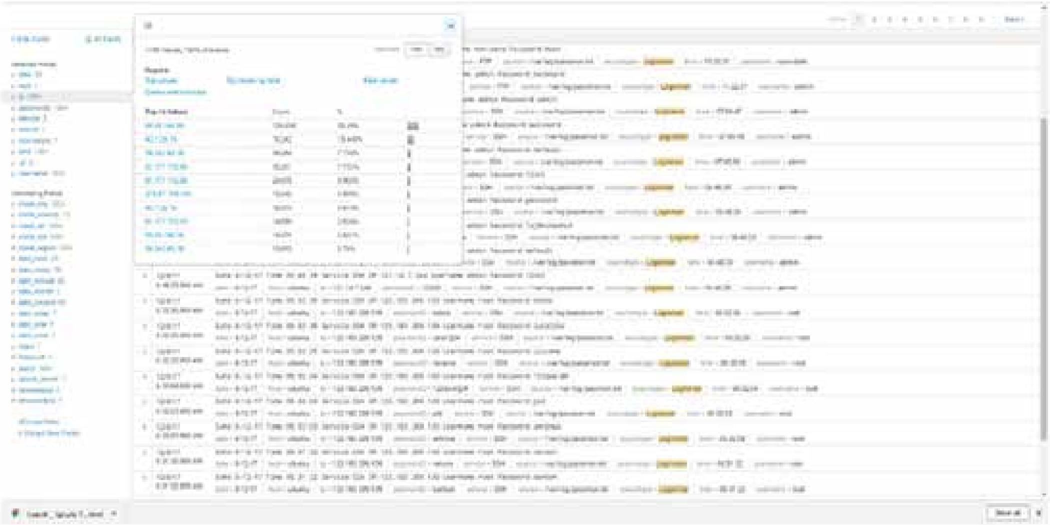


service

Values	Count	%
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]







Hope to see you in the next article. I wish you all a secure day!

Ulaş Fırat Özdemir • htcnian@gmail.com

KRACK
(Key Reinstallation Attack)

What is KRACK?

KRACK is the vulnerability on the so-called safe WPA2 protocol, which protects the wireless communication between the user and the internet provider device.The WPA2 protocol securely authenticates by exchanging passwords between devices and if the shared password is correct, it authorizes and allowing us to connect to the Internet. This protocol also encrypts the communication between the two devices, providing an encrypted connection. Thus, attackers cannot attack devices using this protocol. However, until the KRACK vulnerability, this protocol was considered safe.

The KRACK attack doesn't give us the password of the device that provides the wireless connection. This attack gives us a way to weaken the encryption mechanism. It doesn't give us the connection password to the device moreover we do not need to know the password to do this attack. This attack enables us to decrypt the incoming and outcoming packets from a user, and a bidirectional communication in some devices. In some cases, it finds the hash key that provides the verification (message integrity) between, even in an encrypted communication, and allows us to change the integrity and modify the contents of the packets.

The word *cipher* comes from the Italian word "cifra" which means number. As you can see from here, passwords are texts created using numbers, characters, and signs. Passwords are the only texts that we know and keep in mind. It is possible to see different characters in the password, but the password received is not stored as it is. After the password is received, it creates the text that we call the cipher by pushing it through various mathematical operations.

WEP, WPA, WPA2

There are three wide-scale protocols that secure wireless communication technologies. These are WEP, WPA and WPA2.

WEP

- WEP is a protocol used in wireless communications defined in IEEE 802.11b .
- WEP is the first standardized protocol defined in wireless communication.
- WEP aims to communicate as securely as wired networks.
- There are a lot of vulnerabilities in WEP, and it is harder to adjust compared to new methods.
- WEP uses 24-bit IV (initialization vector) and is therefore weak.
- Alongside with the stream cipher RC4, WEP uses 64/128 bit keys with and the master key must be entered manually.

WPA

- WPA was intended to eliminate the weaknesses of WEP.
- WPA is compatible with WEP devices, and there is no need to change hardware.
- Personal and enterprise modes are available.
- Continues to use RC4 encryption, but the key size is 256 bits.
- Each receiver receives its own password through TKIP.
- Enterprise Mode authenticates using 802.1x and EAP.
- Although it is more powerful in enterprise use, it is open to many attack methods.
- It is no longer safe to use.

WPA2

- The standard that comes with the new hardware.
- Personal and enterprise modes are available.

- Replaces RC4 and TKIP methods with AES (block encryption) and CCMP.
- It has more complex key generation methods than WPA.
- Although brute-force attacks against the generated key are possible, it isn't possible to find the key in a trivial / convenient time.

WPA3

- After the vulnerabilities in WPA2, it is time to use a more secure protocol.
- The protocol was announced on 8.01.2018 and discussions began on the draft.
- The protocol is expected to be announced at mid-2018.

Some of its features:

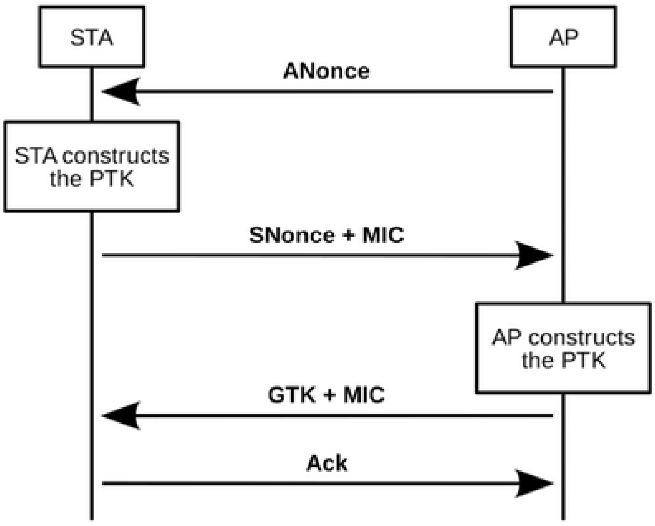
- It will facilitate the connection with non-display devices such as IoT.
- Will provide security even if users choose non-secure passwords.
- Personal encryption will be used to improve wireless internet security in private places.
- This method aims to increase security by providing each person with a different key.
- It plans to make Brute-force attacks harder by disconnecting devices attempting to attack.

The 4-Way Handshake

WPA2 doesn't send the password in an encrypted way from wireless channels, unlike older protocols (doesn't send the password, performs authentication on both sides). Instead it sends a text to verify that both sides have the same information and initiates the encryption process based on the response generated using text.

To explain this scheme, the STA represents us while AP is the device we want to connect to.

Validation is required before encryption. With authentication and encryption, the password to be used for decryption must go through an encrypted channel. The key used to encrypt this encrypted channel will be called PTK. The PTK key consists of the PMK (SHA1 status of the field obtained by wireless password or 802.1x) and some information coming together. (|| is stands for "joining"): PTK = PMK || ApNonce ||STaNonce ||ApMac || STaMac



Nonce values are randomly generated values. The values that start with AP are presented by AP, and the values that start with STA are presented by STA.

1) At first, AP gives us its own generated nonce value and MAC address. Thus, the STA has all the data required to generate the PTK key and generates it.

2) The STA sends its own random value to the AP device. In this way, the AP device generates the PTK key. Now they are ready to communicate.

3) The AP device sends the GTK key to STA device, which will be used to encrypt the remaining communication. This key is sent after encrypting it with the previously known PTK. This way, someone from outside can't decode it. With this step, the STA device receives the parameters required to generate the password and sets the existing values to commence password generation and starts password generation operation (resets values such as default counters during this setting).

4) The STA device returns the response that it received the GTK key. Thus, if the STA cannot return the ACK response or the GTK response is lost on the way, the STA will return the third section again. To catch this package, we can repeat the package as many times as we want.

KRACK

As previously mentioned, the KRACK attack does not need the key that provides the encryption in between. This attack takes advantage of a vulnerability found in the 4-way handshake protocol between the two devices. The internet provider device in the third step of this protocol sends a packet to the user indicating that the handshake occurred. The user who receives this packet completes the installation of the device key and begins to encrypt the data. The internet provider device thinks that this packet may be lost anywhere and sends this packet again. So the user can receive this package more than one time. It resets the IV (initialization vector) and packet counter, which helps in encrypting the key each time it receives the key and X provides the installation of the key again. Thus, it is encrypted with the same key and start vector each time.

So what is this initialization vector (IV) and packet counter? WPA2 protocol provides encryption with one of the block encryption methods: AES. AES encrypts with the counter mode (CTR) of the block encryption. In this encryption method, nonce/IV (initialization vector) is used as input and is combined / concatenated with the counter. Therefore, a different password is generated each time. The packet counter is a 128-bit value and changes with each packet. So, how are AES encryption, CTR mode and block encryption done?

0) Initialization vector and encryption algorithm are determined.

1*) Data is divided into blocks (32 bit).

1.a*) Information such as nonce, counter etc. are starting to be used

2*) The input to be processed for each block is determined by XOR. (If the input is counter, the counter is determined, otherwise the XOR are sorted)

3*) XOR operation is performed on each block.

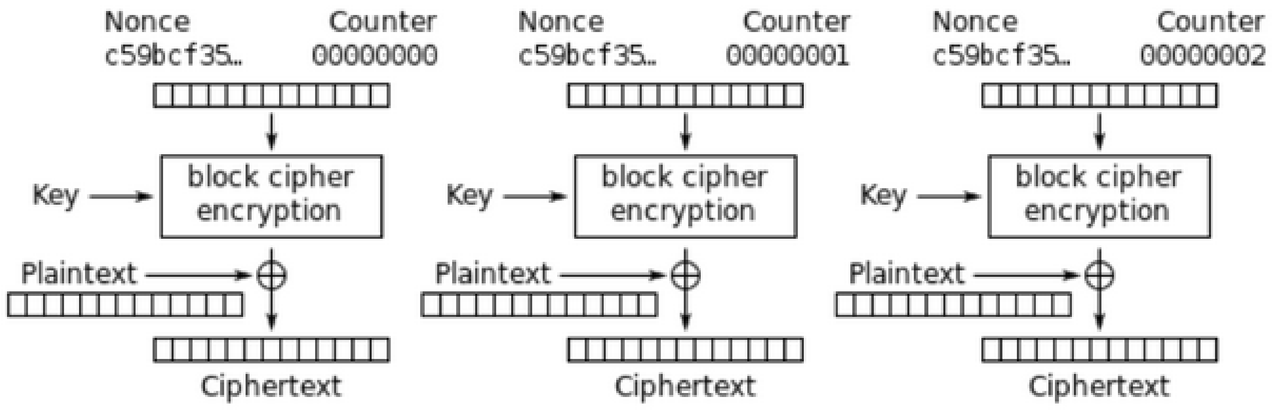
New blocks are sorted, and used for communication. (The operations specified by * belong to the general usage of block encryption, operation zero sets the mode of block encryption).

As you can see, what makes each block special is a nonce value and a counter value. Counter and nonce join together to create part of block encryption. What would happen if we could only generate a password with nonce and key, without counter and apply it to text blocks?

Every time we use the password, we will obtain the same password. For this reason, if we process two encrypted blocks by XOR with each other, we can eliminate the password part (m_1 x p x m_2 x p = m_1 x m_2). Therefore, if we have some part of m_1 or m_2 messages, we can attain the other message. At the same time, if we have one of the messages, it is also possible to obtain the password. So would no longer need the password used in encryption. KRACK attack tries to keep the counter value constant, which is the only changing value to keep the password the same. It can reset the counter value by resending the package in the third step of the 4-way handshake. Thus, it has the same password every time, decoding the text with this password.

Device independently, this attack is valid for all devices using WPA2 since it uses a point defined in the standards. For this reason, please install the security update for your device.

Do not forget to follow the cryptography section in our magazine for the miraculous mathematics mentioned here.



Counter (CTR) mode encryption

Utku Şen • utku@utkusen.com

DANIEL BOHANNON INTERVIEW

1) How old are you, where do you live, what are you doing in your current job?

I am in my late twenties and live in Washington, D.C. I am a Senior Applied Security Researcher on FireEye's Advanced Practices team where I get to hunt the most interesting threat actors all around the world. My goal is to understand specific attackers' TTPs (Tools, Techniques and Procedures) and how to translate this information into numerous layers of host- and network-layer detections. My second "job" is that I am the official barista for my office because I enjoy making good coffee and research shows that well-caffeinated researchers and investigators find evil more effectively.

2) When did you start to study computer security and why did you choose security instead of something else? Could you please explain your journey into computers and security?

My father was a software developer his entire career. As a kid I remember being amazed at all the cool "code stuff" on his computer screen when I would ask what he was working on. Since I enjoyed math and science as a kid, from the age of ten I decided that I wanted to study Computer Science at university and "write code" (whatever that meant). Surprisingly I didn't write my first line of code until university, but from that first Hello World I was hooked. However, I did not discover the world of information security until after university.

My first job after university was working on a database administration team where I fell in love with automating operational tasks with various scripting languages. A year into this job I began looking for a bigger challenge so I began graduate school part-time while working full-time. I wanted to get my masters degree in something IT-related but I wanted a more specific degree than Computer Science, so I chose Information Security. From the first semester I became engrossed in the security world. I began hanging out with the security team at my job and asked at least 100 questions each week about how the subjects I was learning at university translated to security in "the real world." By the time I finished the masters program I officially joined the security team at that job doing operational security.

Three and a half years ago I joined Mandiant, the consulting arm of FireEye, where I was an Incident Response consultant for two years. During that time I became obsessed with developing resilient detections for attacker activity that we found as we responded to breaches. However, I also developed a love for obfuscation and evasion research so I could break (and then fix) my own detections before attackers did. This eventually led me to my current role as a full-time security researcher where I channel these detection ideas into finding the most interesting attackers on the front lines of our active investigations.

3) What are your future plans for the "Invoke-Obfuscation" project? Do you have any new projects that will be released soon?

Oh man, Invoke-Obfuscation has already gotten me in enough "trouble" with my fellow defenders, haha. I still have colleagues sending me payloads that are obfuscated with Invoke-Obfuscation, jokingly telling me "this is your fault" and asking me to decode it for them. I'm not sure that I will be releasing much new content for that project in the near future simply because I believe it proved the point I was trying to make with it. Namely, that string-based detections can be easily evaded with obfuscation and that we as defenders have to approach detecting malicious PowerShell activity in a more thoughtful way.

However, I don't write these obfuscation projects to continually evade detections. If that were my goal then I would release less robust obfuscation frameworks and then keep updating them incrementally every few weeks to keep evading defenders' incremental updates.

Instead, I spend 6-9 months on each project developing automated obfuscation capabilities as deep and crazy as I can go and then release all of it. This is so all defenders can approach the whole problem from the very beginning instead of spending years of "catching up" to incremental changes.

It's funny because most people consider me a red teamer (offensive) instead of a blue teamer (defensive). However, I develop these obfuscation frameworks to be "custom fuzzers" that I then use to develop my detections for these concepts. This is exactly how I used Invoke-Obfuscation and Invoke-CradleCrafter internally before I released them. And when I release Invoke-DOSfuscation earlier this year I also release the Test Harness framework for the project so defenders an easily generate thousands of obfuscated payloads and check their detection ideas against these payloads directly within the Test Harness. So this obfuscation research really is to help me and other defenders increase our detection capabilities, though it is not a simple task.

New projects? Well, I have been working off and on for the past 1.5 years on a very different obfuscation project, but it is one that I probably will not release anytime soon. Recently I have been focused on internal development projects for threat hunting. I have also spent a lot of time developing workshop material so I can teach others about developing effective detections at conferences, workshops, etc.

4) What is your favourite programming language?

I am a PowerShell fanatic :) I have spent time developing in Java, C++, C#, etc. but PowerShell is the most enjoyable language for me in the kind of work that I do. Attackers also love using PowerShell so it's helpful to be comfortable with the language when hunting for malicious usage of it. PowerShell is also a great language to begin with if you do not have any programming experience. I have taught my wife many one-liner PowerShell commands that she uses in her job to save her several hours a week. Programming isn't just for hard-core developers — it can be fun, simple and enjoyable for anybody willing to spend an afternoon learning the basics.

5) What are the hardest parts of the incident response process? What was your hardest incident?

[I decided to skip this question since my other answers were all very long.]

6) You are presenting your research in security conferences all around the world. Isn't it a bit tiring? What is your motivation?

Speaking at conferences has by far been the most fulfilling creative outlet during my career. Periods of heavy travel can certainly become tiring, but I am always mentally recharged with new ideas from conversations with new friends I meet along the way. Speaking at conferences has allowed me to travel to fascinating cities all around the world and meet with researchers, practitioners and security hobbyists from many different backgrounds and walks of life. In addition, it has led me to meet many great friends, many of whom I primarily see in person at various conferences each year.

My motivation initially was simply to "share knowledge" from the conference stage — and sharing ideas that help defenders better protect themselves from offensive techniques is still an important component of my involvement in speaking. But I soon realized that the most valuable knowledge transfers occur off of the stage, and it is usually a mutual exchanging of ideas while talking with other conference attendees. I am an introverted person and typically find it difficult to initiate introducing myself to new people in larger social settings. However, I have no difficulty in talking with people if they introduce themselves. Some of my favorite conversations at conferences aren't even been about security topics, but about cultural intricacies, personal hobbies, etc. So if you are reading this then please consider this my personal invitation to come say hello next time we are at the same place, even if it is just to talk about the weather, to grab a coffee, or to simply exchange names and high-five a fellow introvert.

7) What are your messages to students who want to be like you in future?

My message to students: If you are interested in information security then there is a seemingly infinite amount of information freely available to help you learn. However, knowing where to start can be difficult. In my experience I have found many people in the security community to be friendly and willing to have a conversation with me at a conference, in a Twitter DM, etc. where I can ask them where to start. Worst case scenario they do not reply. Best case scenario we have a great conversation and become good friends. We all learn from each other and are far better together than in isolation.

Another piece of advice is that the security community's body of knowledge is both incredibly broad and deep. Most of the conference talks that I attend are far over my head because they are in areas that I do not encounter in my current role or that I do not have a specific interest in exploring on my own time. This statement is meant to be encouraging. Do not be afraid of not knowing something. Saying "I don't know" is something that I say several times a week, and I appreciate that my teammates are honest and say the same thing. You can find any set of niche areas in the security community to focus on, and if you have no experience in other areas that does not make you better or worse than somebody else who chose to focus on those areas.

Lastly, 95% of good research is simply determination and the willingness to spend a LOT of time focused on understanding and solving a problem, debugging code, developing and redeveloping an idea or a presentation, etc. At the same time it is important to have healthy outlets that are not related to your daily job, so do not neglect the importance of spending time with family, friends and hobbies.

My Twitter handle is @danielhbohannon and my DMs are open if you have any questions or just want to say "Merhaba" (hi). Keep learning, keep helping others, and remember that when it comes to developing yourself in the world of information security, "Damlaya damlaya göl olur" (many a little makes a mickle).

