

A screenshot of a web browser window titled "U2U Online". The URL in the address bar is "online.u2u.be/courses/25896/modules/2/24840". The main content area displays a course titled "Encryption, Hashing and Signing". On the left, a sidebar lists course modules: "Encryption, Hashing and Signing" (selected), "Why do we need encryption?", "Alice and Bob's house", "Renewing Cryptography Keys", "Public Key Exchange", "Asymmetric key encryption", "Hybrid Encryption", "Encryption vs. Decoding", "Sharing Passwords", "Homing", and "Public Signing". On the right, the "u2u" logo is displayed. The bottom of the screen shows a Windows taskbar with icons for File Explorer, Task View, Start, and other system functions, along with the date and time (12/29/2023) and battery status.

A screenshot of a Microsoft Edge browser window. The address bar shows the URL 'online.u2u.be/courses/25896/modules/2/24840'. The main content area displays a slide titled 'Agenda' with a red horizontal line below it. The agenda lists ten items in a bulleted list: 'Why do we need encryption, and what does it mean?', 'Encryption building blocks', 'Generating Cryptography keys', 'Public Key Exchange', 'Symmetric Key Algorithms', 'Asymmetric Key Algorithms', 'Hybrid Encryption', 'Storing Passwords safely', 'Hashing', and 'Digital Signing'. To the right of the agenda is a large image of a vintage cryptographic machine, specifically an Enigma machine, shown from a top-down perspective. The machine is black with numerous circular keys and internal mechanical components visible. The browser interface includes a sidebar on the left with user information and course navigation, and a dark theme header with the 'u2u Online' logo.

Agenda

- Why do we need encryption, and what does it mean?
 - Encryption building blocks
 - Generating Cryptography keys
 - Public Key Exchange
 - Symmetric Key Algorithms
 - Asymmetric Key Algorithms
 - Hybrid Encryption
 - Storing Passwords safely
 - Hashing
 - Digital Signing



Why do we need encryption?

- **Non-disclosure**: protect data from being accessed by unauthorized people
 - Financial, personal, medical, military (!), etc...
- **Tampering**: to protect data from being modified, again by unauthorized people

26°C Mostly sunny 3:08 PM 12/29/2023

u2u U2U Online

online.u2u.be/courses/25896/modules/2/24840

Developer and IT Training

Demo

Traceroute/tracert demo

Run tracert www.google.com

26°C Mostly sunny

Search

3:08 PM

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page content is a slide from a presentation. At the top right is the **u2u** logo. Below it, the text "Developer and IT Training" is displayed in red. A large red word "Demo" is centered on the slide. Below "Demo", the text "Traceroute/tracert demo" and "Run tracert www.google.com" are shown in gray. On the left side of the slide, there is a vertical navigation menu with several items listed. The bottom of the slide has a footer bar with various icons. The browser's address bar and toolbar are visible at the top and bottom of the window respectively.

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "What does encryption mean?". On the left, there's a sidebar with a navigation tree under "Encryption, Hashing and Signing". The main content area contains a bulleted list explaining encryption:

- Encryption: **"Make data unreadable for third parties"**
- Decryption: **"Make readable again"**
- Technically: **plaintext** becomes (seemingly) pure random data (**ciphertext**) which can become readable again using the correct **cryptographic key**

The browser interface includes a top bar with tabs, a search bar, and various icons. The bottom taskbar shows system status and a date/time indicator.

Substitution Cypher

- Replace each letter with another one

The diagram illustrates the ROT13 substitution cipher. It shows two rows of the English alphabet. The top row contains letters A through M, and the bottom row contains letters N through Z. Arrows between corresponding letters in the two rows indicate a shift of 13 positions. This shift is labeled "ROT13".

Below this, two examples demonstrate the ROT13 encryption process:

- The word "HELLO" is encrypted to "URYYB". The letters are shifted by 13 positions: H (blue) to U (blue), E (orange) to R (orange), L (green) to Y (green), L (green) to Y (green), and O (pink) to B (pink).
- The word "URYYB" is decrypted back to "HELLO". The letters are shifted back by 13 positions: U (blue) to H (blue), R (orange) to E (orange), Y (green) to L (green), Y (green) to L (green), and B (pink) to O (pink).