Furkan Senan • furkan@hackerspace.ist

THE ROLE AND COMPARISON OF OPERATING SYSTEMS IN MASS SURVEILLANCE

(QUBES OS, TAILS OS, SUBGRAPH OS)

In the article, you can read the description related to the stars and digits in the references section at the end of the article.

It takes less than 1 minute to break the 12-digit passwords of the" NSA" * 1

"It is on the boil that digging coins to obtain unfair profit over users' computer in Turkey and Egypt and force users to download spyware in the eastern of Turkey by Türk Telekom …" *2

"The Wannacry protagonist faces up to 40 years in prison for malicious software allegedly written years ago in the US…" *3

"The Cambridge Analytica scandal is on the air. The possibility of gathering information from millions of people and the manipulation of elections of many countries, including America, via Facebook is being talked about …"

In addition to these, we can start with some examples such as when you speak loudly about any topic in an environment, you may suddenly start seeing it on Google, or by using its location services Facebook offering the courier to you as a friend when you sit down on your computer right after you saying goodbye to the courier who dropped your package..

Dear readers, finally and slowly we indeed entered into the dystopia which we have watched in animations and have read in mangas. Welcome to the future of monitoring all of our digital prints and saving them for further use which both are planned to be controlled by the United States and giant corporations. In particular the United States who holds the advanced technologies, establishes a policy in which everyone is seen as a potential terrorist and that the good ones are being searched for!

(Artwork Josan Gonzalez)

I do not have anything hidden at all!

If are among those who say "Dude, why should I cover my camera or encrypt my disks? I do not have anything hidden thing to be looked at!" Well, sadly I must admit that you are not with us and you will not use any of following information of this article, since it is going to be a fictitious story that will occupy some space your brain. However, I advise you to take a look, and you may use them in any incoming conversation related to it.

> " Y'KNOW WHAT? ALL THESE GOVERNMENTS ARE ABLE TO HACK WHATEVER COMPUTER OR PHONE THEY WANT WITH JUST A CLICK, I READ IT ON FACEBOOK!"

So why it is important for us? Encrypting your data, surfing with an anonymous identity on the internet does not necessarily mean that you are involved in a monkey business.

Joanna Rutkowska, the founder and architect of one of the operating systems (Qubes OS) that I will explain in a moment, says;

"We (people) are actually moving our lives to those personal devices. These are becoming extensions of our

brains. If someone spies on your personality, your thoughts and private life (that's why we call it private) your individuality is in danger."

In a nutshell, this is a matter where your personality is in danger. That is why, in order to be protected from mass surveillance, we first set up anti-vi… No, of course, we won't even be talking about anti-viruses.

In this article we will deal with the structure you face with when you turn on any device, the "Operating system". We are going to dig up a variety of systems and try to find the safest system that's right for you. Because of the page and time constraints, we will only go deep for computers, and of course there is the mobile device side. We later may examine different systems in other articles.

First of all, there is a legal spyware, an operating system used by most personal users that we must get rid of it in some way. "Wind*ws!"

With great power comes great responsibility. - Uncle Ben

With extra security comes great toll. – Furkan (It's me)

If you want relatively good and safe devices you will have to give up some of the habits that make your life easier. You will feel more free, safer and more exclusive with every habit you get rid of in the digital environment!

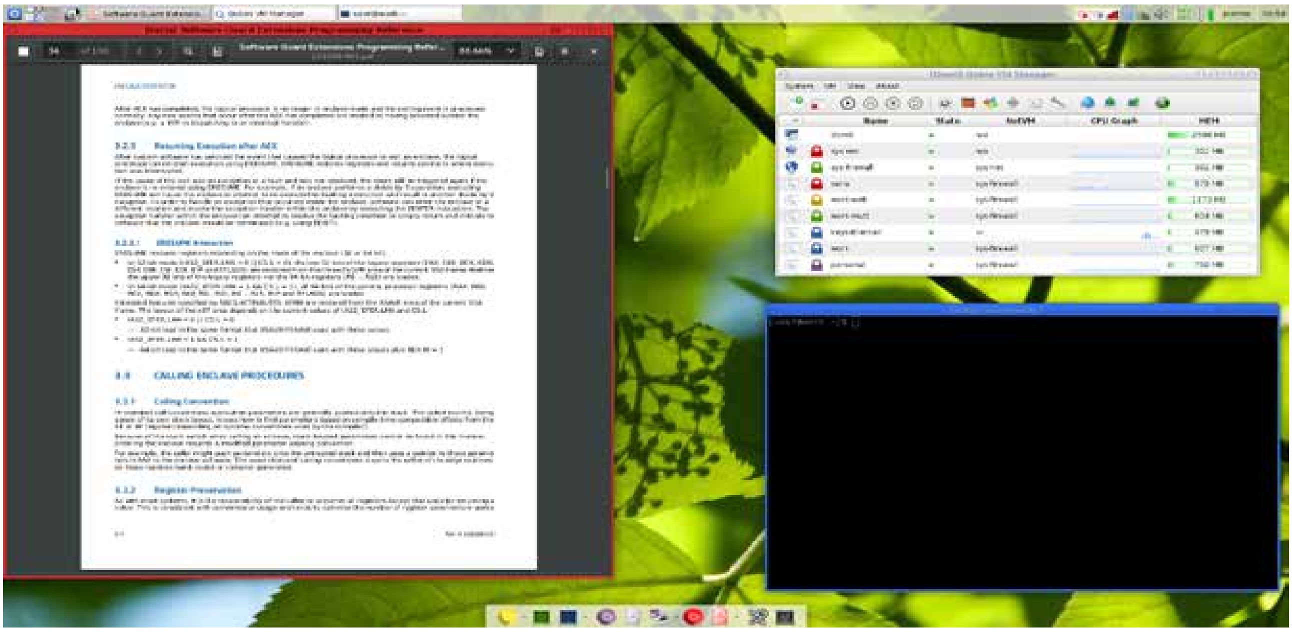
Now it's time to dive into more technical parts. In this article, there will be three different operating systems that prioritize user security.

Qubes OS, Tails OS and Subgraph OS.

We will see the differences, the pros and the cons and decide accordingly.

Qubes OS

Forget all the operating systems and software you know. Because Qubes OS offers an unusual structure with its slogans "Reasonably Safe Operating System".

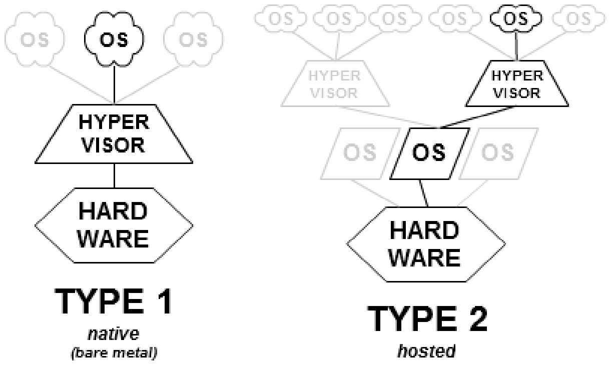


(Sample Qubes OS environment)

While everyone was concerned about developing a secure operating system, the Qubes OS team made a radical decision, saying, “Let’s develop a platform and use all the operating systems that are safe”. In addition, they carefully stated that they do not trust the Linux Kernel and they have built the Qubes OS over the “Hypervisor” *5 technology.

What is Hypervisor? Let’s describe briefly;

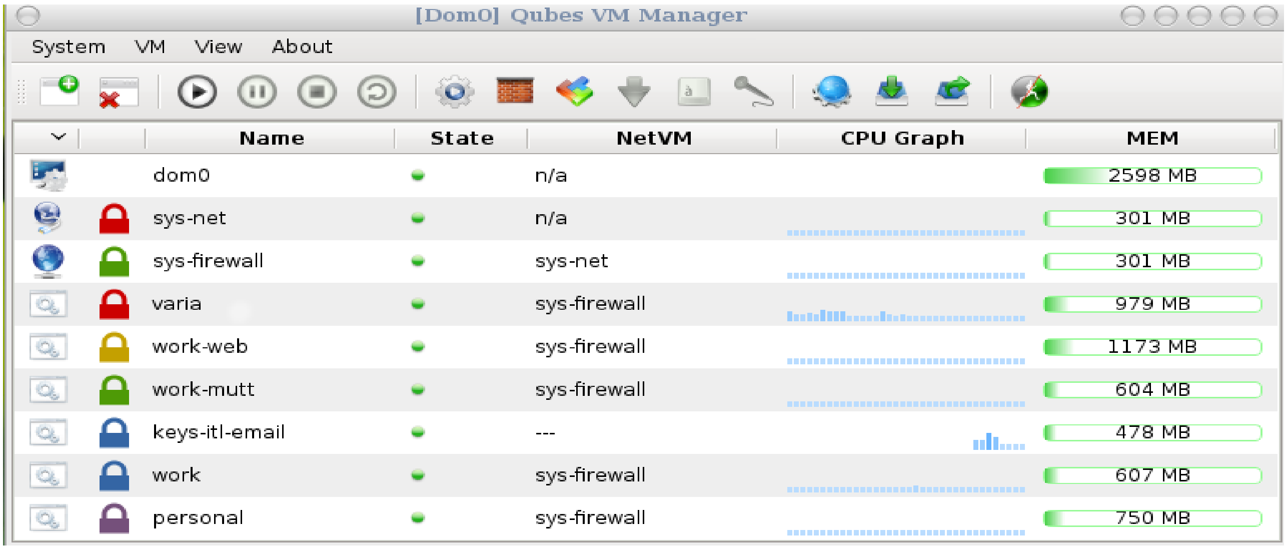
It is the software that enables the installation of more than one operating system by virtualizing the hardware of a physical computer.



We can explain aside from technical information. I assume that you have watched the movie Inception, consider another operating system running inside the operating system. Maybe even another operating system that runs in it? And even isolate these systems from one another. Yes! That’s exactly what you’re imagining right now, Qubes OS.

Hypervisor can be installed in two ways. “Native” means directly on the hardware.

Or “Hosted”, which means that first an operating system is installed, then the hypervisor can be installed, and then another operating system can be installed into it.



Name	State	NetVM	CPU Graph	MEM
dom0	●	n/a		2598 MB
sys-net	●	n/a		301 MB
sys-firewall	●	sys-net		301 MB
varia	●	sys-firewall		979 MB
work-web	●	sys-firewall		1173 MB
work-mutt	●	sys-firewall		604 MB
keys-itl-email	●	---		478 MB
work	●	sys-firewall		607 MB
personal	●	sys-firewall		750 MB

(Qubes OS Virtual Machine Manager)

First of all, it requires you to encrypt your disks in the installation. Then you can start with the 6 pre-installed machines depending on your choice. By default, these are set as Fedora, Debian, Whonix, but if you insist on saying “I’ll install wind*ws!” you can install it by opening a separate virtual machine.

They also made the template of the default systems. In about 20-25 seconds, you can set up a brand-new clone system.

If we explain the above screenshot;

Each color represents a separately running operating system, and the only connection between each other is **“Dom0”**, your host.

From now on, each virtual operating system will be referred to as ***"machine"***

Sys-net- is the only machine has internet port.

Sys-Firewall – accesses internet over Sys-net machine. All other virtual machines are connected to the Internet via Sys-Firewall as NAT, and the network packets are parsed according to the rules.

The rest of the machines were created by the user.

Personal - In this virtual machine, personal mails can be viewed, internet can be surfed and so on. No action has any effect on other systems e.g. Say that you entered duckduckgo.com while browsing in Firefox. The history does not appear in Firefox on the "**work**" machine. There is no connection between them.

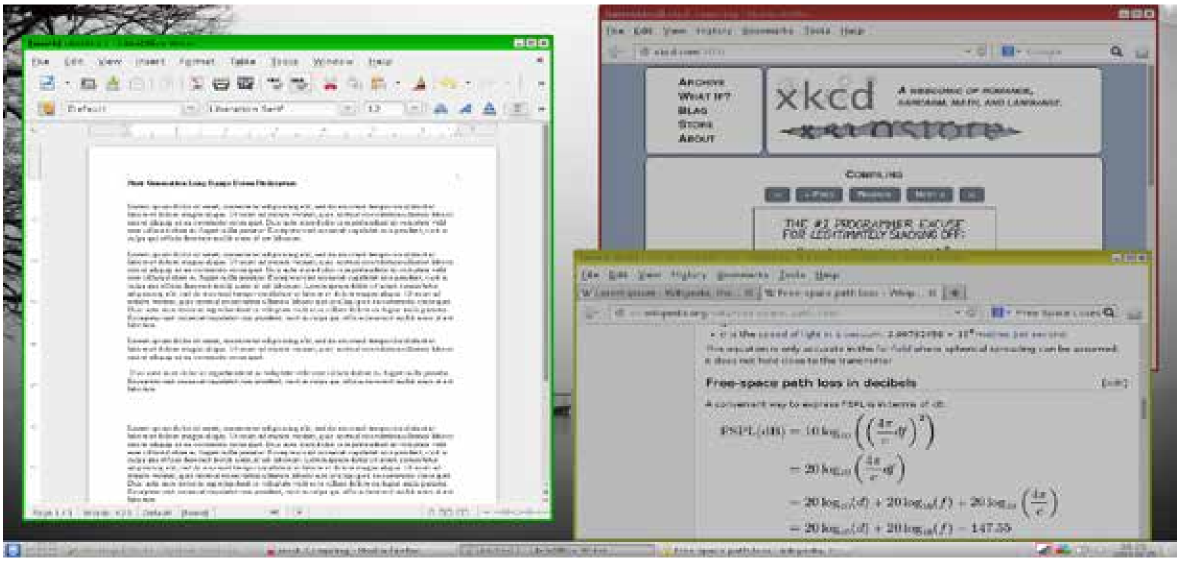
In the same way, you accidentally ran a malware over e-mail on your **Varia**. What happens only happens in the Varia machine. No other machine is affected. (Unless you get Dom0 infected)

The right-side panel shows the RAM usage in the "MEM" column. As you can understand, if you want to use Qubes OS in a performance manner, you need a computer with at least 8 GB RAM. (For example, the Thinkpad x220T + 8 GB RAM is quite well.)

Windows (Just a synonym!)

"Oh! How do we press the start button of the 4th machine?"

It's not how you think it is. There are no hundreds of windows, nor dozens of desktop environments. The only thing you will see is Dom0's desktop screen of your host.



(Sample desktop environment)

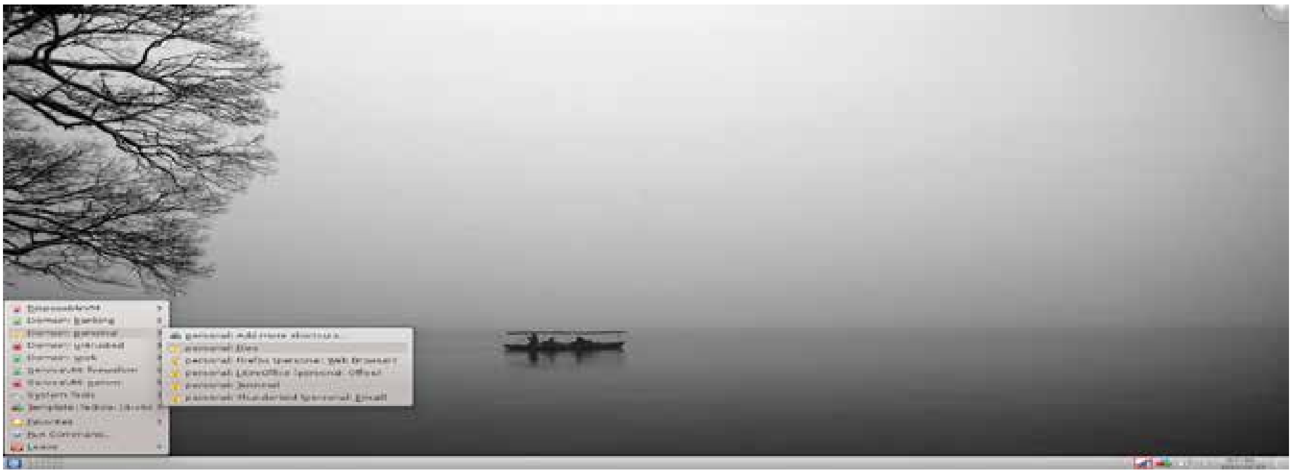
The rest will be the application windows you will run on your preferred machines. If we look at the example desktop environment above;

Green window indicates the LibreOffice Writer application run by **work** machine.

Red window indicates the Mozilla Firefox application run by **Untrusted** machine.

Yellow window indicates the Mozilla Firefox application run by **work-web** machine.

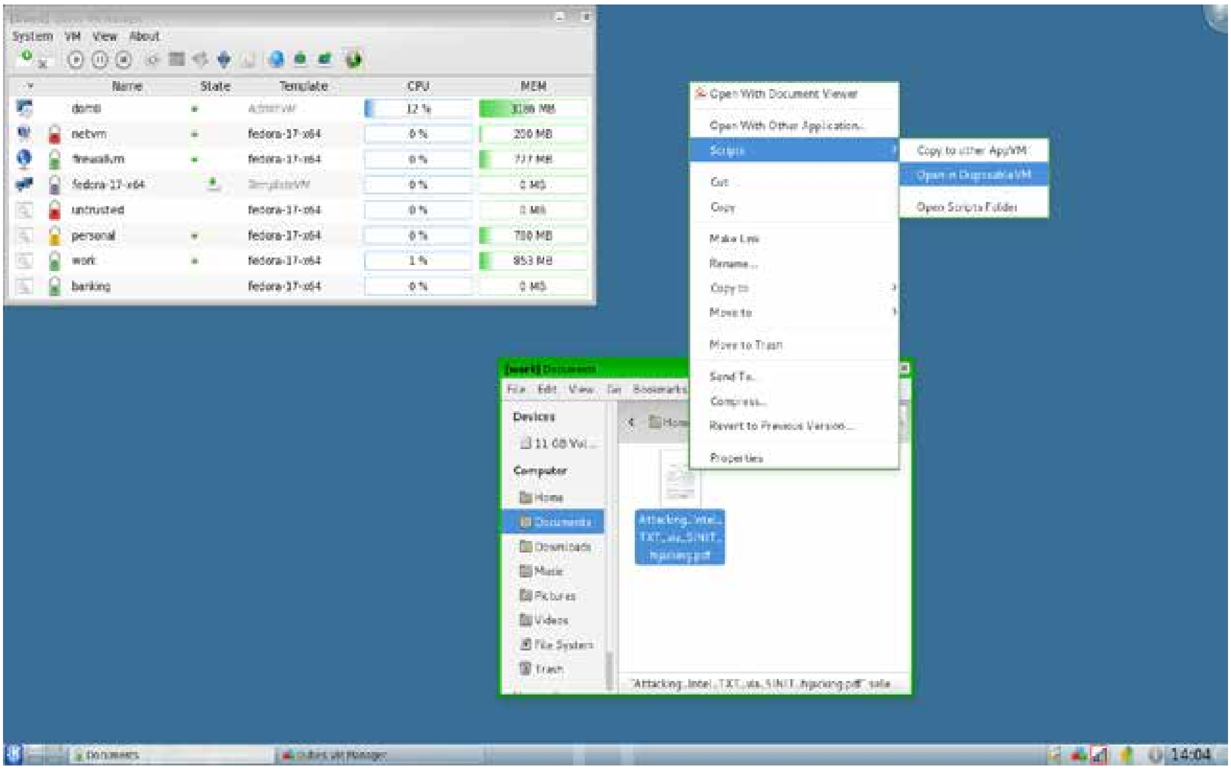
As you can see, you can easily call the application you want from within the virtual operating system and see it on the same screen.