13

A	B	C	D	E	F	G	H	I	J	K	L	M
---	---	---	---	---	---	---	---	---	---	---	---	---

ROT13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
---	---	---	---	---	---	---	---	---	---	---	---	---

13

H	E	L	L	O
---	---	---	---	---

ROT13

U	R	Y	Y	B
---	---	---	---	---

26°C
Mostly sunny

Search

3:08 PM 12/29/2023

u2u U2U Online

online.u2u.be/courses/25896/modules/2/24840

Home > Data Security Fundamentals Techniques > Transposition cipher

Encryption, Hashing and Signing
2. Symmetric
Transposition cipher

Encryption, Hashing and Signing
Why we need encryption?
Everywhere building blocks
Generating a symmetric key
Public Key Exchange
Symmetric Key Encryption
Asymmetric Key Encryption
Hybrid Encryption
Encryption vs. Processing
Sharing Passwords
Hashing
Digital Signing
Summary

Transposition cipher

u2u

- Letters in **plaintext** are shifted according to a regular system, so that the **ciphertext** constitutes a **permutation** of the plaintext

6 3 2 4 1 5
W E A R E D
I S C O V E
R E D F L E
E A T O N C
E Q K J E U

EVLNE ACDTK ESEAQ ROFOJ DEECU WIREE

26°C Mostly sunny

Search

3:08 PM 12/29/2023

Modern Cipher

- Combination of Substitution, Transposition, Expansion and Merging of bits
- Example: Simplified DES

```
graph LR; A[64 bit Plain Text] --> B[Initial Permutation]; B --> C[Left Plain Text]; B --> D[Right Plain Text]; C --> E[16 Round (ENC)]; D --> E; E -- key --> F[16 Round (ENC)]; F --> G[Final Permutation]; G --> H[64 bit Cipher Text];
```

The diagram illustrates the Simplified DES cipher process. It starts with a 64-bit plain text input, which undergoes an initial permutation. This splits the text into two 32-bit halves: Left Plain Text and Right Plain Text. Each half passes through 16 rounds of encryption (ENC). A key is used to control both encryption processes. Finally, the two halves are combined through a final permutation to produce the 64-bit cipher text.

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Key generation". The left sidebar shows a navigation tree for "Encryption, Hashing and Signing" and "2.0 Cryptography". The main content area contains a bulleted list about key generation, with a red link to a YouTube video. The browser has a dark theme, and the system tray at the bottom shows weather, search, and system icons.

Key generation

- The process of generating keys in cryptography, using a key generator (**keygen**)
- Keys are large integers
 - Generated using a **Random Number Generator**
 - Generated by deriving it from a passphrase and a **key derivation function**
 - Or use Lava lamps:
<https://www.youtube.com/watch?v=1cUUfMeOijg&feature=youtu.be>
- One way to decrypt data is by using a brute force attack
 - Keys have to be sufficiently long

Key Derivation Function

- Important for generating a key is a sufficiently long passphrase
 - Use salt, which is a random number, to prevent dictionary attacks
 - Use iterations
- Derivation functions are slow on purpose as to frustrate brute force attacks
- Most used today is PBKDF2 (**Password-Based Key Derivation Function 2**)
 - <https://en.wikipedia.org/wiki/PBKDF2>
- PBKDF2 is implemented in .NET using the **Rfc2898DeriveBytes** class

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Exchanging keys". The left sidebar contains a navigation tree with categories like "Encryption, Hashing and Signing", "Public Key Exchange", "Hybrid Encryption", "Asymmetric key encryption", "Symmetric key encryption", "Generating a symmetric key", "Generating a public key", "Why we need encryption?", "Encrypting a file", and "Summary". A user profile icon is visible at the top left. The right side features a large "u2u" logo. The main content area discusses "Public Key exchange" with sub-sections "Direct exchange", "PKIX", and "OpenPGP", each with bullet points describing their characteristics.

Exchanging keys

- Public Key exchange
 - Direct exchange
 - Public key is sent in a simple way
 - Verification is performed manually, for example by comparing the checksum provided by key owner
 - Example: SSH
 - PKIX
 - Public key is embedded in a X.509 certificate, signed by trusted certification authority
 - Example: HTTPS, S/MIME
 - OpenPGP
 - Allow everyone to sign keys
 - At the enterprise level this involves tools to ensure key ownership (e.g. LDAP)
 - Example: PGP, GnuPG

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Symmetric Key Encryption". The left sidebar contains a navigation tree for "Encryption, Hashing and Signing" and "Digital Signatures". The main content area has a red horizontal line separating the title from the text. The text lists pros and cons of symmetric key encryption. The bottom of the screen shows a taskbar with various icons and a system tray.