Domain: banking
Domain: personal
Domain: untrusted
Domain: work
System Tools
Favorites
Run Command...
Leave
personal: Add more shortcuts...
personal: Files
personal: Firefox (personal: Web Browser)
personal: LibreOffice (personal: Office)
personal: Terminal
personal: Thunderbird (personal: Email)

(Launching an application from start menu)

Disposable VM

Imagine there is an email that you do not know where it came from. It contains a file that interests you but you also suspect that it is a malware. In this case, you can leave the whole job to the **Disposable VM** of Qubes OS.

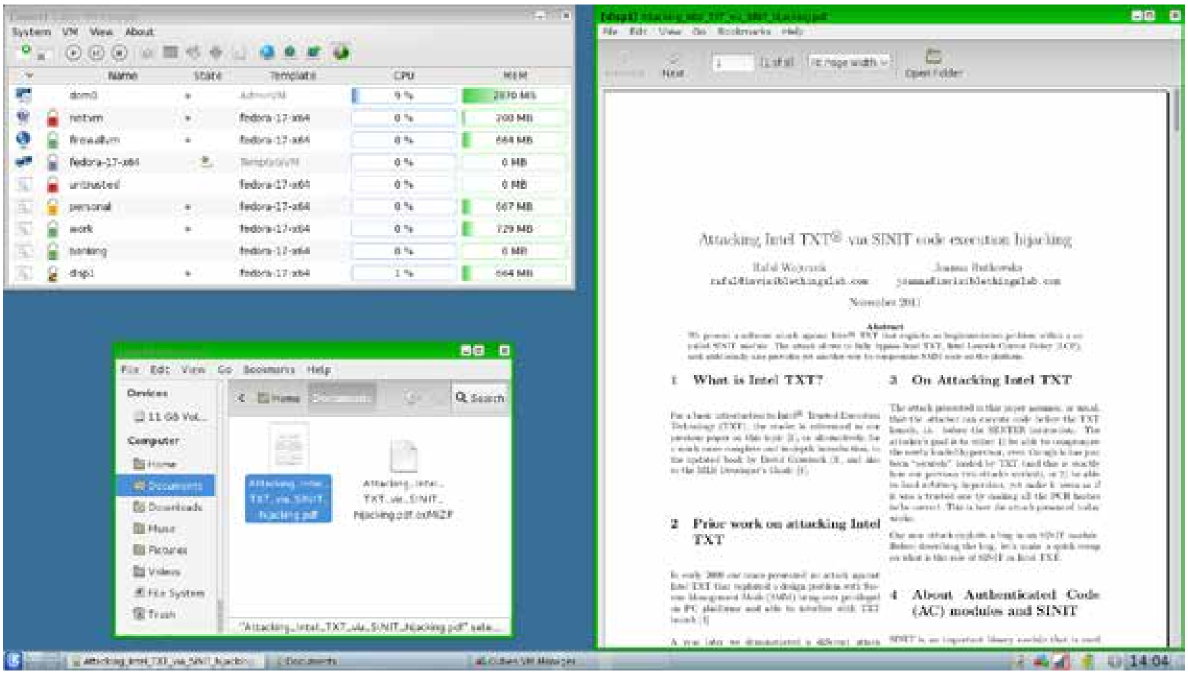


Name	State	Template	CPU	MEM
dom0	●	AdminVM	12 %	3086 MB
netvm	●	fedora-17-x64	0 %	200 MB
firewallvm	●	fedora-17-x64	0 %	727 MB
fedora-17-x64		TemplateVM	0 %	0 MB
untrusted		fedora-17-x64	0 %	0 MB
personal	●	fedora-17-x64	0 %	700 MB
work	●	fedora-17-x64	1 %	853 MB
banking		fedora-17-x64	0 %	0 MB

"Attacking_Intel_TXT_via_SINIT_hijacking.pdf" sele

(Utilization of Disposable VM)

When you right-click on the file you downloaded and click Open in Disposable VM, the system will then install you a virtual machine and open the application on it, and if it has malware, it will survive only up to the lifetime of the Disposable VM. I mean, until you shut down that machine.



Name	State	Template	CPU	MEM
dom0	•	AdminVM	9 %	2870 MB
netvm	•	fedora-17-x64	0 %	200 MB
firewallvm	•	fedora-17-x64	0 %	664 MB
fedora-17-x64		TemplateVM	0 %	0 MB
untrusted		fedora-17-x64	0 %	0 MB
personal	•	fedora-17-x64	0 %	667 MB
work	•	fedora-17-x64	0 %	729 MB
banking		fedora-17-x64	0 %	0 MB
disp1	•	fedora-17-x64	1 %	664 MB

Attacking Intel TXT® via SINIT code execution hijacking

Rafal Wojtczuk
rafal@invisiblethingslab.com

Joanna Rutkowska
joanna@invisiblethingslab.com

November 2011

Abstract

1 What is Intel TXT?

2 Prior work on attacking Intel TXT

3 On Attacking Intel TXT

4 About Authenticated Code (AC) modules and SINIT

After the simple explanation of its usage, let's talk about the differences, pros and cons of Qubes OS.

Video Walkthrough

You can access the official video presentation of Qubes OS by scanning the QR code above.

Differences

First of all, we can easily say that Joanna Rutkowska and her team are aware that 100% security is out of question, and this is enough to sympathize them. The sentence below she tells in each of her speeches reveals that these people are aware of what they are really doing.

“If the hardware you are using is full of vulnerabilities, the operating system you use does not have meaning.”

At this stage, I think it would be more appropriate to mention the questions asked to and answered by the Qubes team. *6

Question: Why bother with Qubes OS, if any Linux/BSD already allows to setup different user accounts, or some form of light-weight containers or sandboxes, such as chroot, LXC, SELinux?

Answer: - First, if you use Xorg or similar X-based server as your GUI server, and this is what nearly all Linux, and most of the other non-Windows OSes use, then you don’t have any form of GUI-level isolation, which is essential for a desktop system. I wrote more about this surprising problem some time ago. Proper GUI-level isolation was one of the main goals for Qubes.

Second, all mainstream desktop OSes, such as Windows, Linux, BSD, even OSX, are all based on a monolithic kernel, which present a significant security problem. This is because a typical monolithic kernel of a contemporary desktop OS contains tens of millions of lines of code, and to make it worse, most of this code is reachable from (untrusted) applications via all sorts of APIs, making the attack surface on the kernel huge. And it requires just one successful kernel exploit to own the whole system, bypassing any security mechanism that might have been built on top of it, such as SELinux, LXC, etc.

[^Monolithic core is a single file operating system core. (WIKIPEDIA)]:

Additionally, all the various drivers, networking and USB stacks, are also hosted in the kernel, making attacks via buggy networking (e.g. via buggy 802.11 stacksor buggy firmware) or USB stacks a practical possibility. And there is essentially nothing one can do about it, when using an OS based on a monolithic kernel.

In Qubes, on the other hand, we use Xen hypervisor to provide security isolation between domains, and Xen is just a few hundreds of thousands of lines of code. It also doesn’t need to provide all sorts of APIs to applications, because the Xen hypervisor is essentially only interested in CPU scheduling, memory management and power management, and very few things beyond that. Most notably, the Xen hypervisor knows nothing about networking, disk storage, filesystems, USB stacks, etc, as all those tasks are delegated to (often untrusted) service VMs.

Question: How is Qubes better than just running a bunch of VMs on an ordinary OS?

Answer: First, products such as VMWare Workstation, Fusion and Virtualbox, are all classified as the second type, that is to say as hosted hypervisors. So, since they run on an ordinary OS your machine is again being controlled by a monolithic kernel. This means that they use the OS-provided services for all sorts of things, from networking, USB stacks, to graphics output and keyboard and mouse input, which in turn implies they are only as secure as the hosting OS. If the hosting OS got compromised, then it is a game over, also for all your VMs.

Pros

- Installation is possible. (Stable and persistent)
- A platform on its own, not a continuation of an existing distribution.
- It consists of different systems that are insulated and provide a comfort regarding security.
- If you plug in a bad USB, take a kernel damage, or have an indirect attack from any drive, your isolated systems remain stable and secure.
- Can be combined with other operating systems and their unique features (eg Subgraph OS’s anti-exploitation architecture)

Cons

- Not achieving the same stabilization in every hardware.
- Requires powerful hardware features (e.g. 8GB RAM).
- Difficult to use (even if you copy a file from one VM to another, it gets a little hassle until you catch a certain habit.)
- If you do not run regularly, you may be lost within the virtual machines (for example, if you may not remember which machine you have saved, like I did)

Analysis of potential attacks

I hope we are all aware that there will be no 100% safe operating system, especially in modern and complex architectures. The purpose of Qubes's architecture is to minimize the number of attacks and to control the main components of the system, rather than checking each application running in the system.

At this stage, I am going to explain the Potential Attack Analysis done by the Qubes OS team.

The example scenario assumes that the attacker seizes one of the Virtual machines and wants to jump over to the others.

1 Phase and 2 Phase Attacks

We can divide Qube OS potential attacks into two groups.

- 1 Phase attacks are based on an attacker finding a vulnerability in the system and exploiting it.
- 2 (or more) Phase attacks are based on the exploitation of more than one weakness in two different places in system by bounding these weaknesses.

Potential 1 phase attacks for any virtual machine

- May be due to potential errors in Hypervisor
  - Since the most authoritative element in the system is hypervisor, any attack on this system will cause the system to be completely captured.
- A potential error in Xen Store Daemon (Dom0)
- A potential error in GUI Daemon (Dom0)
  - A successful exploitation will be harmful because GUI is run within Dom.
- Potential Processor Errors
  - It is important to note that any errors to be found in the CPU will be very troublesome not only for Qubes OS but for any operating system. (See. Specter and Meltdown weaknesses. Detailed information can be obtained from the articles written by Chris Stephenson in the 1st and 2nd issues of Arka Kapı Magazine. Editor)

You can browse the reference link * 7 for more detailed potential attack scenario analysis. **(Page 41)**

> *"Please do not bother if you use the "secure" operating systems just because you trust us, think that we look like good people and are not likely to place a backdoor. Backdoors are already present in the hardware you use. Here is the illusion of security."*-Joanna Rutkowska

Tails OS – The Amnesic Incognito Live System

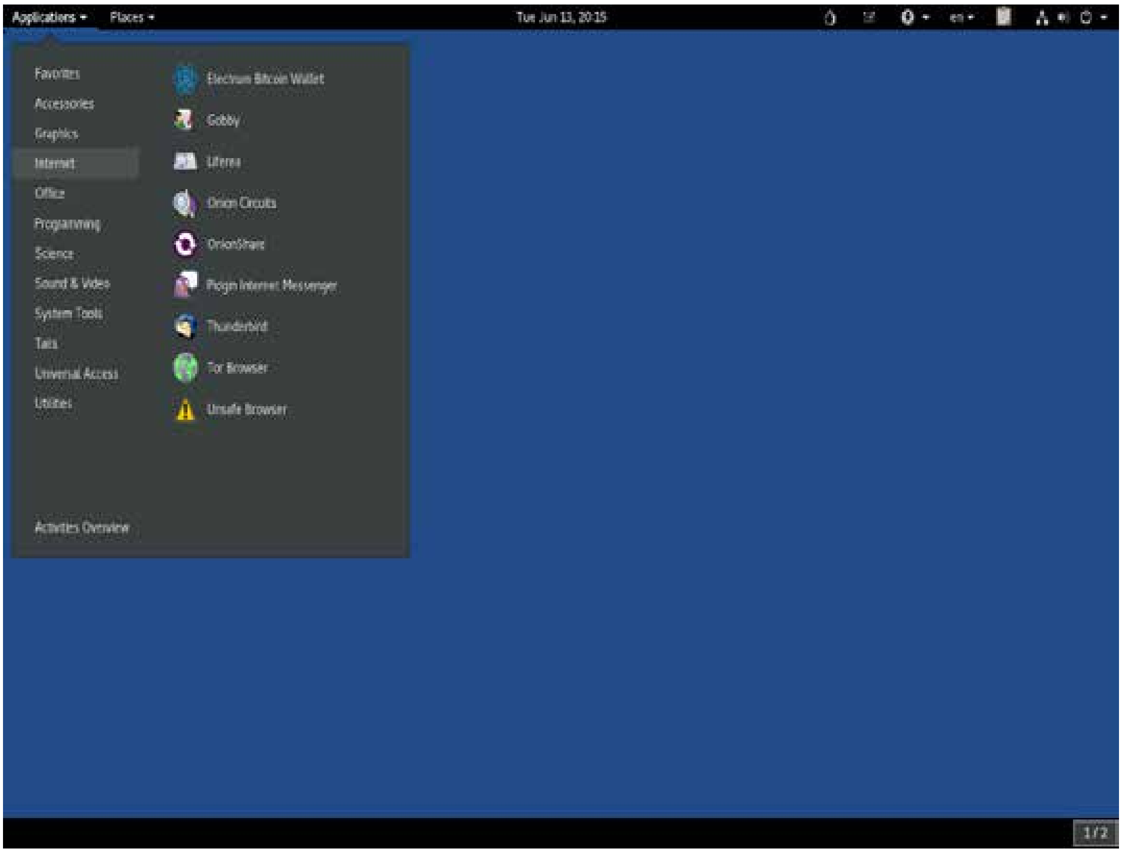
amnesia, *noun*: a medical condition that makes you unable to remember things, long-term memory loss.

incognito, *adjective & adverb:* avoiding being recognized, by changing your name or appearance.

The **A**mnesic **I**ncognito **L**ive **S**ystem

Tail OS, driven by the motto "Privacy for everyone everywhere", is a live-executable operating system. If you ask "what does this live mean?" I can simply answer that it can be run fully on RAM without having to need installing it on computer. So, when you save a file, if you do not back it up on your USB flash drive, you will lose your file.

Will the file be the only thing lost? No. As soon as you click the shutdown button, the entire operating system will be vanished for good with the very same button.



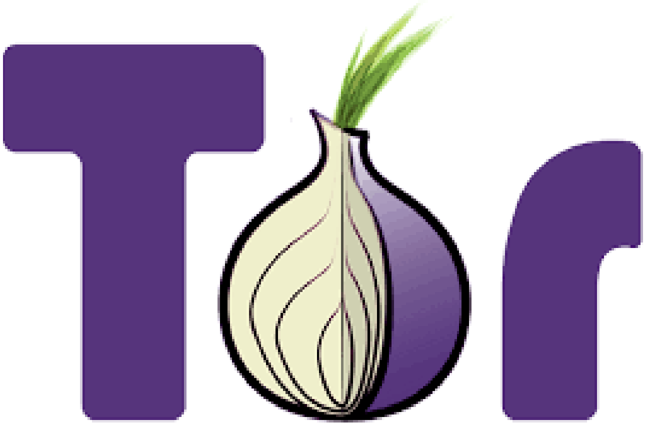
In fact, Tails, a basically customized Debian distribution, comes with some configuration settings and applications, and it's completely free *8 software.

TOR

If you are planning to use Tails OS, you will have to deal with the TOR network. Because all applications used on Tails are connected to the Internet via TOR. Even if there is an application that tries to go connect to the internet with something instead of TOR, it's connection is automatically blocked with security reasons.

Since we are not going to talk about it deeply, let's briefly explain the TOR network:

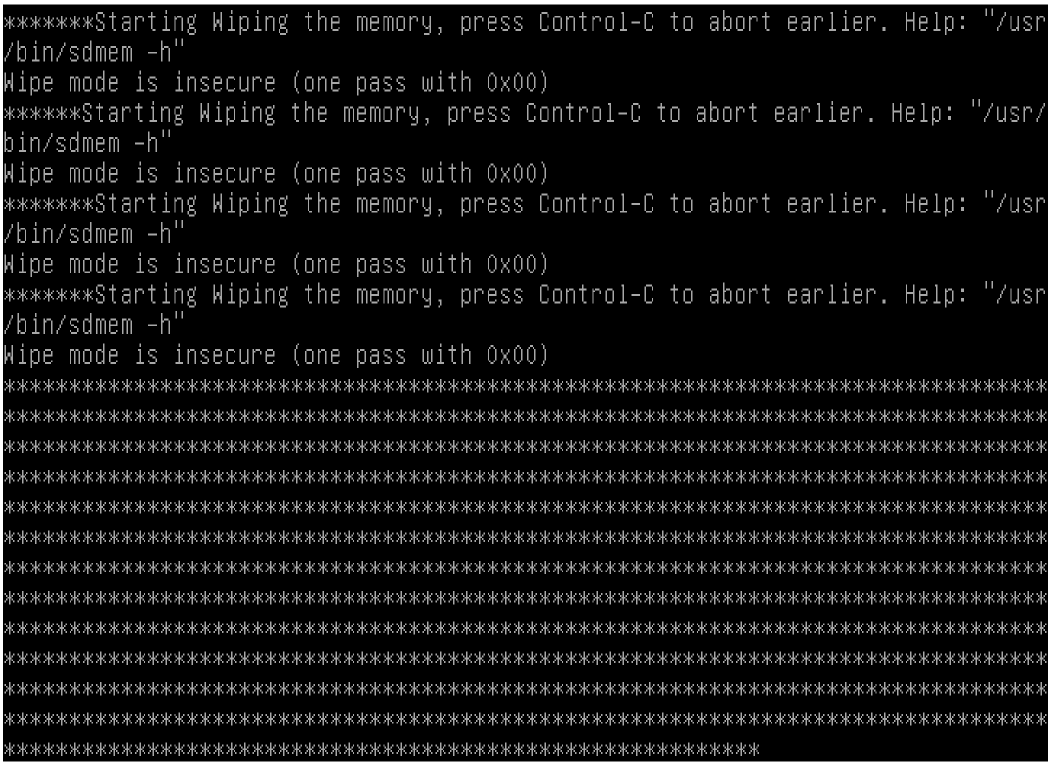
The Tor network, which uses an acronym derived from the words **"The Onion Router"**, is a free network project that provides its users with anonymity on specific topics on the internet by routing their traffic in an encrypted way through network nodes (TOR Relay) created by volunteers. According to the data of Wikipedia 2018, there are currently over 7000 voluntary network nodes in the world.