Symmetric Key Encryption

- Secret (shared) key
- The same key is used for encryption and decryption of content

- Pro:
 - Fast
 - Simple
- Con:
 - Less secure
 - Shared key must be stored and exchanged in a secure way

26°C Mostly sunny 3:08 PM 12/29/2023

u2u U2U Online x +

online.u2u.be/courses/25896/modules/2/24840

Manage Activities

Encryption, Hashing and Signing

COURSES

Your interests

Create a new course

- How do we need encryption?
- How can we need encryption?
- How can we need hashing?
- Generating Cryptographic Keys
- Public Key Exchange
- Symmetric Key Encryption
- Asymmetric Key Encryption
- Hybrid Encryption
- Generating Passwords
- Hashing
- Metric Logging
- Recovery

u2u

Symmetric Key Encryption

The diagram illustrates the process of symmetric key encryption. On the left, a blue box labeled "Alice (Sender)" contains a "Shared key" represented by two blue horizontal bars. Below it is a "TXT" file icon. A large black arrow points from the "Shared key" down to a green box labeled "Encryption". To the right, a red box labeled "Bob (Recipient)" contains a yellow key icon and a "Shared key" represented by two blue horizontal bars. Below it is a "Decryption" box. A large black arrow points from the "Shared key" down to the "Decryption" box. Between the two boxes is the text "Bob and Alice share the same key". Below the "Encryption" box is a sample of binary "Cypher text".

Alice (Sender)

Shared key

Encryption

Bob and Alice share the same key

Cypher text

Bob (Recipient)

Shared key

Decryption

TXT

TXT

26°C
Mostly sunny

Search

3:08 PM
12/29/2023

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Symmetric Key Implementations". The left sidebar contains a navigation tree for a course module, and the right sidebar features a large "u2u" logo. The main content area lists several bullet points about symmetric key algorithms and their implementation.

Symmetric Key Implementations

- Symmetric Key Algorithms
 - Popular algorithms: Twofish, Serpent, Rijndael, Blowfish, RC4, ...
 - Standardized algorithm: **AES (Advanced Encryption Standard – aka Rijndael)**
 - In .NET use the **Aes** class
- When to use it
 - Encrypting a bigger stream of data
 - Encrypting communication between two parties when the key is securely agreed upon
- Notable algorithms during history
 - ROT13 (rotate by 13 places, special form of Caesar cipher)
 - ENIGMA
 - DES

26°C Mostly sunny 3:08 PM 12/29/2023

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Asymmetric Key Encryption". The left sidebar contains a navigation tree for a course module, with the current section being "Asymmetric Key Encryption". The main content area displays a bulleted list of points about asymmetric key encryption. The browser interface includes a search bar, a toolbar with various icons, and a system tray at the bottom showing weather, battery, and system status.

Asymmetric Key Encryption

- Key pair (mathematically related)
 - Public key
 - Available to anyone
 - Private key
 - Secret, only you know it
- Impossible to derive one key from the other
- Encrypt with public key, decrypt with corresponding private key
- Pro:
 - More secure
- Con:
 - Slower
 - Requires more processing power

Asymmetric Key Encryption

The diagram illustrates the process of Asymmetric Key Encryption. It is divided into two main sections: Alice (Sender) on the left and Bob (Recipient) on the right.

Alice (Sender):

- Contains a blue box labeled "Alice (Sender)".
- Shows two blue key icons: "Private key" and "Public key".
- Shows one orange key icon: "Public key Bob".
- An arrow points from the "Public key Bob" icon down to a green box labeled "Encryption".
- Below "Encryption" is a document icon labeled "TXT".

Bob (Recipient):

- Contains a red box labeled "Bob (Recipient)".
- Shows two orange key icons: "Private key" and "Public key".
- An arrow points from the "Public key" icon down to a red box labeled "Decryption".
- Below "Decryption" is a document icon labeled "TXT".