(To get the answers to what Tor is, what it is not and how to install it, please refer to item 9 in the references section or see the 2nd issue of the Arka Kapı Magazine Turkish.)

Use Everywhere But Do not Leave A Trace

(Tails fills RAM with 0's (zeros) when shutting down to prevent data from remaining.



```
*******Starting Wiping the memory, press Control-C to abort earlier. Help: "/usr
/bin/sdmem -h"
Wipe mode is insecure (one pass with 0x00)
******Starting Wiping the memory, press Control-C to abort earlier. Help: "/usr/
bin/sdmem -h"
Wipe mode is insecure (one pass with 0x00)
*******Starting Wiping the memory, press Control-C to abort earlier. Help: "/usr
/bin/sdmem -h"
Wipe mode is insecure (one pass with 0x00)
*******Starting Wiping the memory, press Control-C to abort earlier. Help: "/usr
/bin/sdmem -h"
Wipe mode is insecure (one pass with 0x00)
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
********************************************************************************
**********************************************************
```

We mentioned that one of the most important features of Tails is the live work. That's why you can connect to the internet and organize your files by running Tails via any computer with a USB stick that you carry in your pocket, or by hanging it down a necklace or wristband.

Imagine a scenario where you don't have your own layout. You are traveling or you must work on a workplace computer, but you need to read some files or emails in which you need to pay attention to protecting your privacy. You don't have a secure computer. Tails takes place in here.

You insert a USB stick and just as if you were formatting the computer (booting the computer from a USB memory stick or a Tails-connected media device). When you initiate Tails OS live on the computer, the main operating system behind does not work in any way. So, when you finish your work, everything will be cleaned as if you never touched that computer.

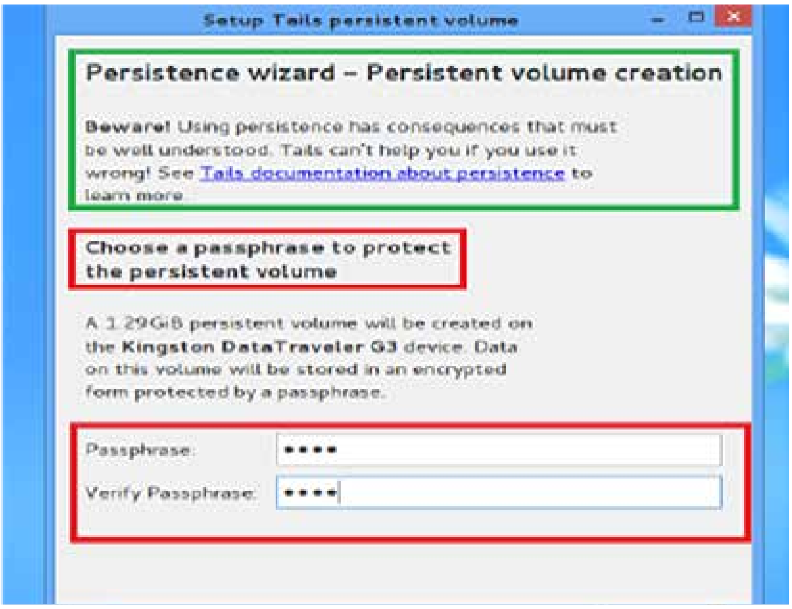
Persistency

Will we set up all the files and settings with each boot again and again?

> Of course not. Tails is a system that is produced entirely for live operation, however if you want to store your files, read and edit them in the next session, and if you do not want to reinstall and set up a program every time you launch the system, you can have partition your USB memory stick and place the files and configuration settings in there.

Tails "Encrypted Persistence USB Disk"

To do this, you need a USB memory of at least 8 GB.



Persistence wizard – Persistent volume creation

Beware! Using persistence has consequences that must be well understood. Tails can't help you if you use it wrong! See Tails documentation about persistence to learn more.

Choose a passphrase to protect the persistent volume

A 1.29GiB persistent volume will be created on the **Kingston DataTraveler G3** device. Data on this volume will be stored in an encrypted form protected by a passphrase.

Passphrase: ••••

Verify Passphrase: ••••

Storing sensitive data

The Persistent module on your USB disk is not hidden. You may be forced to give your password with an attack or you may be exposed to phishing attacks.

Overwriting configurations

Programs in Tails are meticulously configured as a result of security concerns. An incorrect setting can change the settings of Tails and you may have problems with security.

Installing an application

The chosen Tails applications had been examined to address generally everyone while paying attention to security concerns. Installing programs that you do not know the source of and contents may damage the security of Tails.

List of all programs in Tails *10

Initiating the persistence module in another operating system

This is a possible option, but doing so means ignoring the security Tails provides for you. Please be careful.

Video Walkthrough

You can watch Tails OS login and promotional video by scanning the QR code above.

Pros

- No trace is left after the system is shut down.
- Core network structure uses TOR completely
    - This is also a disadvantage. Will be discussed in the Potential Attack analysis.
- Fully portable.
- Incredibly useful in case of any instant scenarios/circumstances.
- Can work on any hardware.
- Provides protection against "Cold Boot" attacks thanks to RAM cleaning feature.
    - When you turn off your computer, a certain amount of data can be recovered from the RAM by the electricity that is still in the device within a certain period of time. Physical access is required for this, but in any case, Tails protects you against it.

Cons

- In-direct funding by the USA
    - This is one of the situations that confuses minds and questions the reliability of Tails. They receive financial support from a state-funded technology association. Another source of income is the donations. (see https://youtu.be/Nol8kKoB-co?t=242)
- Monolithic kernel is used, a weakness in the system can disable the whole system.
- Modified version of an existing distribution.
    - I want to consider this as a disadvantageous point because most vulnerabilities in Debian affect Tails.
- Using TOR as the default network
    - We cannot consider TOR as a 100% secure system. It provides anonymity only up to a certain level.
- No regular persistence.
    - Persistence is only provided with a specific USB memory module, so it is a problem to install Tails and continue for months in the same order.

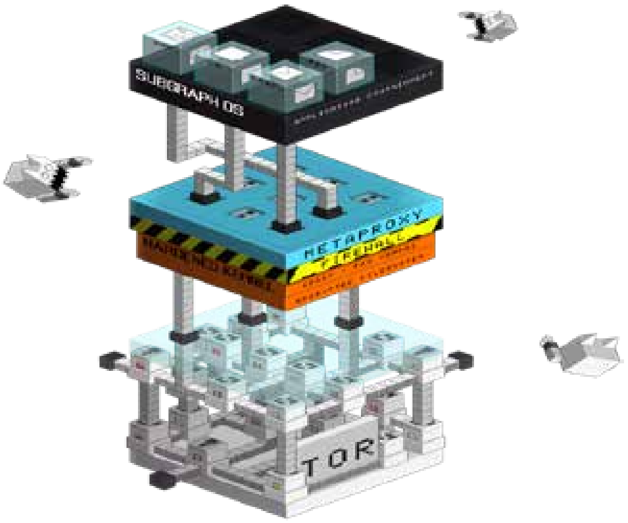
Potential Attack Vectors

- Tails cannot protect you from physical or hardware attacks.
    - If someone accesses your computer physically and installs malicious software, Tails can't help.
- Installation is required when Tails is in a secure system.
    - Tails OS which you install on your USB disk in an untrusted system may undergo malicious corruption or change.
- Tails cannot defend you against BIOS or Firmware attacks.
    - see: BIOS necromancy *11
- TOR exit nodes can secretly listen to you.
    - This is a TOR network vulnerability. Connecting from the TOR network is advantageous in terms of anonymity and disadvantageous because of the possibility of you being listened by an unknown exit node.

- Tails can reveal your usage of TOR and / or Tails.
  - Tails cannot show you any way as if you were a random internet user.
- Tails cannot protect you against Man-in-the-Middle attacks.
  - You can face an MiTM attack between the Tor exit node and the destination server. *12
- Tails does not encrypt your files by default
  - You have to do this yourself.
- Tails does not delete the metadata of your documents, does not encrypt the subject and heading of your sent mails.
- Tails does not magically make your weak password strong.
- Since it is a Debian distribution, it is possible that your device will be compromised during the session you use, but this will be reset each time you log off.

Tails is still being developed.

Subgraph OS



(Subgraph OS Layer Representation)

The construction of Subgraph OS was originally inspired by Tails OS. A group (or a company?) called Subgraph were dreaming, saying “Tails is nice, but we wish it could be installed, and was more secure during the session”, therefore starting to make Subgraph OS, hence successfully bringing the project up to alpha version.

The developers of Subgraph OS have long been in the security industry. Besides, “hackers” can recognize themselves from the “Vega” tool in Kali Linux. Because the Subgraph team also develops tools like “Vega” and “Orchid”.

Basically, Subgraph OS is designed for people to share and collaborate on the Internet without having the fear of surveillance and intervention. In its design, it has been kept important that people would be able to do their daily work easily and safely in a unique environment.

It uses GNOME for desktop environment and Debian GNU/Linux distribution as Tails.

Some changes of course occur in its core;

- Anonymizing Internet traffic through TOR network. (Not as strict as Tails.)

- Additional security measures
- Runs most applications in a safe environment called “sandbox” and aims to limit risks in case of any attack.

Security and privacy means?

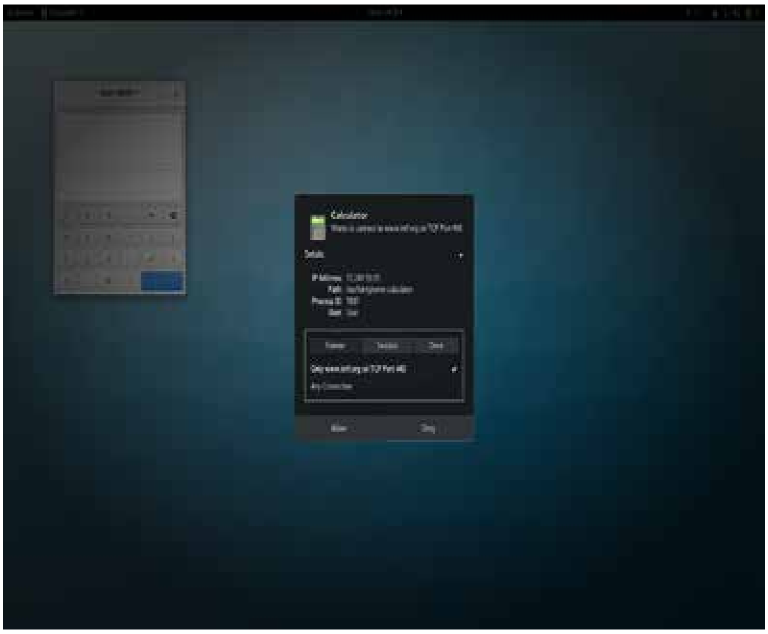
Subgraph is basically based on three keywords when talking about security and privacy:

Privacy: Ensures that the information is not transferred to any person or organization other than the owner.

Integrity: Ensures that information has not been altered and cannot be altered by an unauthorized person.

Utility: Ensures safe access to information.

Anti-Exploitation



“Systems claimed to be safe must be prepared for everything.”

Subgraph OS has an anti-exploitation module in itself. This module is also divided into different parts. The weakness of any application or encryption algorithm installed in the system may have arose. This is divided into two by theoretically and practically possible attacks.

First, a part of this module, which is called the "**Harvester**", collects the exploits which has attacked the system but failed. So, you can analyze the attacks on your system.

Its kernel is protected by **PaX of Grsecurity.**

Container insulation is prepared by isolating risky applications in the sandbox.

Requires **mandatory** file system encryption.

Metaproxy layer step in against memory corruption.

Pros

- A permanent system
- Basic logic is based on isolation like Qubes OS, but here only applications are isolated from each other.
- Harvester module is very well designed for analysis.

Cons

- It is a development over an existing system.
- Still in alpha stage.

David Mirza - (Subgraph OS Team Leader) Interview

Conclusion

I've tried to write an article which appeals to everyone.

In this article, we only talked about operating systems, but being anonymous is of course not limited to this. Your identity can be discovered by the hardware you use, your behavior on the internet, the reliability of the applications you use, your mobile devices, even the emojis your use in messages. A lot of factors are involved.

I hope that we will be completely free of authority oversight with the following articles.

I would like to express my gratitude to Arka Kapı Magazine and its editors for giving me the opportunity to write this article.

For All Questions & Reviews & Suggestions:

You can contact me at **furkan@hackerspace.ist**.

To understand the article more deeply, you can watch a video of a panel where leaders of Tails, Subgraph and

Qubes developer teams participated, with reference *13.

Until next time..!

Furkan Senan

References:

*1 https://www.imdb.com/title/tt4044364/

*2 https://thehackernews.com/2018/03/cryptocurrency-spyware-malware.html

*3 http://www.dailymail.co.uk/news/article-4760140/British-hero-23-faces-40-years-jail.html

*4 https://en.wikipedia.org/wiki/Facebook%E2%80%93Cambridge_Analytica_data_scandal

*5 http://www.wikiwand.com/en/Hypervisor

*6 https://blog.invisiblethings.org/2012/09/12/how-is-qubes-os-different-from.html

*7 https://www.qubes-os.org/attachment/wiki/QubesArchitecture/arch-spec-0.3.pdf

*8 https://www.gnu.org/philosophy/free-sw.html

*9 https://git.hackerspace.ist/Whiterabbit/Ozgurlesin_Articles/src/master/Tor/Tor_Agi.md

*10 https://tails.boum.org/doc/about/features/index.en.html

*11 http://www.legbacore.com/Research.html

*12 https://web.archive.org/web/20120113162841/http://www.teamfurry.com/wordpress/2007/11/20/or-exit-node-doing-mitm-attacks

*13 https://www.youtube.com/watch?v=Nol8kKoB-co

Note: Wind*ws = Microsoft Windows, we prefer not to mention its name, especially in such articles.

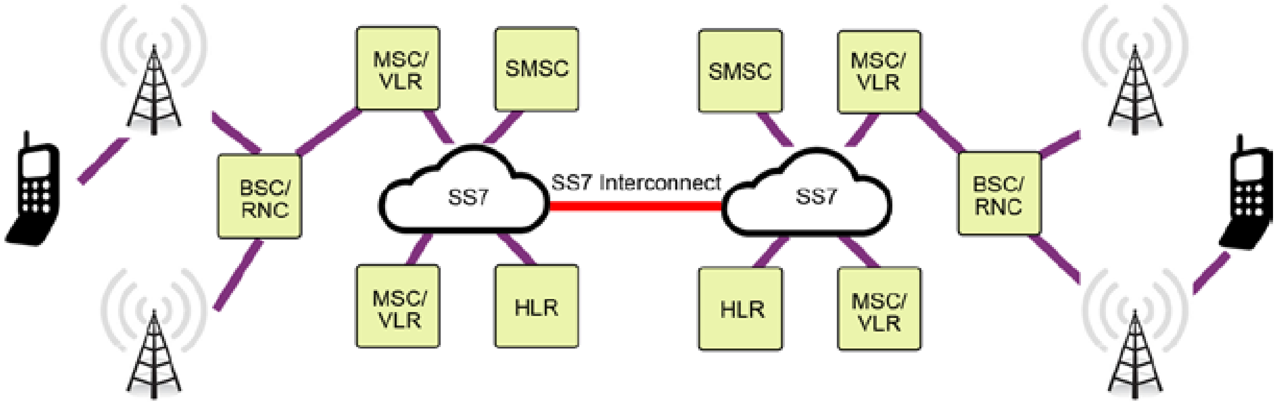
Murat Şişman • info@muratsisman.com

THE SS7 PROTOCOL AND POTENTIAL DANGERS IN GSM NETWORKS

SS7 (Signal System 7) is a communication network that allows mobile phones communicate with GSM carriers other than their own carriers. This network that has an international standard works in the same way everywhere in the world. To give an example, the network we use with wireless modem in our homes can be defined as our local network. When we connect to the internet with our modem, we also connect to the international internet network. SS7 network is just like this. Your GSM carrier is your local network, however, in order to communicate with another carrier you must connect to the international network SS7, that is why all GSM processes take place through the SS7 network.

SS7 was developed in 1975 and since it has not been developed much since then, is full of potential vulnerabilities. Although the first vulnerability known in this field was announced in 2008, much longer ago, researchers such as Kevin Mitnick exploited the vulnerabilities of GSM systems, using them for whatever purpose they wanted. Again, it was reported in 2014 that this system has 70% potential vulnerabilities globally.

By exploiting the SS7 vulnerabilities, it is possible to monitor almost every action of mobile phone users as well as intervening and performing actions on their behalf. One of the most spreaded news in this field is the two-factor authentication (2FA) vulnerabilities. Through the SS7 network, such news are frequently encountered that many people had their internet banking accounts seized using a method in which the SMS messages sent to the target are redirected to another phone.



Some of the potential vulnerabilities:

- Learning the geographic location of the target
- Redirecting the SMS traffic of the target to another number
- Having the encryption key of the target and decrypting the traffic that is being listened
- Learning such data as IMSI, IMEI, MSISDN

Learning the Current Location of Target Mobile Phone

Before the 2000s, in electronic systems security had always been kept at the second place while designing some features for the convenience of users, because factors like cyber attacks were not conceived as risky.

In the SS7 network, the feature enabling the target's location with IMSI or MSISDN (telephone number) for public or corporate use can be used maliciously by the attackers. When this feature that is used for vehicle or personnel tracking and target cell phone number is used in a specific method in the SS7 network, the base station that number is connected to can be learned immediately. There exist code numbers specific to base stations all around the world and by using the code number; the country, city and location of that base station can be displayed via the internet, allowing the target cell phone's location to be instantly found with 50 meters tolerance. Moreover, in any way, the target can not be informed. As I mentioned above, this feature is still used for tracking in fields like sea and land transportation (this method in the SS7 network cannot be used in Israeli lands).

Redirecting Target's SMS Traffic to Another Number

One of the biggest vulnerabilities is the possibility of redirecting the target phone's network traffic through SS7 to another phone determined by the attacker. I, of course, am not going to cover how this is done, but we can talk a bit about the reasons of the vulnerability.

MSISDN = Mobile phone number

IMSI = International Mobile Subscriber Identity

First 3 digits: Country Mobile Calling Code (MCC) - 286 for Turkey

Other 2 digits: Mobile Network Code (MNC) i.e. GSM carrier code (for example, Turkcell 02)

Next 10 digits: Code determined by the carrier - 286 02 xxxxxxxxxx

IMEI = Special code number specific to each mobile phone

We mentioned that in the SS7 network, we can learn the location of the target phone, just as with this method, again with the phone number or IMSI number, it is possible to access MSISDN, IMSI, and IMEI numbers. In fact, almost all information of the target cell phone can be accessed. From this point on, attackers can use these information to copy target SIM card or again over this system, can redirect to cell phones under their control.

Such features in SS7 network are done by using the cell phone as modem. However, since this kind of features are disabled in new generation telephones, older cell phones that use the Qualcomm chipset are used.

You Can Set Up Your Own GSM Carrier!

Although the GSM carriers own countless electronic hardware, everything is controlled by software. Everyone who has a Full Duplex featured hardware (both transmitter and receiver at the same time, USRP, BaldeRF) and an antenna can set up their own GSM carrier. A software called OpenBTS provides you with everything you need to set up your own GSM carrier. On this occasion, we once again understand the importance of the open source world with software such as OpenBTS.

With OpenBTS, we too can own almost all of the features that other carriers do. In fact, if we had a strong enough antenna network, we could even make GSM broadcast to all of Istanbul. When this beautiful software is used by people with bad intentions, enormous information can be seized. Just as we can create our own carrier called MuratCell using OpenBTS and create a code specific to our carrier, it also is possible to provide service as the target carrier using the information of a real GSM carrier. At places where the GSM base stations cannot serve, portable base station tools use a similar method.

All information received and sent by a cell phone connected to an OpenBTS network created with the information of target carrier can be saved by the attacker. A photograph taken during a BlackHat conference shows how easy this process is.

Requirements:

- Linux Operating System
- OpenBTS (openbts.org)
- Full Duplex Hardware (USRP, BladeRF)

Murat Kaygısız • TA1IHE@arkakapidergi.com

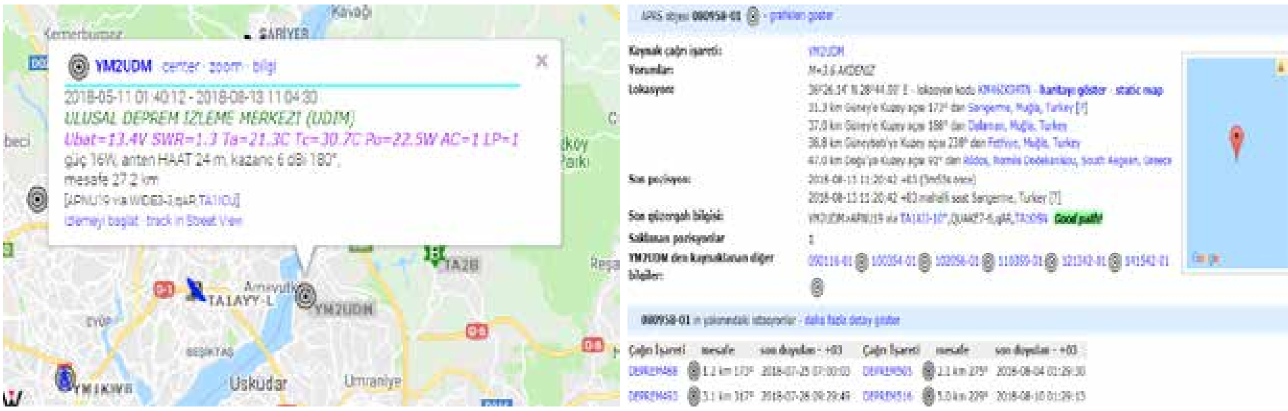
APRS: WHAT IS IT, WHAT DOES IT DO AND HOW TO USE IT

Dear readers, I would like to begin by offering you my regards. As stated consistently, Amateur Radio -partially- allows the principle of research and development within Radio Frequencies. This relatively partial freedom actually brings us unlimited freedom. APRS which has been put into use and developed setting off of this basis therefore became the subject of this issue.

What does APRS mean?

APRS is comprised of the initials of "Automatic Packet Reporting System." The most significant objective is to automatically transmit / broadcast a variety of information (location, weather, text message, etc.) over a particular radio / radio frequency in a certain data format. It is possible to send any desired data such as GPS location, warmth, humidity, altitude, voltage, earthquake information etc. through APRS.

Although sending location information is the main aim, APRS is also used to send weather report, details of wind, pressure and earthquake.



YM2UDM
2018-05-11 01:40:12 - 2018-08-13 11:04:30
ULUSAL DEPREM İZLEME MERKEZİ (UDIM)
Ubat=13.4V SWR=1.3 Ta=21.3C Tc=30.7C Po=22.5W AC=1 LP=1
güç 16W, anten HAAT 24 m, kazanç 6 dBi 180°,
mesafe 27.2 km
[APNU19 via WIDE3-2,qAR,TA1ICU]

APRS objesi 080958-01

Kaynak çağrı işareti: YM2UDM
Yorumlar: M=3.6 AKDENİZ
Lokasyon: 36°26.14' N 28°44.00' E - lokasyon kodu KM46XK
31.3 km Güney'e Kuzey açısı 173° dan Sarıgerme, Muğla, Turkey [?]
37.0 km Güney'e Kuzey açısı 188° dan Dalaman, Muğla, Turkey
38.8 km Güneybatı'ya Kuzey açısı 238° dan Fethiye, Muğla, Turkey
47.0 km Doğu'ya Kuzey açısı 92° dan Rodos, Nomós Dodekanísou, South Aegean, Greece
Son pozisyon: 2018-08-13 11:20:42 +03 (3m53s önce)
2018-08-13 11:20:42 +03 mahalli saat Sarıgerme, Turkey [?]
Son güzergah bilgisi: YM2UDM>APNU19 via TA1AYY-10*,QUAKE?-6,qAR,TA2
Saklanan paketler: 1
YM2UDM den kaynaklanan diğer bilgiler: 090116-01 100204-01 102056-01 110359-01 121342-01 141542-01

080958-01 in yakınındaki istasyonlar

Çağrı İşareti	mesafe	son duyulan - +03	Çağrı İşareti	mesafe	son duyulan - +03
DEPREM468	1.2 km 173°	2018-07-25 07:00:00	DEPREM505	2.1 km 275°	2018-08-04 01:29:30
DEPREM463	3.1 km 317°	2018-07-28 09:29:49	DEPREM516	5.0 km 299°	2018-08-10 01:29:15

National Earthquake Monitoring Center APRS Image Example

Who Can Use APRS?

The APRS system can only be used by persons with Amateur Radio Operator certificate. The automation password used to access the structure login system is created by a combination in which the Amateur Radio Call Sign is also used. Long story short, it is a special structure for Amateur Radio Operators.

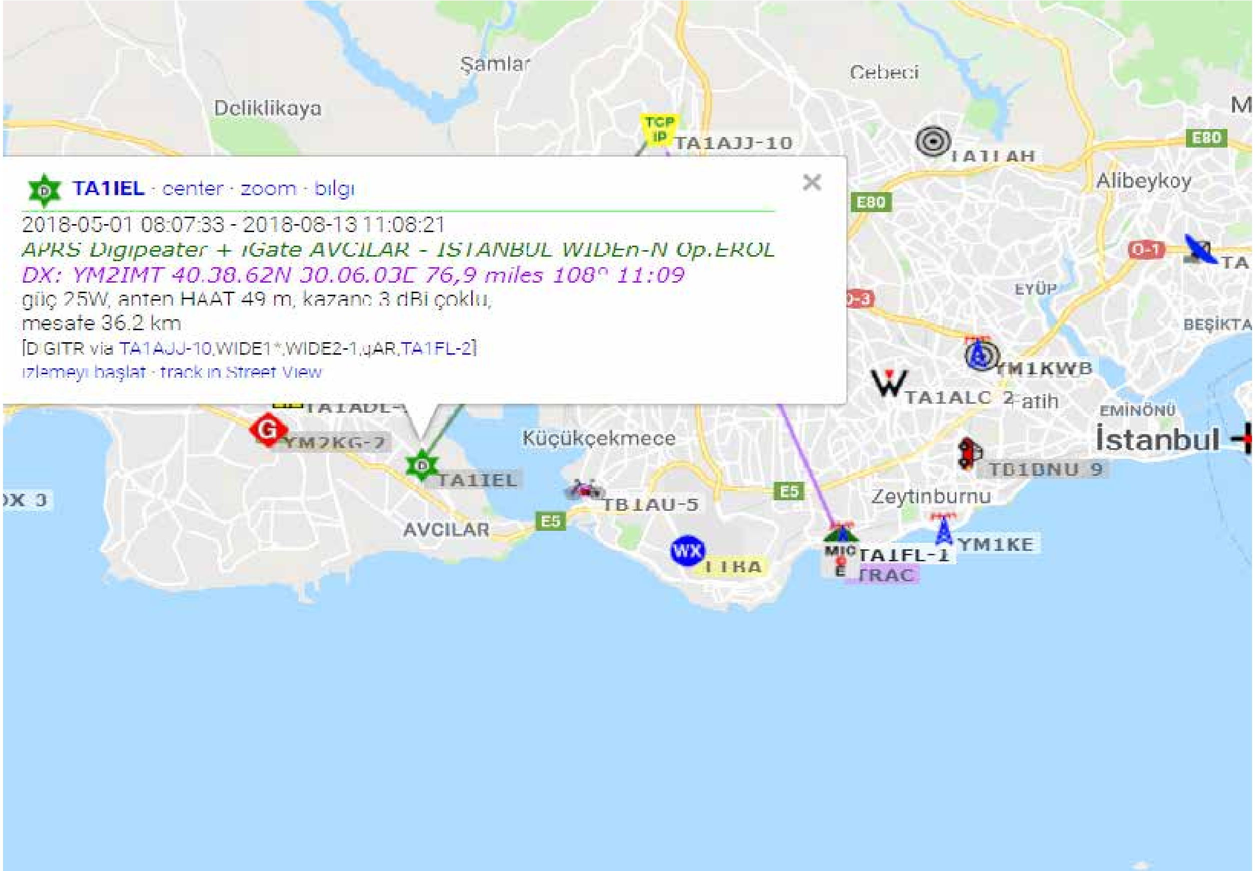
APRS systems can be considered as an alternative to the Vehicle Tracking Systems given by the GSM carriers. However, since the vehicle tracking systems provided by GSM carriers are dependent on internet infrastructure, it can be seen that it is actually completely different from the Amateur Radio structure. Because the main purpose of APRS systems is for the radio operator locations to be seen directly via Radio Frequencies in possible emergency cases, its ability to work independently from any internet infrastructure is the main idea.

A system that is disabled for a while because of infrastructure interruptions is completely contrary to the Amateur Radio perspective. As you know, under any circumstances, this is contrary to the idea of communication. APRS system is a completely free structure.

On Which Type Of Devices Can The APRS System Be Used?

There are lots of alternative uses of the APRS system, but using it as a wireless device is the most suitable for the main structure. Nowadays APRS system is available on many new generation devices. The most important thing to understand here is that a device capable of shooting APRS is able to read the messages sent in the same way.

The system works as follows: the location information, altitude, vehicle or pedestrian speed and similar information are digitally coded and sent through an RF. The devices in the area of the target are able to receive this information. For areas where the propagation is not sufficient, repeater systems called DIGIPEATER and I-GATE come in handy.



TA1IEL · center · zoom · bılgı

2018-05-01 08:07:33 - 2018-08-13 11:08:21

APRS Digipeater + iGate AVCILAR - ISTANBUL WIDEn-N Op.EROL

DX: YM2IMT 40.38.62N 30.06.03E 76,9 miles 108° 11:09

güç 25W, anten HAAT 49 m, kazanç 3 dBi çoklu,

mesafe 36.2 km

[DIGITR via TA1AJJ-10,WIDE1*,WIDE2-1,qAR,TA1FL-2]

izlemeyi başlat · track in Street View

An example of DIGIPEATER and I-GATE Application

As seen on the map below, the APRS data are sent in the VHF band at 144,800 MHz frequency in Turkey, but are different around the globe. In other words, the frequency that should be used in Turkey differs from that in other countries. The reason for this is the change shown in the bandwidth found suitable within national frequency plannings.

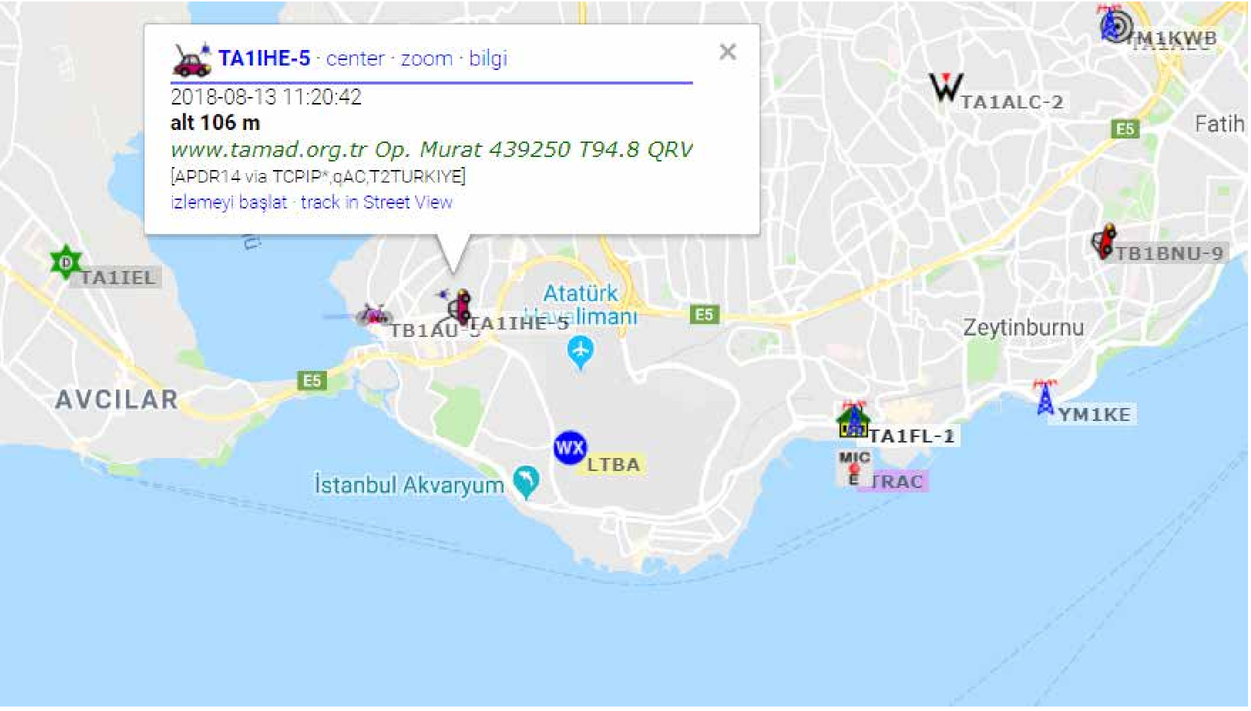
Here, the repetition of APRS packages through the radio device is very important. The more frequently the user repeats the signal, the more complexity the system will experience, hence the server systems repeating the sent signals will experience the same amount of business. Since the readability of the packages reaching the system will be corrupt, the data sent will lose its importance. The sound level of digital toning carrying the data for the APRS systems -excluding the original device systems- is also very significant. It is crucial that the level is precisely set at a point making the readability appropriate.

Can APRS Be Used Without Radio ?

The answer to that question is a yes! Apart from using the system over the radio as it is supposed to be, getting involved in the system is also possible with systems working on computers or with applications on the smart phones. However, it should be configured in a way that doesn't allow any businesses. A well-synced structure has to be created so that the main structure is not harmed. The signals received from the air by I-GATE and DIGIPEATER systems mentioned above are transferred to other systems via repetition or the internet. In the meantime, it is possible to share data packages with approximately 15 different stations for each user.

In other words, some operators do not use the APRS system through RF. Instead, they track APRS using certain applications on a computer. However, the data packages travel through the internet, visiting all the servers. Since these can generate RF, the entire area turns into a large radio frequency through APRS. Briefly, we can say that the data packages generated with computers or smartphones can be sent by the radio devices, as well as being sent to the air as it reaches DIGIPEATER.

When it all comes together, what needs to be remembered is calculating the density that will form over the zone frequency. Each data mixed on the frequency and their occurrence rate should be in such a way not to disrupt functionality. The ideal measure of data shooting interval is at least 1 minute for a radio device made into APRS . If the non-moving or non-vital data are in a stationary system, this interval should be chosen to be at least 30 minutes.



TA1IHE-5 · center · zoom · bilgi
2018-08-13 11:20:42
alt 106 m
www.tamad.org.tr Op. Murat 439250 T94.8 QRV
[APDR14 via TCPIP*,qAC,T2TURKIYE]
izlemeyi başlat · track in Street View

Example of APRS Station Information

Does APRS Include Only the Location Information?

Since lots of features on the APRS structure are supported with computer and internet structure, the answer to this question would be “NO”. Messaging and even emailing are among the options with the APRS system. For these types of uses, the properties of the system you have should be examined. If you are connected to the system with the help of a computer, it is possible to use these things mentioned. Radio devices that will allow you to do all these operations in hardware are available on the market.

Here we share with you a website that you can examine and watch this frequently used system: **https://aprs.fi**

As you can see via this link, the usage of the system in Turkey is actually relatively very low compared to other countries. As we say all the time, in a possible disaster or emergency case, any infrastructure will collapse, however the structure working the RF way will always stand.

It should not be forgotten that in a case of a power outage, the internet and smartphones that make everyday life easier and the stations that are thought to be using APRS and are active on the map would become passive. Hoping that the number of those who understand the real importance of radio devices and systems increase. See you in the next issue, 73!