Middle:

- A vertical column of binary code is labeled "Cypher text".
- A small triangle icon is positioned above the binary code.

Left Sidebar:

- Contains a user profile icon and the name "Maartje Kuijper".
- A navigation menu titled "Encryption, Hashing and Signing" with several items listed.

Bottom Bar:

- Shows the weather: "26°C Mostly sunny".
- Includes a search bar, a building icon, and several application icons (File Explorer, Task View, Edge, Google Chrome).
- Shows system status: "3:08 PM", "ENG", battery level, and a notification bell.

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Asymmetric Key Implementations". The left sidebar contains a navigation tree for a course module, and the right sidebar features the "u2u" logo. The main content area lists several bullet points about asymmetric key implementations.

Asymmetric Key Implementations

- Asymmetric Key Algorithms
 - Diffie Hellman, RSA, ElGamal,...
 - Most known: **RSA**
 - In .NET use the **RSACryptoServiceProvider** class
- Protocols
 - PGP (Pretty Good Privacy) or GPG (Gnu Privacy Guard)
 - Bitcoin
 - SSH
 - IKE (Internet Key Exchange, for IPSec)
- Use this for sending small messages like communicating a shared key

26°C Mostly sunny 3:09 PM ENG 12/29/2023

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Hybrid Encryption". The left sidebar contains a navigation tree for a course module, with "Hybrid Encryption" selected. The main content area displays a list of bullet points about hybrid encryption:

- Most applications combine the use of symmetric and asymmetric encryption
 - Generate a symmetric key
 - Protect the symmetric key with public key encryption
 - Send the encrypted symmetric key to the recipient
 - Encrypt/decrypt data with symmetric key
- Secure and fast
- Implementation
 - TLS (Transport Layer Security)

The browser interface includes standard navigation buttons, a search bar, and a toolbar at the top. The taskbar at the bottom shows various open applications and system status icons.

U2U Online

online.u2u.be/courses/25896/modules/2/24840

Hybrid Encryption

The diagram illustrates the Hybrid Encryption process between Alice (Sender) and Bob (Recipient).

Alice (Sender):

- Has a **Public key** and a **Private key**.
- Uses the **Public key** to generate a **Shared key**.
- Encrypts the **TXT** file using the **Shared key**.
- Encrypts the **Shared key** using the **Private key**.
- Sends the **Cypher text** and the **Encrypted Shared key** to Bob.

Bob (Recipient):

- Has a **Public key** and a **Private key**.
- Decrypts the **Encrypted Shared key** using the **Private key**.
- Uses the **Public key** to generate a **Shared key**.
- Decrypts the **Cypher text** using the **Shared key**.
- Receives the **TXT** file.

Encryption vs. Encoding

- Encoding != Encryption
- For example: ROT

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

MNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyzABCDEGHIJKL

- Base64 Encoding

26°C Mostly sunny

Search

3:09 PM 12/29/2023

u2u U2U Online

online.u2u.be/courses/25896/modules/2/24840

Home > Web Security Fundamentals Techniques > Principles

Encryption, Hashing and Signing

3. Games

Encryption, Hashing and Signing

Why we need encryption?

Encryption building blocks

Generating a symmetric key

Public Key Exchange

Symmetric Key Encryption

Asymmetric Key Encryption

Hybrid Encryption

Encryption vs. Decryption

String Passwords

Hashing

Digital Signatures

Summary

```
/// <summary>
/// Function is used to encrypt the password
/// </summary>
/// <param name="password"></param>
/// <returns></returns>
private string Encryptdata(string password)
{
    string strmsg = string.Empty;
    byte[] encode = new byte[password.Length];
    encode = Encoding.UTF8.GetBytes(password);
    strmsg = Convert.ToBase64String(encode);
    return strmsg;
}
/// <summary>
/// Function is used to Decrypt the password
/// </summary>
/// <param name="password"></param>
/// <returns></returns>
private string Decryptdata(string encryptpwd)
{
    string decryptpwd = string.Empty;
    UTF8Encoding encodepwd = new UTF8Encoding();
    Decoder Decode = encodepwd.GetDecoder();
    byte[] todecode_byte = Convert.FromBase64String(encryptpwd);
    int charCount = Decode.GetCharCount(todecode_byte, 0, todecode_byte.Length);
    char[] decoded_char = new char[charCount];
    Decode.GetChars(todecode_byte, 0, todecode_byte.Length, decoded_char, 0);
    decryptpwd = new String(decoded_char);
    return decryptpwd;
}
```