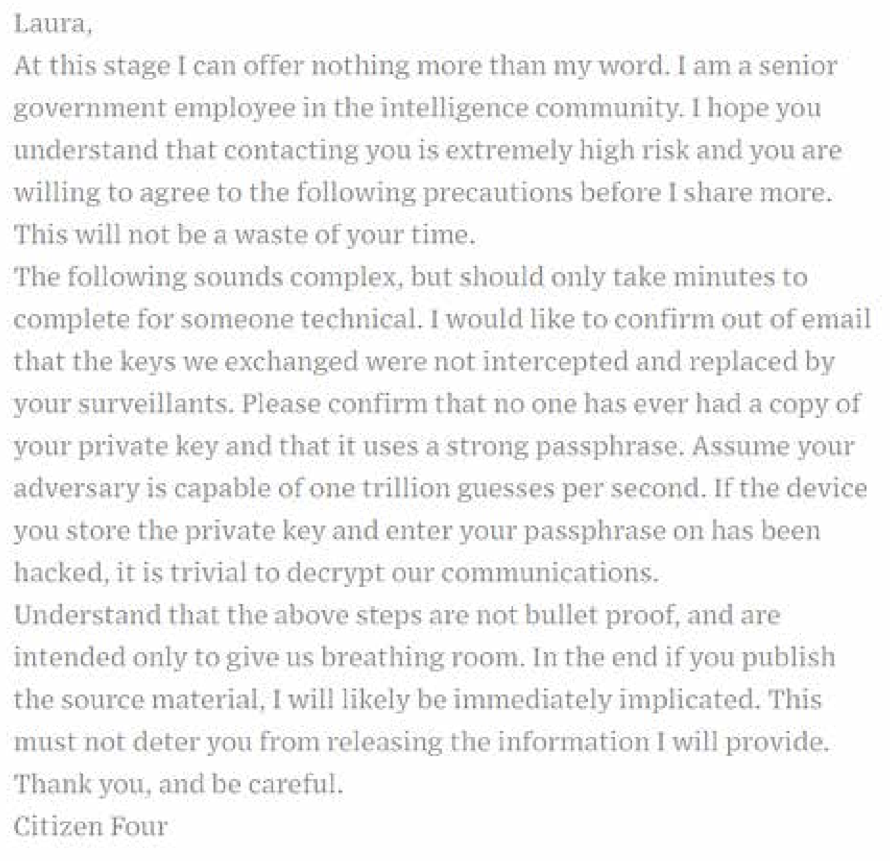
Recep Kızılarslan • recep@arkakapidergi.com

KEY EXCHANGE PROBLEM IN PUBLIC KEY CRYPTOGRAPHY AND KEYBASE

Before his famous disclosure, Edward Snowden decided to make contact with the film director Laura Poitras. He wanted to make sure that the things he share with her will remain secret and unchanged by any potential third party intruder.

The only way to ensure data integrity and security is to use the public key pinning (PKP) method we'll describe below.

Snowden taught Poitras how to generate a PKP and then requested her to share it with him. However, he also had to make sure of the inregrity of the public key. He could only ensure this either by having a mutual friend of Poitras and Snowden to confirm that the key is indeed Poitras' or Poitras had to confirm this on a platform which Snowden is absolutely certain that the owner is Poitras. When she tweeted her key summary which seemed like gibberish on her Twitter, Snowden was convinced and clear that the key is indeed hers.



Laura,

At this stage I can offer nothing more than my word. I am a senior government employee in the intelligence community. I hope you understand that contacting you is extremely high risk and you are willing to agree to the following precautions before I share more. This will not be a waste of your time.

The following sounds complex, but should only take minutes to complete for someone technical. I would like to confirm out of email that the keys we exchanged were not intercepted and replaced by your surveillants. Please confirm that no one has ever had a copy of your private key and that it uses a strong passphrase. Assume your adversary is capable of one trillion guesses per second. If the device you store the private key and enter your passphrase on has been hacked, it is trivial to decrypt our communications.

Understand that the above steps are not bullet proof, and are intended only to give us breathing room. In the end if you publish the source material, I will likely be immediately implicated. This must not deter you from releasing the information I will provide.

Thank you, and be careful.

Citizen Four

What if when the two parties first established contact and a third party intervened and pretending to be Poitras sent his key to Snowden?

In this article, we're going to talk about a quick and easy solution to problems of sharing public keys as such.

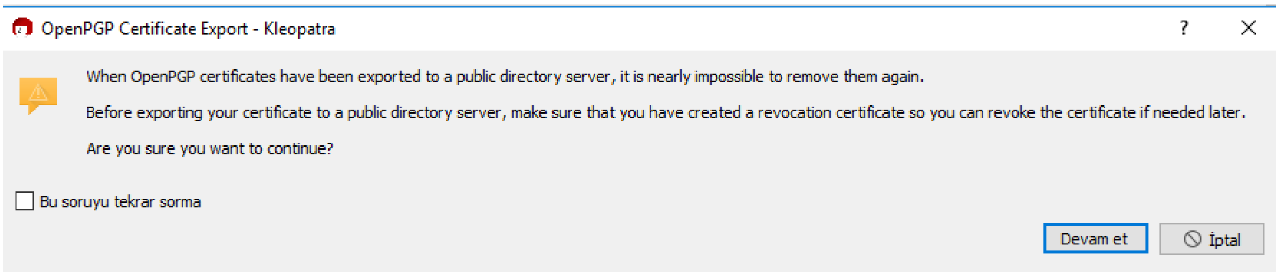
We all know by now that on the internet, all the unencrypted data can be read and modified on the locations it hops during transactions. Therefore, over time many encryption algorithms are born. In fact, the history of encryption began long before computers and even electricity were invented.

The reason why PGP has become so widespread is most certainly the Public Key Pinning technology. You can generate and share your public and private key pair and encrypt your text and files using PGP. Here are the steps to secure messaging with PGP:

- The sender encrypts the message using the recipient's public key.
- The encrypted message is sent to the recipient.
- The recipient decrypts the message using her private key.

PGP also has some downsides to it. The short issue between Snowden and Poitras in the beginning of this article is one of these.

Users can share their keys with one another using various platforms but securely messaging with someone you've never met before brings you to a dilemma. The most common solution to this are the public key servers which hold the keys of many users. However, these servers have two problems. One is that you cannot remove your keys from the server, ever (unless you take extra steps). If your keys expire or become unusable, they'll be stuck in the server forever.



When OpenPGP certificates have been exported to a public directory server, it is nearly impossible to remove them again.

Before exporting your certificate to a public directory server, make sure that you have created a revocation certificate so you can revoke the certificate if needed later.

Are you sure you want to continue?

☐ Bu soruyu tekrar sorma

The second problem in key servers is that they don't directly verify the identity of the owners. You can easily generate a PGP key in the name of anyone, such as Kevin Mitnick or Bill Gates. Someone who searches for bill@microsoft.com or kevin@mitnick.com on the servers will be able to see the keys you uploaded on their name.

This can be prevented by verifying the identities using third party mutuals. This method resembles the verification of TLS certificates of the websites through trustable third-party certificate authorities.

We'll be introducing an alternative solution to PGP's sharing public keys problem: Keybase.

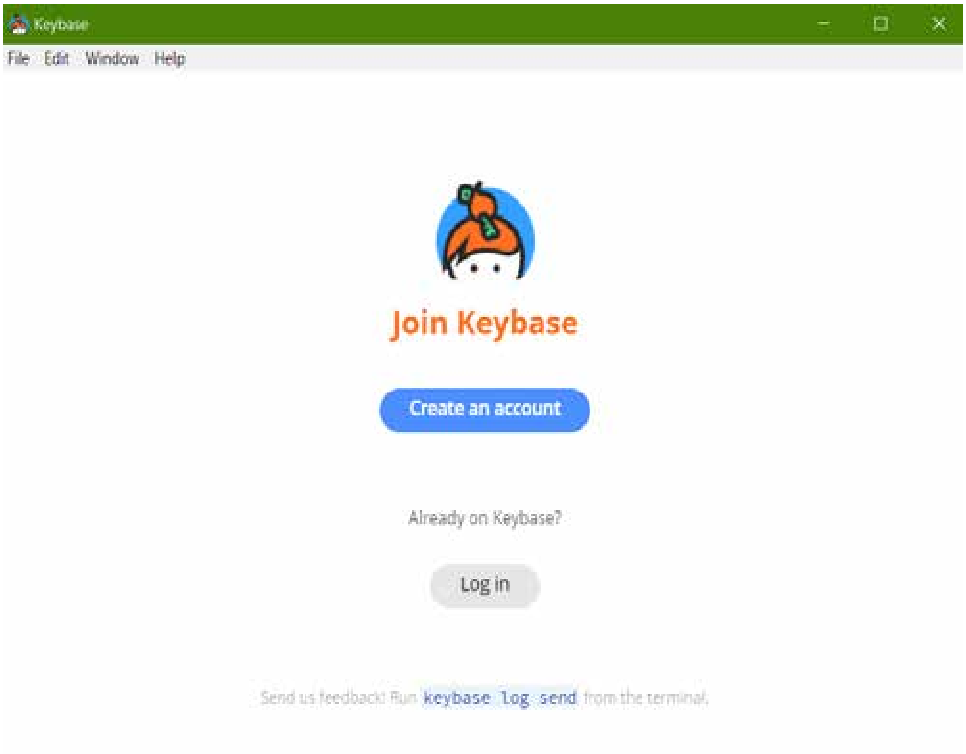
You can verify your identity and validate your public key using your domain and accounts on Bitcoin Wallet, Twitter, Github, Reddit, and Hacker News.

Plus, someone who doesn't understand the technical details of public key encryption can send secure messages to you over Keybase.

Here's how you can use Keybase:

Download Keybase on your desired platform (Mac OS, Linux, Windows, iOS, Android) from https://keybase.io/download. You can also add the extensions of Keybase on your Chrome and Firefox browsers.

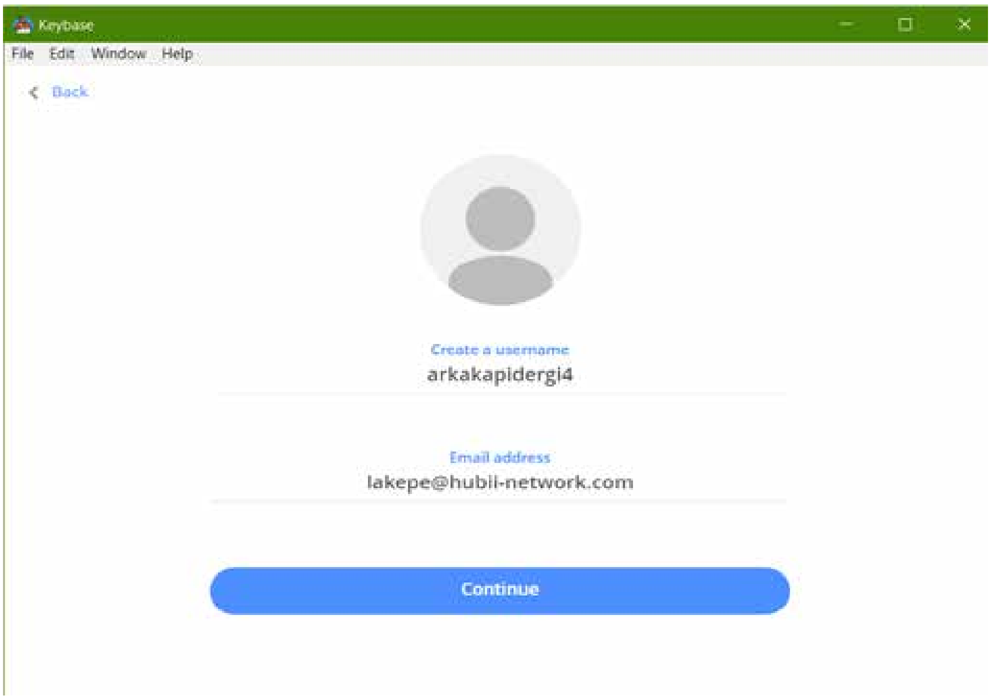
We're using Windows so we'll download the Windows installation file. Once the download is complete, we're launching the file, and after a short setup, we're introduced with the following screen:



Already on Keybase?

Send us feedback! Run keybase log send from the terminal.

After we click the **"Create an account"** button, we specify a username and an email address. I'm going to give a random email address I got from TempMail[1].

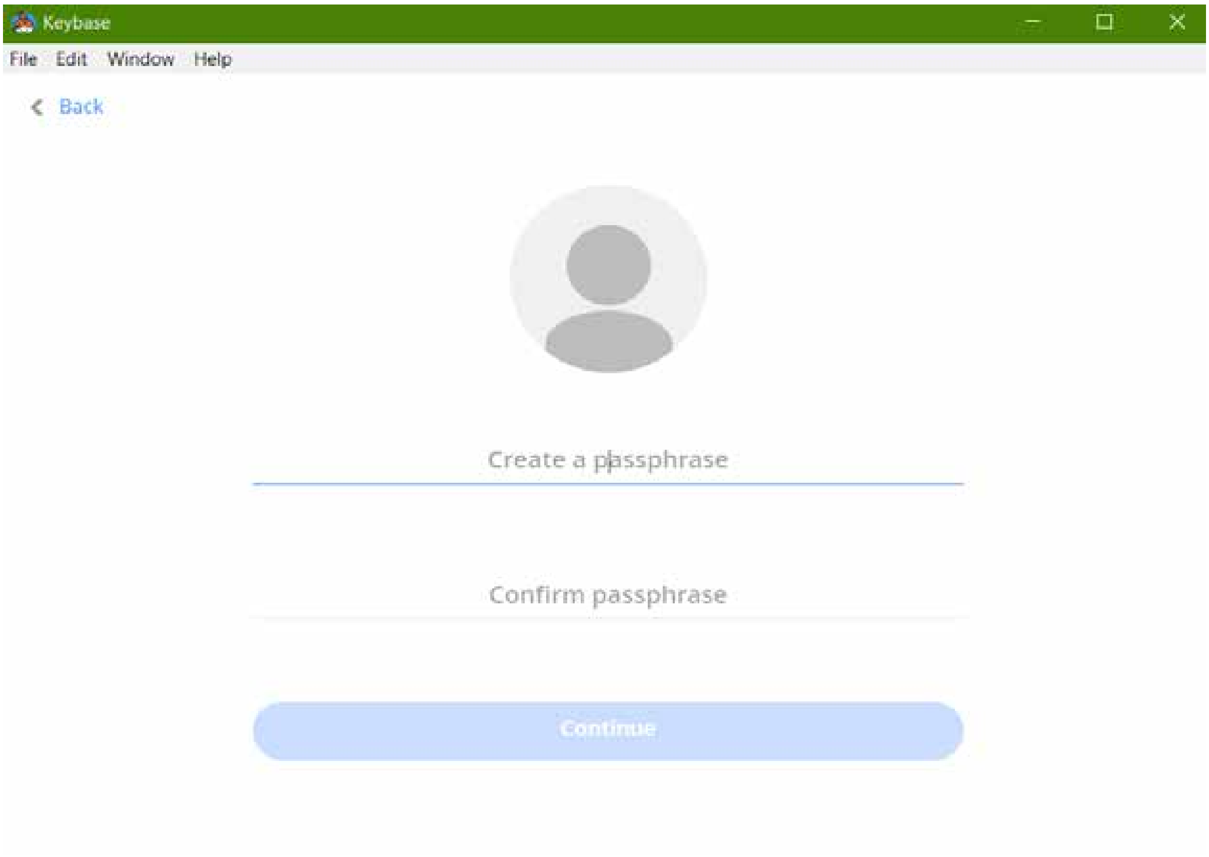


Create a username
arkakapidergi4

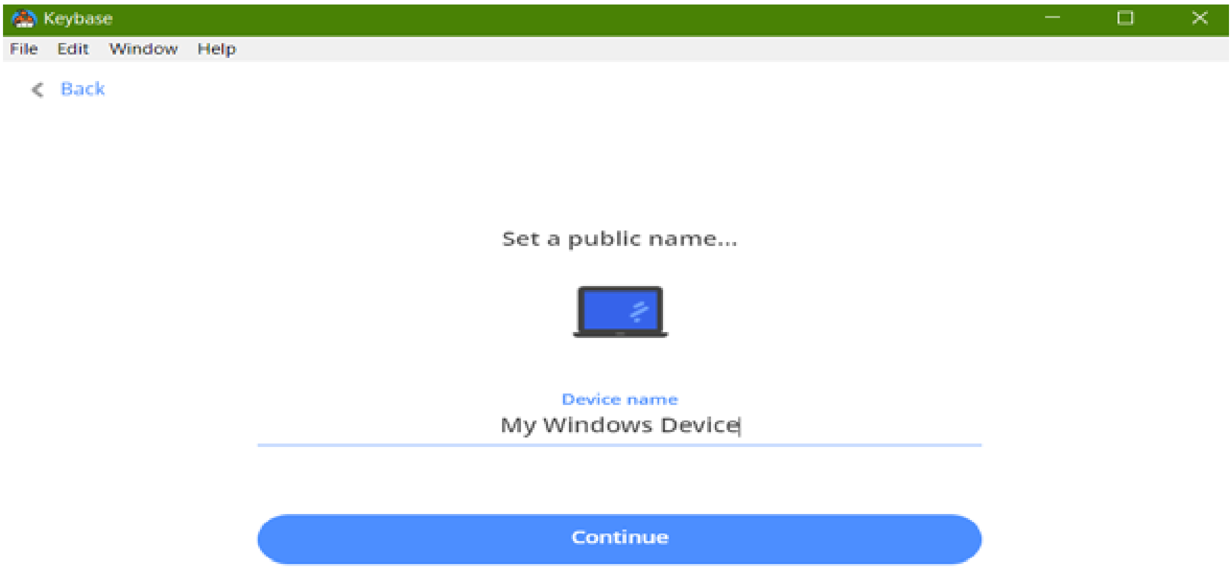
Email address
lakepe@hubli-network.com

[1] **TempMail:** An email platform that gives you a random email without the entire registration process. **Details:** https://temp-mail.org/

Next, we'll be asked to set a password. We're using KeePass application to generate a strong password.

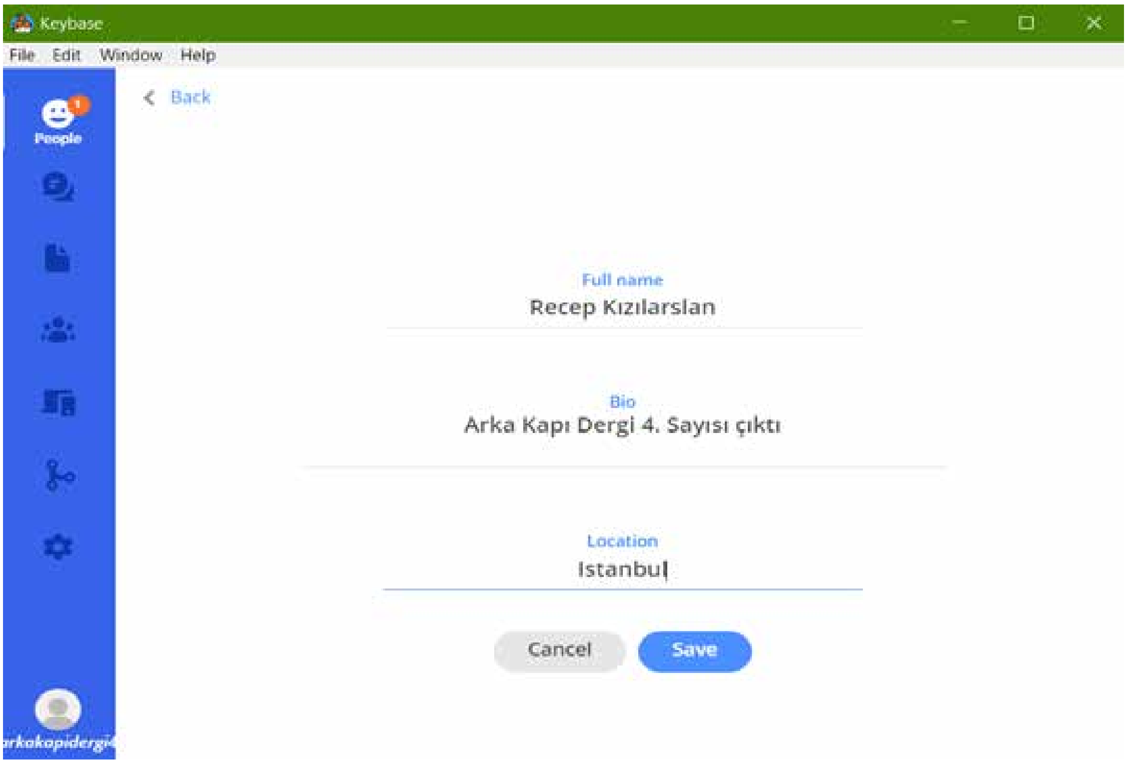


In the next step, we have to name our device and add the necessary information for our profile



Set a public name...