share | edit | flag

edited 4 hours ago

answered 4 hours ago

Hitesh 61 5

W A T F 4

26°C Mostly sunny

Search

3:09 PM 12/29/2023

u2u U2U Online

online.u2u.be/courses/25896/modules/2/24840

Encryption, Hashing and Signing
3. Games

This is somewhat simple

```
string inp = "hai";
StringBuilder strb = new StringBuilder();
foreach (char s in inp)
{
    int sin = s + 5;
    char newch = (char)sin;
    strb.Append(newch);
}
string output = strb.ToString();
```

Now the output contains the encrypted string "mfn" (ie., 5 letters away from the original)in it....

share | edit | flag

answered 4 hours ago
kovilpatti C sharper
93 ● 7

26°C Mostly sunny

Search

3:09 PM 12/29/2023

u2u

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Storing Passwords". The left sidebar contains a navigation menu with sections like "Encryption, Hashing and Signing", "Public Key Exchange", "Hybrid Encryption", "Hashing", and "Digital Signing". The main content area displays a bulleted list:

- Cryptography in web sites is mainly used for storing secrets
 - Passwords
 - Credit card information
 - ...
- Three major ways for storing passwords
 - Plaintext (no protection)
 - Encrypted (allows for the original password to be retrieved)
 - Hashed (one-way, only allows for checking if two strings are the same)

The bottom of the screen shows a taskbar with various icons and a system tray indicating the date and time as 12/29/2023 at 3:09 PM.

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "What is hashing?". On the left, there's a sidebar with a navigation tree under "Encryption, Hashing and Signing". The main content area contains a bulleted list about hashing, followed by three red-highlighted hash examples. The browser has a dark theme, and the u2u logo is visible in the top right corner.

What is hashing?

- Hash functions are one way functions
 - Turning any amount of data into a fixed-length “fingerprint”, that cannot be reversed!
 - Have the property that when the input changes even a tiny bit, the hash is completely different
- hash("hello") = 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
hash("hblllo") = 58756879c05c68dfac9866712fad6a93f8146f337a69afe7dd238f3364946366
hash("waltz") = c0e81794384491161f1777c232bc6bd9ec38f616560b120fda8e90f383853542
- Great way (and the only way) to store passwords
 - When the user attempts to login, the password is hashed and compared to the one in storage
 - If the hashes match, the user is allowed in
- Only use special cryptographic hashing functions!
 - These have been optimized for security, not speed!

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "How hashes are cracked". On the left, there's a sidebar with a navigation tree under "Encryption, Hashing and Signing". The main content area contains a bulleted list of cracking methods:

- **Brute force attack:** simply try all combinations up to a given length
 - That is why passwords should be long enough so it will take too long
 - Use passphrases!
- **Dictionary attack:** simply try all likely combinations from a “dictionary”
 - Dictionaries are built by extracting words from large bodies of text
 - Extended by common equivalents, e.g. s3cr3t (substituting E by 3)
- **Lookup tables:** pre-compute the hashes from the dictionary
 - Then try to find the database entry in the lookup table (very fast!)
<https://crackstation.net/>
- **Reverse lookup tables:** match same hashes to list of users
 - It is quite common for users to accidentally pick the same password (e.g. 1234)
- **Rainbow tables:** optimized tables to hold more hashes
 - Rainbow tables holding all md5 hashed 8 character passwords exists!