Device name: My Windows Device

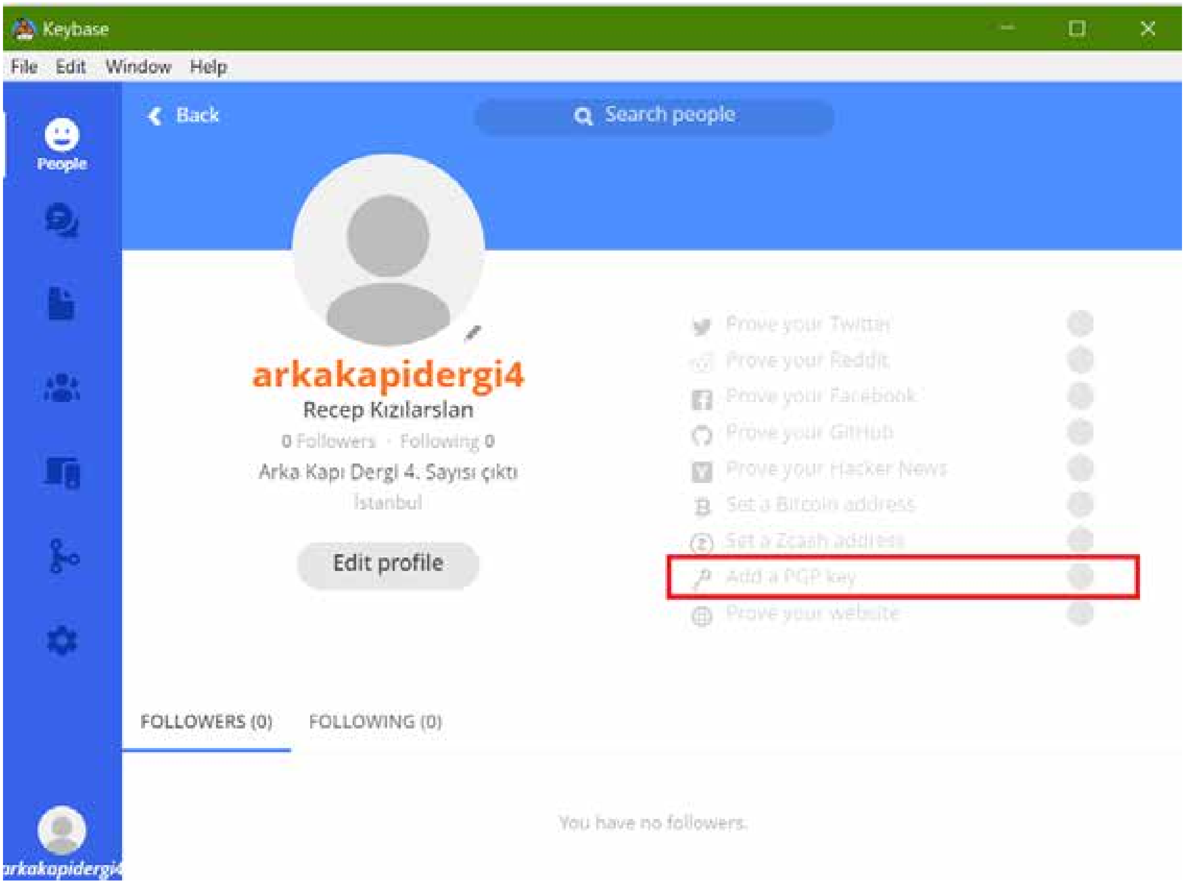


Full name
Recep Kızılarslan

Bio
Arka Kapı Dergi 4. Sayısı çıktı

Location
Istanbul

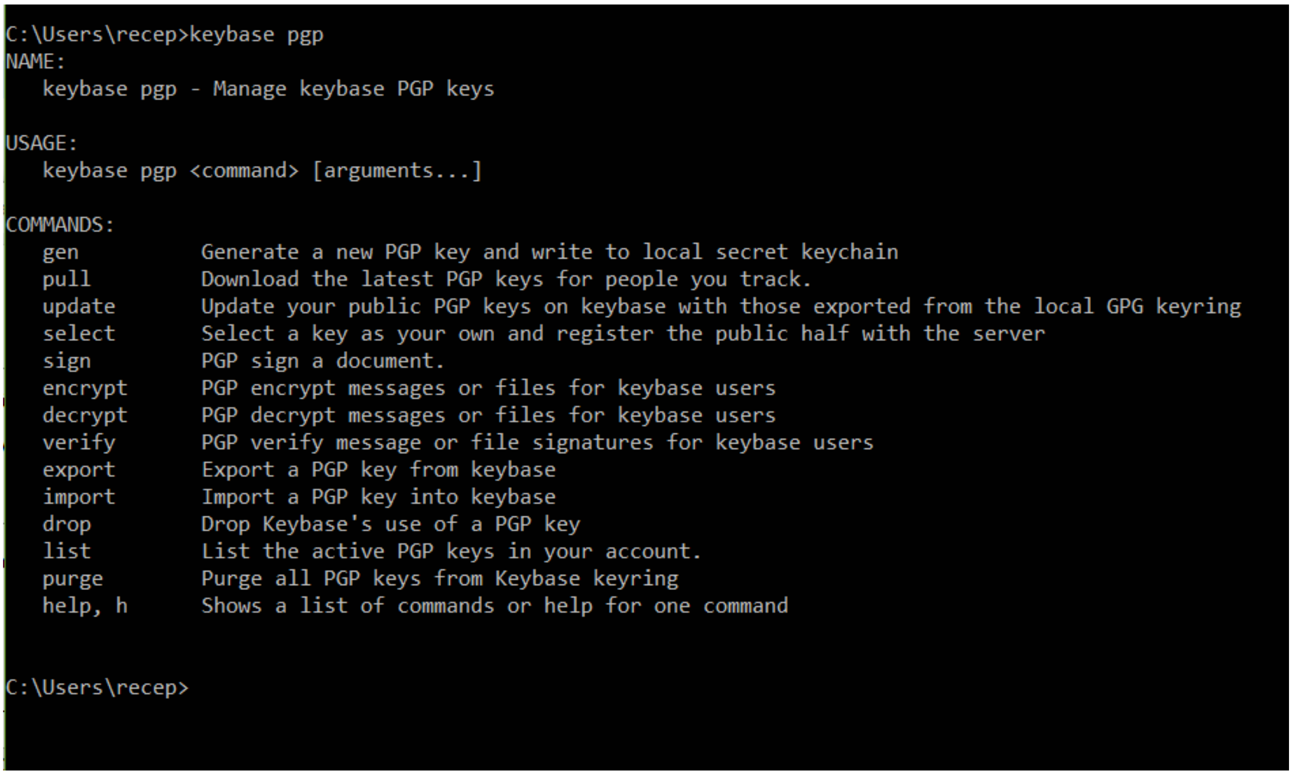
After we set our profile, we add our PGP key. You can generate a new PGP key pair using Keybase or can add a previously generated key to your profile. You can add one by clicking "Add a PGP Key".



arkakapidergi4
Recep Kızılarslan
0 Followers · Following 0
Arka Kapı Dergi 4. Sayısı çıktı
Istanbul

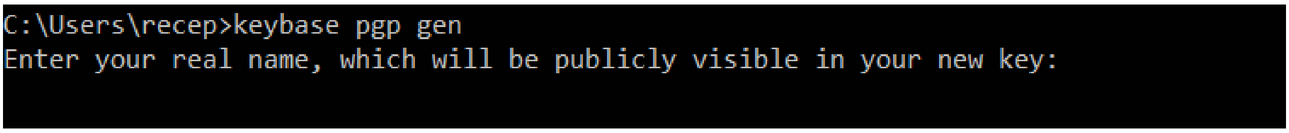
You have no followers.

We're going to make a new PGP key using Keybase from the Windows command line. You can make or import PGP key pairs using the command line interface (CLI). Write "keybase pgp" to see the available Keybase parameters.



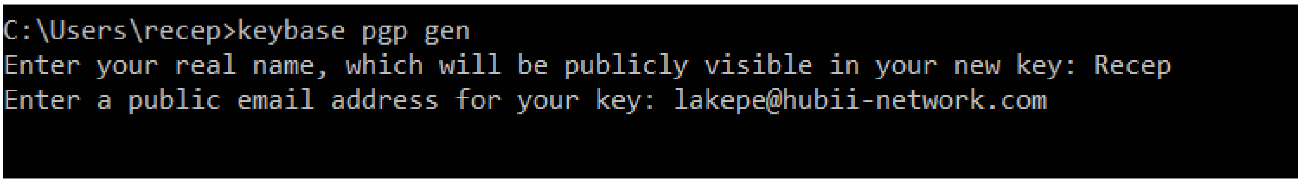
```
C:\Users\recep>keybase pgp
NAME:
   keybase pgp - Manage keybase PGP keys

USAGE:
   keybase pgp <command> [arguments...]

COMMANDS:
   gen          Generate a new PGP key and write to local secret keychain
   pull         Download the latest PGP keys for people you track.
   update       Update your public PGP keys on keybase with those exported from the local GPG keyring
   select       Select a key as your own and register the public half with the server
   sign         PGP sign a document.
   encrypt      PGP encrypt messages or files for keybase users
   decrypt      PGP decrypt messages or files for keybase users
   verify       PGP verify message or file signatures for keybase users
   export       Export a PGP key from keybase
   import       Import a PGP key into keybase
   drop         Drop Keybase's use of a PGP key
   list         List the active PGP keys in your account.
   purge        Purge all PGP keys from Keybase keyring
   help, h      Shows a list of commands or help for one command


C:\Users\recep>
```

We're going to write the "keybase pgp gen" command to make a new PGP key. We have to enter our name which will be visible in the key.



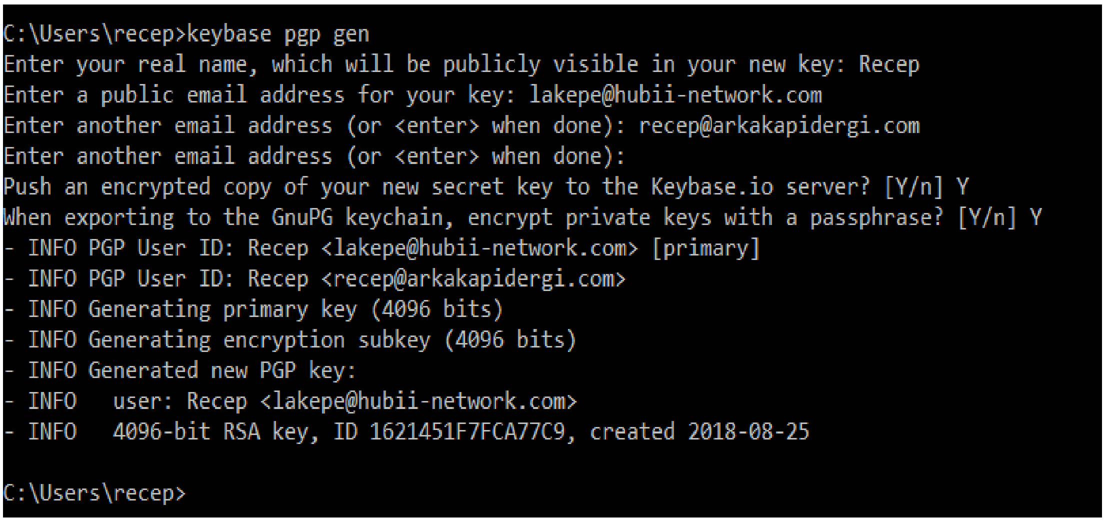
```
C:\Users\recep>keybase pgp gen
Enter your real name, which will be publicly visible in your new key:
```

Then we enter the email address required for the key.



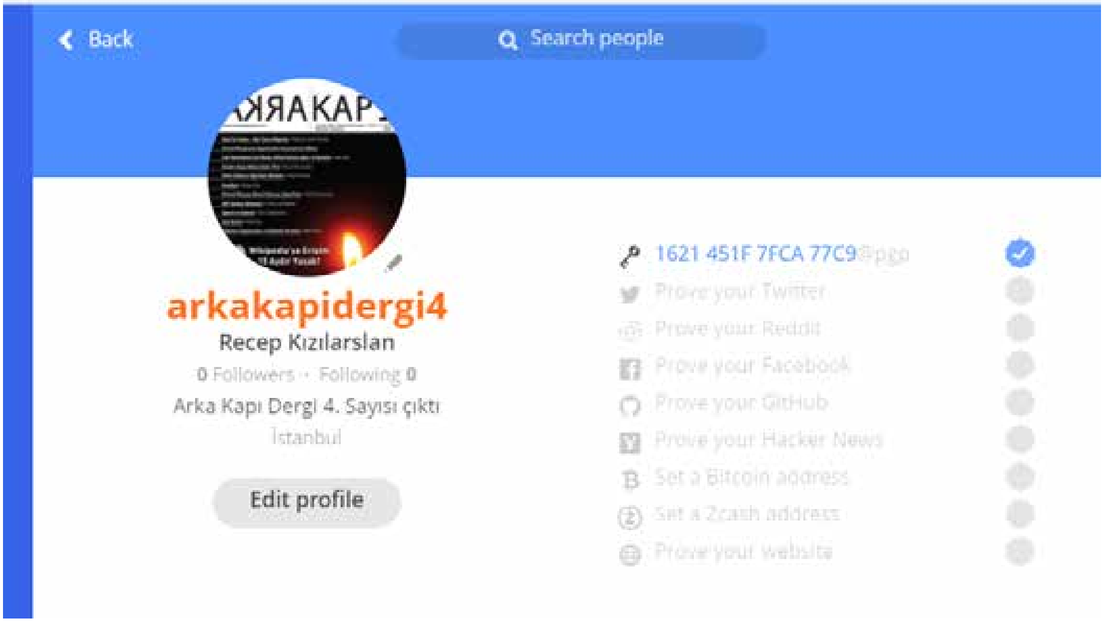
```
C:\Users\recep>keybase pgp gen
Enter your real name, which will be publicly visible in your new key: Recep
Enter a public email address for your key: lakepe@hubii-network.com
```

After we enter a second email address, Keybase asks if you'd like your key to be stored in its servers in case the key gets lost.



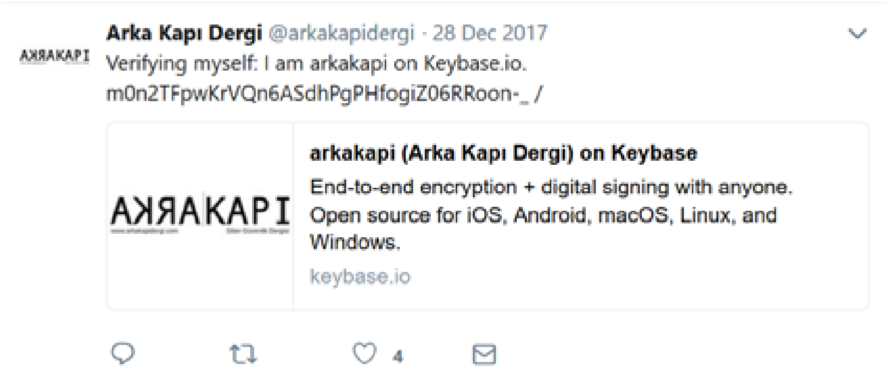
```
C:\Users\recep>keybase pgp gen
Enter your real name, which will be publicly visible in your new key: Recep
Enter a public email address for your key: lakepe@hubii-network.com
Enter another email address (or <enter> when done): recep@arkakapidergi.com
Enter another email address (or <enter> when done):
Push an encrypted copy of your new secret key to the Keybase.io server? [Y/n] Y
When exporting to the GnuPG keychain, encrypt private keys with a passphrase? [Y/n] Y
- INFO PGP User ID: Recep <lakepe@hubii-network.com> [primary]
- INFO PGP User ID: Recep <recep@arkakapidergi.com>
- INFO Generating primary key (4096 bits)
- INFO Generating encryption subkey (4096 bits)
- INFO Generated new PGP key:
- INFO   user: Recep <lakepe@hubii-network.com>
- INFO   4096-bit RSA key, ID 1621451F7FCA77C9, created 2018-08-25

C:\Users\recep>
```

We can see that we successfully generated a PGP key with the ID "*1621451F7FCA77C9*". Now we can go back to our profile and link our other social media accounts. The verification/proofing method is highly improved on Keybase, therefore it gives us many options for confirming our key ownership.



arkakapidergi4
Recep Kızılarslan
0 Followers · Following 0
Arka Kapı Dergi 4. Sayısı çıktı
İstanbul

1621 451F 7FCA 77C9 @pgp



Arka Kapı Dergi @arkakapidergi · 28 Dec 2017
Verifying myself: I am arkakapi on Keybase.io.
m0n2TFpwKrVQn6ASdhPgPHfogiZ06RRoon-_ /

arkakapi (Arka Kapı Dergi) on Keybase
End-to-end encryption + digital signing with anyone. Open source for iOS, Android, macOS, Linux, and Windows.
keybase.io

We can share the verification message prepared by Keybase on our Twitter, Facebook, and Github accounts:

Thus, anyone can verify their identity through their social media or crypto currency accounts supported by Keybase.

Cansu Topukçu • topukcucansu@gmail.com

OLDEST OF THE HACKERS I

BILL GOSPER

"This Hacker Ethic is their gift to us: something with value even to those of us with no interest at all in computers. It is an ethic seldom codified, but embodied instead in the behavior of hackers themselves. I would like to introduce you to these people who not only saw but lived the magic in the computer, and worked to liberate the magic so it could benefit us all." - Steven Levy, HACKERS: Heroes of the Computer Revolution

The essence of being a hacker involves discovering where you can push your limits, going through wit. If we examine the word "hack" itself, contrary to what it is perceived like nowadays, any activity done with a playful cleverness has a hack value, and whomever does this activity is said to be a hacker. These activities include developing software, discovering the tunnels of MIT's campus, as well as a quick-witted joke.