U2U Online

online.u2u.be/courses/25896/modules/2/24840

Encryption, Hashing and Signing

1. Concepts

2. Hashing

3. Cryptography

4. Cryptographic Hash Functions

5. Hashcat

6. Hashcat Performance

7. Hashcat vs. Hashcat

8. Hashcat vs. Hashcat

9. Hashcat vs. Hashcat

10. Hashcat vs. Hashcat

11. Hashcat vs. Hashcat

12. Hashcat vs. Hashcat

13. Hashcat vs. Hashcat

14. Hashcat vs. Hashcat

15. Hashcat vs. Hashcat

16. Hashcat vs. Hashcat

17. Hashcat vs. Hashcat

18. Hashcat vs. Hashcat

19. Hashcat vs. Hashcat

20. Hashcat vs. Hashcat

21. Hashcat vs. Hashcat

22. Hashcat vs. Hashcat

23. Hashcat vs. Hashcat

24. Hashcat vs. Hashcat

25. Hashcat vs. Hashcat

26. Hashcat vs. Hashcat

27. Hashcat vs. Hashcat

28. Hashcat vs. Hashcat

29. Hashcat vs. Hashcat

30. Hashcat vs. Hashcat

31. Hashcat vs. Hashcat

32. Hashcat vs. Hashcat

33. Hashcat vs. Hashcat

34. Hashcat vs. Hashcat

35. Hashcat vs. Hashcat

36. Hashcat vs. Hashcat

37. Hashcat vs. Hashcat

38. Hashcat vs. Hashcat

39. Hashcat vs. Hashcat

40. Hashcat vs. Hashcat

41. Hashcat vs. Hashcat

42. Hashcat vs. Hashcat

43. Hashcat vs. Hashcat

44. Hashcat vs. Hashcat

45. Hashcat vs. Hashcat

46. Hashcat vs. Hashcat

47. Hashcat vs. Hashcat

48. Hashcat vs. Hashcat

49. Hashcat vs. Hashcat

50. Hashcat vs. Hashcat

51. Hashcat vs. Hashcat

52. Hashcat vs. Hashcat

53. Hashcat vs. Hashcat

54. Hashcat vs. Hashcat

55. Hashcat vs. Hashcat

56. Hashcat vs. Hashcat

57. Hashcat vs. Hashcat

58. Hashcat vs. Hashcat

59. Hashcat vs. Hashcat

60. Hashcat vs. Hashcat

61. Hashcat vs. Hashcat

62. Hashcat vs. Hashcat

63. Hashcat vs. Hashcat

64. Hashcat vs. Hashcat

65. Hashcat vs. Hashcat

66. Hashcat vs. Hashcat

67. Hashcat vs. Hashcat

68. Hashcat vs. Hashcat

69. Hashcat vs. Hashcat

70. Hashcat vs. Hashcat

71. Hashcat vs. Hashcat

72. Hashcat vs. Hashcat

73. Hashcat vs. Hashcat

74. Hashcat vs. Hashcat

75. Hashcat vs. Hashcat

76. Hashcat vs. Hashcat

77. Hashcat vs. Hashcat

78. Hashcat vs. Hashcat

79. Hashcat vs. Hashcat

80. Hashcat vs. Hashcat

81. Hashcat vs. Hashcat

82. Hashcat vs. Hashcat

83. Hashcat vs. Hashcat

84. Hashcat vs. Hashcat

85. Hashcat vs. Hashcat

86. Hashcat vs. Hashcat

87. Hashcat vs. Hashcat

88. Hashcat vs. Hashcat

89. Hashcat vs. Hashcat

90. Hashcat vs. Hashcat

91. Hashcat vs. Hashcat

92. Hashcat vs. Hashcat

93. Hashcat vs. Hashcat

94. Hashcat vs. Hashcat

95. Hashcat vs. Hashcat

96. Hashcat vs. Hashcat

97. Hashcat vs. Hashcat

98. Hashcat vs. Hashcat

99. Hashcat vs. Hashcat

100. Hashcat vs. Hashcat

Typical Cracking Hardware

- Lots of GPUs
- Google
 - Hashcat performance

http://the password project.com/oclhashcat_benchmarking



u2u U2U Online

online.u2u.be/courses/25896/modules/2/24840

Developer and IT Training

Demo

Using Google to crack hash

21bd12dc183f740ee76f27b78eb39c8ad972a757

26°C
Mostly sunny

Search

3:10 PM 12/29/2023

Rubbing salt into the wounds

- Salt is a random string added to the hash
 - Doesn't even have to be stored securely
- Helps to protect against lookup table attacks
 - These only work because each password is hashed the exact same way
 - With salt: same password results in different hash

```
hash("hello") = 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
hash("hello" + "QxLUF1bgIAdeQX") = 9e209040c863f84a31e719795b2577523954739fe5ed3b58a75cff2127075ed1
hash("hello" + "bv5PehSMfV11Cd") = d1d3ec2e6f20fd420d50e2642992841d8338a314b8ea157c9e18477aaef226ab
hash("hello" + "YYLmfY6IehjZMQ") = a49670c3c18b9e079b9cfaf51634f563dc8ae3070db2c4a8544305df1b60f007
```

- Common mistakes
 - **Using the same salt:** same salt => same password has same hash!
 - **Short salt:** hacker simply builds a larger lookup table
 - Rule of thumb: use salt with same length as hash output length

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Bruce-forcing hashes with salt". The left sidebar shows a navigation tree under "Encryption, Hashing and Signing". The main content area contains a bulleted list about hash cracking:

- Salt makes it a lot harder to “crack” hashes but not impossible
 - It’s always a matter of time
- If you can generate hashes fast enough
 - Modern GPUs are very good at generating hashes
 - Easily more than **7.5 billion hashes per second** on consumer hardware
 - And with dictionaries you can increase hit-count

The browser interface includes a search bar, a toolbar with various icons, and a system tray at the bottom showing weather (26°C, Mostly sunny), a date (12/29/2023), and a battery level.

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "How to hash properly". The left sidebar contains a navigation menu with sections like "Encryption, Hashing and Signing", "Public Key Exchange", "Symmetric Encryption", "Asymmetric Encryption", "Hybrid Encryption", "Encryption vs. Hashing", "Hashing", and "Digital Signing". A "Summary" section is also present. The right sidebar features a large "u2u" logo. The main content area lists several bullet points about hashing best practices.

How to hash properly

- Generate salt using a **cryptographically secure random number generator**
 - In .NET use the **RNGCryptoServiceProvider** class
- Salt needs to be unique per user/per password!
 - Never re-use a salt
- Salt needs to be **long**, best as long as the hash output
 - Stored in the table next to the hash
- Use a **slow** hash function, slow enough, but not too slow to annoy users
 - Also consider that a slow hash function can be exploited as a DDoS attack...
- Which hash function?
 - Don't use the fast hash MD5, SHA1, SHA256, SHA512,
 - Use **PBKDF2**

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Digital Signing". The left sidebar lists course modules: "Encryption, Hashing and Signing", "2. Topics", "Digital Signatures", and "Digital Signing". The main content area contains a bulleted list describing the digital signing process:

- **Sender**
 - Based on message content a one-way hash is generated
 - Hash is encrypted with sender's private key
 - Encrypted hash is added to message
- **Recipient**
 - Extracts encrypted hash
 - Decrypts hash with sender's public key
 - Regenerates hash
- **Checks:**
 - If hashes match, content wasn't modified
 - If hash can be decrypted, the sender is verified successfully

The browser interface includes a search bar, a toolbar with various icons, and a system tray at the bottom showing weather (26°C, Mostly sunny), system status (ENG, battery level, signal strength), and the date/time (3:10 PM, 12/29/2023).

U2U Online

online.u2u.be/courses/25896/modules/2/24840

Digital Signing

The diagram illustrates the digital signing process between Alice (Sender) and Bob (Recipient).

Alice (Sender):

- Has a **Public key** and a **Private key**.
- Performs **Encryption** on a **TXT** file.
- The result is a **Signed doc**, which contains the original **TXT** file and a **Hash**.

Bob (Recipient):

- Has a **Public key** and a **Private key**.
- Uses Alice's **Public key** to perform **Decryption** on the **Signed doc**.
- The decrypted result is compared to the original **TXT** file, showing they are equal.

u2u

The screenshot shows a web browser window with the URL online.u2u.be/courses/25896/modules/2/24840. The page title is "Summary". On the left, there's a sidebar with a navigation tree for "Encryption, Hashing and Signing" and a "Summary" link. The main content area contains a bulleted list under the heading "Summary".

Summary

- **Encryption** allows you to make it harder to discover secrets
 - Encryption is two way
 - Protects against disclosure
- **Hashing** allows you to see if the data is the same
 - Hashing is one way, you cannot get the data back
 - Protects against tampering
- **Digital Signatures** allow you to see if the data has been tampered with
 - Protects against non-repudability

At the bottom of the screen, there's a taskbar with icons for weather (26°C, Mostly sunny), search, file explorer, and various application icons. The system tray shows the date (12/29/2023) and time (3:10 PM).