As most of us know the early history of hackers begin at the AI (Artificial Intelligence) Lab, and a student club called TMRC (Tech Model Railroad Club) in the MIT, dating back to the 50s and 60s. These two precious places are where the people who laid a foundation for today's hacker culture grew up and found their feet, along with the hacker culture itself blooming! We can say that the seeds of being a hacker were sowed into the hackers of the MIT with the help of something in the AI Lab, PDP-1: the first one of the PDP (Programmed Data Processor) series. TMRC is a student community who make very detailed models of railway tracking systems, trains, networks and even cities bringing them to life with real electronic circuits. It is divided into two teams: the first team dealt with modeling and landscaping, whereas the other team -called the Signals and Power Subcommittee - created the circuits required for the trains to work. Most members of the second team later shifted to computer programming and are among the first hackers. Legendary hackers of this period such as Bill Gosper, Peter Deutsch, Richard Greenblatt and Tom Knight have emerged from this club and spreaded the word hacker.

Just as when this concept was born, a mathematical genius from New Jersey began studying Mathematics at MIT in 1962. It was Bill Gosper who started to being pulled towards computers with the idea of hacking the *math world* itself instead of systems. As a matter of fact, Gosper is one of the few real hackers Steven Levy has mentioned in his book Hackers: Heroes of the Computer Revolution that had serious impacts on Hacker Ethics.

Ralph William Gosper Jr., born on 26 April 1943 in Pennsauken is a mathematician, programmer and hacker. Gosper's experience with computers did not beyond watching a UNIVAC printer rapidly print out Benjamin Franklin photographs from behind a glass - up until a course he enrolled in in his second semester in MIT. That programming course was given by John McCarthy only to those students who merited the semester before. The syllabus started with FORTRAN, continued with IBM Machine Language thereby leading the roads to PDP-1, which hackers also used to play Steve Russell's *Spacewar!* game. Diving deeper into the the ocean of computing day by day, Gosper was using PDP-1 all the time even after course hours, being subject to anti-computer attitude of the Math Department professors. Despite the reactions he was getting from the professors, Gosper eventually became a dexterous PDP-1 hacker, he was even using it to discover where he could push the boundaries of the mathematical world.

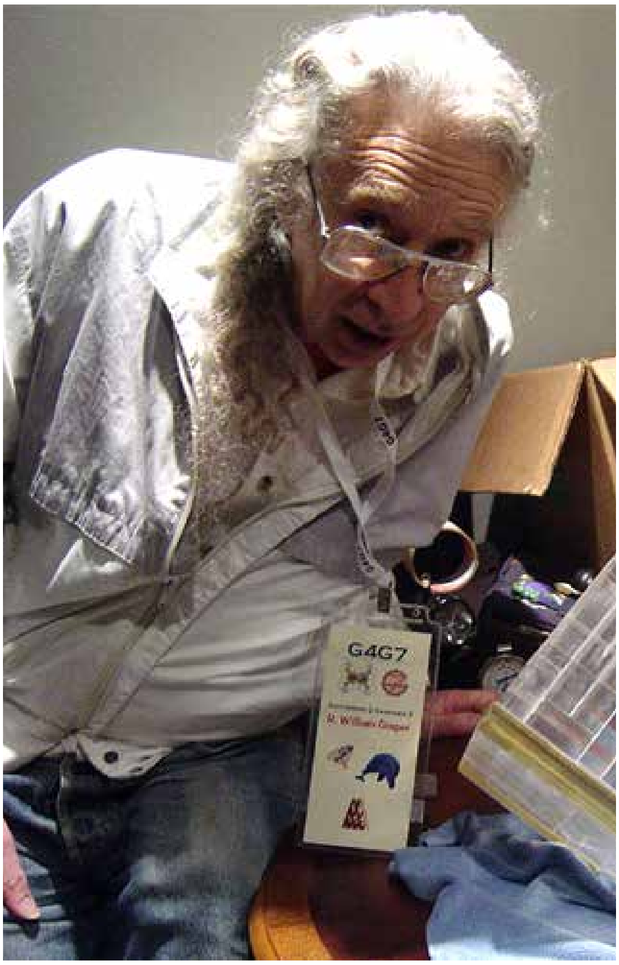
After a while, Gosper started taking Artificial Intelligence course from the director of MIT AI Lab, Marvin Minsky. Minsky, who was deeply impressed by the hackers' achievements, was also very appreciative of, and sympathized with, their desire to discover. That is why he gave them full direct access to the machines at the AI Lab. Gosper's work was again on PDP-1. As his first real project for the AI course, he wrote a program to print functions to the screen, and showed a part of his program to Alan Kotok (member of TMRC, legendary TX-0 and PDP-1 hacker MIT graduate). Kotok mentions him as having reached a "godlike status".

Gosper and two of his friends had a project idea for their AI course: to *hack* a program with the PDP-1 that will finish Peg Solitaire game with only one peg left at the center. Our hackers did not only solve the case, "*We demolished it*" Gosper later said. PDP-1 could solve the Peg Solitaire in an hour and a half! Bill Gosper was walking on the path of mathematical exploration-driven hacking.

Talking about the hacking, let's dig the hacker culture up. After some good digging, you see the Hacker Ethic at the core. The Hacker Ethic consists of the collection of morals and philosophy frequently encountered in hacker culture. Those who embrace the hacker ethic think that sharing information and data responsibly is both beneficial and benevolent. The key points to this ethic are increasing quality of life and the accessibility and freedom of information. According to the hacker ethic, access to computers—and anything which might teach you something about the way the world works—should be unlimited and total; and decentralization should be promoted. Hacker Ethic is a lifestyle, a philosophy.

One of the places where this philosophy emerged from was the Tool Room next to the TMRC. In the Tool Room, there were arguments that contributed a lot to the hacker culture. The arguments here of Gosper and his friends were what gave life to the hacker society. Alongside with his friends, Gosper fit into the hacker definition described in the first paragraph, using his creativity and intelligence not only before computer screen but in any occasion in his everyday life. That was because for him, everything was a challenge - a game, therefore having a *hack value.* If there was something he loved just as much as hacking, it was the Chinese food. When the arguments in the Tool Room extended through the night -which meant almost every night- the squad usually went to Chinese restaurants. They even gamified the menu that was written in Chinese. Peter Samson was able to speak Chinese not too badly and could even read the menu. In other words, Chinese was a system, and if there was a system, why not hack it?

Unlike his friends Greenblatt and a few other hackers who dropped out of school just because they did not want to spend their time on anything other hacking, Bill Gosper graduated from MIT in 1965. Gosper, who then took a civil service exam and placed high enough to be included in an exclusive student engineering development program before entering MIT, worked summers for the Navy from 1961 to 1964. During this time, his attempt to make them Univac-free failed. His work helped him pay half of the tuition but he had to work for the Navy for 3 years after graduation, however, Gosper resolutely decided to pay off the rest of the debt by gaining money at somewhere where he could work with a PDP-6. There were a few reasons why he didn't want to work with the Navy. First of all, he strongly disliked the Univac computers they used there. Furthermore, the navy culture that did not allow programmers near computers. So, he started working at the Adams company where Greenblatt also used to work; there even was a PDP-6. Just as Gosper wanted!



"And there's the sense of connection between you and the environment. The idea of where's the boundary of a computer. Where does the computer leave off and the environment begin? " -Bill Gosper, on LIFE simulation

LIFE, a computer simulation created by Horton Conway in 1970 drew his attention. LIFE is a 0 player game where you determine the starting conditions and observe how they evolve. Since it has the ability to simulate an infinite loop, it was considered to have the calculation ability of that of a Turing machine. The first public appearance of LIFE took place in Martin Gardner's column in Scientific American, in 1970. For Gosper, LIFE was actually questioning what *life* was and it was an untouched goldmine. According to Conway, the creator of LIFE, a population that has a starting configuration with a known number of living cells can not outgrow an upper limit, offering a 50$ prize for the first person proving or disproving the conjecture. The MIT team led by Bill Gosper won this award in November 1970 with the *Gosper Glider Gun*. The Glider Gun is a special gun that follows a pattern that produces the first glider on the 15th generation, and another one every 30th generation. The symbol mentioned here -*Glider*- proposed by Eric S. Raymond is -though disliked by some- what is known as the hacker emblem.

Later in 1972, he contributed as an author to HAKMEM, February 1972 memos - technical reports of the MIT AI Lab. These notes contain different hacks like some algorithms on cognitive maths and schematic hardware graphics. Again, in 1972, Gosper exited where he mentions that he had learned most of the things he knows: the TMRC that he had entered in 1963.

Between 1966 and 1974, Gosper wrote bignum GCD algorithm in machine language for MacLisp - the programming language in which Richard Greenblatt was the main developer of the original code base for PDP-6. Besides that, he made great contributions to project MAC's (Project on Mathematics and Computation - which had been donated $ 2 million by DARPA) computer algebra system MACYSMA by writing fibonacci and integer factor loops in Lisp. He also wrote some display hacks (mathematically motivated abstract animations), one of these led Schroeppel to discover the algorithm for integer linear programming.

Later in 1974, he moved to California where he would work 3 years at Stanford. There, at Stanford, he served as a research assistant and lecturer, also co-lecturing with Donald Knuth and helping him write the second volume of The Art of Computer Programming and the second edition of Seminumerical Algorithms.

Between the years 1977-1981 he did researches on graphics and mathematics with SmallTalk, Mesa and Lisp. Apart from this, he found the first space-time compressor now known as Hashlife. Hashlife is a computer algorithm where deterministic cellular automata is simulated.

In 1982, Gosper wrote a fancy Remez algorithm for optimal univariate approximation at the Lawrence Livermore Labs for S-1 Project. Again for the Lawrence Livermore Labs, he contributed to the acceleration formulae for their laser physics codes which were not published.

Afterwards, Bill Gosper worked at the Symbolics Inc. as a researcher until 1988, and did researches on experimental mathematics, numeric and symbolic algorithms and mathematical graphics. Apart from this, he worked on mathematical graphics animations, fractal algorithms, acceleration methods of various functions, random number generators and also on R&D projects on Symbolics Lisp arithmetic. Additionally, published several papers, gave several invited conference talks and corresponded with about two dozen mathematicians in related fields, wrote and presented MACSYMA demos and attained the championship of filing most MACSYMA bugs. One year following 1988, he served as a consultant to Symbolics Inc. and continued his previous R&D projects. From 1989 to 1992 Gosper consulted Wolfram Research, Inc. and Symbolics MACSYMA Group and had been the Senior Member of Technical Staff at Macsyma Inc. between 1992 and 1999.

After this point of his business life, not much news from Gosper were obtained, but gratifyingly, Robert Smith, whom Gosper occasionally chatted to during one of his visits to Facebook, states: "Gosper is still that very curious, tinker, thinker and hacker person. He always carries with himself some little games and puzzles, and loves facing people with these small challenges. We hack something together every time we encounter with each other. For instance, we once booted a dusty Alpha 1U machine that had OpenGenera installed on it. Our next target was making MACSYMA work on an old Lisp machine." You can challenge yourself with some nice challenges that Gosper, a packing problem master filled his own blog with!

Hoping to *hack the life itself* freely in the Utopia Gosper and his friend's hackerism intended to create!

Mert Susur • mail@mertsusur.com

TECHNICAL CONSTRAINTS OF THE BLOCKCHAIN PROJECTS

Data Storage Problems and Solutions

Blockchain technology is widely discussed around the world as it is in Turkey.

We've been through the most popular times of blockchain technology in the last ten months. The biggest reason for this was undoubtedly the emergence of many different technologies, and the value of the cryptocurrencies having high volatility on the exchange market. In other words, we can say that a lot of the people who are aiming for consumption have invested in this business and their win/loss results have also pursued this popularity.

Hundreds of people, groups or companies are now looking for applying this technology to their businesses. Even when you attend many conferences or meetups, especially in our country, everybody talks about how wonderful this technology is and how this technology will bring benefits to the world. At this point, I want you to ask yourselves, how many applications are you using in your daily life now that is taking advantage of blockchain technology? Or is this a wonderful technology that has a range of different application areas without a problem or an issue? I mean, why do so many companies make improvements into this field and are not able to present final products for you just yet?

Yes, I will be talking about the bad parts of this technology. Because as a software engineer I know that if your only tool in your toolbelt is a hammer, you would probably start seeing everything as the nail! Therefore, understanding what you can not do with that hammer helps you choose the right tool to do your job and understanding the potential problems that you might have. At this point, I want to quote from the Javascript Guru, Douglas Crockford. He mentions in his book, Javascript: The Good Parts, that 'Most programming languages contain good parts. I discovered that I could be a better programmer by using only the good parts and avoiding the bad parts.' You need to spend your days, perhaps even your nights to learn more about blockchain and its applications. This will let you get into the awful details that other people are afraid to talk about. In this article, I will focus primarily on the data storage problems, and I will try to explain the different issues and their potential solutions in the future articles.

Data storage issues

Blockchain is designed as a database that is meant to store the transactions between parties, without requiring any trusted relationship between peers. In other words, it aims to provide a decentralized, and immutable data storage to transfer information without the necessity of trust.

Even assuming that the first design is the article of Satoshi Nakamoto, if a block header that does not contain any data will occupy 80 Bytes and it is assumed that a block will be produced every 10 minutes, it will be calculated to store 4.2MB data in one year. Of course, Bitcoin's database size has reached around 160,124 MB since 2008. Besides, Bitcoin is not designed to store data. Because it is intended to be an electronic money system which can be understood from its name, this technology can realize the actual aim value transfers.

If we take a look at Ethereum from the other side, we will see grave technical differences. To remind you briefly without going into the details, Ethereum is a platform that you can develop your decent contracts and develop your decentralized applications. If you want to install and use your software within this platform, or if you're going to use apps and transfer values by others in a similar way, you have to pay a fee for each transaction you make. This price is called gas on behalf of the material property independent. This gas value is multiplied by a unit called gas price and is distributed as a reward to the miners. For example, if you need to pay 5 gas, and if the gas price is 10 Ether, the amount you need to pay for this is 50 Ether. In this case, the gas can be described as the provision of the miners' troubles in the simplest form. What kind of trouble do the miners get in?

We know the importance of miners in the blockchain technology, and if we talk specifically about Ethereum, another thing the miners do is realize smart contract calls in the block. If smart contracts are doing data storage at this time, the miners will process the data in the final state of the contract, and they will receive the rewards.

In the Ethereum Yellow Paper, written by Gavin Wood, this gas issue is taken seriously by the mathematical model. Now let's extract the cost of storing the data according to this document. The document says that the cost of storing a 256-bit word value is 20 gas. So with a simple calculation, we can see that the price of storing 1 KB of data will be about 640,000 gas, and today we can see that if we accept an average gas price of 80 gwei (0.00000002 Ether), we would not have to pay 0.512 Ether to store 1 KB of data. Of course, this is not sustainable.

So if I dream of developing applications using the Ethereum network in my project, and if I imagine keeping all of my data in the blockchain for having a decentralized system, obviously I'll have to take a serious investment out of sight.

Blockchain data storage solutions

I know I'm drawing a terrible picture up there, but please be patient. I have good news and bad news for you. The good news is, do not worry; there are a few cool solutions for data storage issues. The bad news is that none of them are as mature enough to work on production systems as of the date this article was written.

Some of those; Filecoin, Sia, StorJ, and Swarm. I will try to give you some details about Swarm, which I have mastered most, as I review it closely and will start to develop the project soon.

What is Swarm?

The Ethereum community has many different projects. These are called the Web 3 Stack. For example, Whisper: a decentralized end-to-end encrypted messaging platform, is working on a blockchain that allows instant value transfers. We will talk about them again at another time, but I want to continue with Swarm right now.

The purpose of the Swarm project is to provide a decentralized data storage solution to the new world order. There are also many security issues within this project. DDoS attacks and well-designed access restrictions are the most important problems for us. It is essential that if the different attack vectors are realized, the network continues to provide files and services without any issues. The Swarm team is also working on PoC (Proof of Concept) to prove that they can provide these features with regular iterations. They will publish their fourth PoC in the second quarter of 2018, and in this version, they will have a lot of interesting features.

I feel like I hear you ask how it works. I'll start to explain this in a moment, but before that, I have something significant to say. At the time this article was written, the features that exist in the first quarter of 2018 may or may not depend on the time you read. In fact, the working method may have changed completely. So before you begin using this product, it is critical that you start by reading the documentation.

Every node running on the network has its address. It may be possible to access different instances using these addresses. So you can reach a client if you know the address and you can make a P2P connection to access your files. The files you send to the Swarm network reach other clients through a subroutine called the bzz protocol. This protocol is designed to work through any Ethereum client (e.g., ethereum geth) as we use the Ethereum protocol Devp2p protocol, which we also know. The PoC 0.2 version is already available in the geth.

Let's say I want to send a file when I choose to send this file; the client will split the file into smaller chunks which will be up to 4K and generate hash values by passing each of them through a hash function. So they have a tree in their hands that they can create a Merkle Constitution. This prevents you from accessing any branch of the tree at random. In other words, you can not recreate the file in a meaningful way without knowing the whole tree. But if you know which order you can access the small pieces, you can jump between them. With this feature, it will also be relatively easy to broadcast video or jump between video frames. However, the type of video codec you use here is also of special importance.

But one thing we need to be aware of is that this data can never be changed and deleted. So Swarm is not an environment where you can store and edit data like your local disk. Instead, it is a platform that you will upload and store your data forever.

The storage of videos, pictures or documents are best areas of usage for Swarm from my standpoint. However, it is useful to give a little warning: Swarm does not yet encrypt the data as of the current version. So it is not right to hide private or sensitive information on this platform. The team's future plans show that they will be allowed to store data either privately or globally, but the platform is not yet ready for it.

There is no economic model to store data here because the platform is not completed. However, two possible income models are mentioned, the services to be carried out for the data to pass, the services can receive a prize in proportion to the amount they carry, or the data storers can earn an award for as much as they keep. But that's all assumptions. Nothing is clear yet. But in the future instead of mining and digging out your USB disks then storing the data seems to be able to get the Ether.

CALL FOR PAPERS

ARKA|KAPI

Do you want your article to be published on Arka Kapi Magazine? Submit now to be featured in the next issue! Your article can be of any title as long as it fits to the cyber security context. Make sure it's an original article that isn't previously published elsewhere.

Email your articles to:
editor@arkakapimag.com

FEEDBACK

Got any feedback about Arka Kapi Magazine? Found a bug? Want us to add or remove something? Let us know!

follow us

Don't miss the news!

arkakapimag