

Cyber security and cloud computing

1. Implementation of S-DES

```
INPUT:
# Simplified S-DES Example (for educational purposes)

# Initial and final permutations
IP = [1, 5, 2, 0, 3, 7, 4, 6]
IP_inv = [3, 0, 2, 4, 6, 1, 7, 5]

# Key generation - P10, split, shifts, P8
P10 = [2, 4, 1, 6, 3, 9, 0, 8, 7, 5]
P8 = [5, 2, 6, 1, 7, 3, 9, 0]

# S-boxes (substitution)
S0 = [[1, 0, 3, 2], [3, 2, 1, 0], [0, 2, 1, 3], [3, 1, 3, 2]]
S1 = [[0, 1, 2, 3], [2, 0, 3, 1], [3, 0, 1, 2], [2, 1, 0, 3]]

# Example key and plaintext (in binary)
key = '1010000010' # 10-bit key
plaintext = '01101110' # 8-bit plaintext

# Function for applying permutation
def permute(bits, permutation):
    return ''.join(bits[i] for i in permutation)

# Function for key generation
def key_generation(key):
    key = permute(key, P10) # Apply P10 permutation
    left, right = key[:5], key[5:]

    # Left shift 1 and apply P8 to get K1
    left, right = left[1:] + left[0], right[1:] + right[0]
    K1 = permute(left + right, P8)
```

```

# Left shift 2 and apply P8 to get K2
left, right = left[1:] + left[0], right[1:] + right[0]
left, right = left[1:] + left[0], right[1:] + right[0]
K2 = permute(left + right, P8)

return K1, K2

# Function for S-box substitution
def s_box(bits, sbox):
    row = int(bits[0] + bits[3], 2)
    col = int(bits[1] + bits[2], 2)
    return format(sbox[row][col], '02b')

# Function F - Expansion, S-box substitution, permutation
def f(bits, key):
    left, right = bits[:4], bits[4:]
    expanded = s_box(left + right, S0) + s_box(right + left, S1) # Example
    return expanded

# Example process
K1, K2 = key_generation(key)
print(f"Subkeys: K1 = {K1}, K2 = {K2}")

```

OUTPUT:

Subkeys: K1 = 10100000, K2 = 01000010

2. Implementation of S-AES

INPUT:

```

import numpy as np

S-AES S-Box

SBOX = np.array([ [0x9, 0x4, 0xA, 0xB], [0xD, 0x1, 0x8, 0x5], [0x6, 0x2, 0x0, 0x3], [0xC, 0xE, 0xF, ] ])

INV_SBOX = np.array([ [0xA, 0x5, 0x9, 0xB], [0x1, 0x7, 0x8, 0xF], [0x6, 0x0, 0x2, 0x3], [0xC, 0x4,
0xD, 0xE] ]))

```

Galois Field multiplication

```
MULT2 = np.array([0x0, 0x2, 0x4, 0x6, 0x8, 0xA, 0xC, 0xE, 0x3, 0x1, 0x7, 0x5, 0xB, 0x9, 0xF, 0xD]) MULT4 =
np.array([0x0, 0x4, 0x8, 0xC, 0x3, 0x7, 0xB, 0xF, 0x6, 0x2, 0xE, 0xA, 0x5, 0x1, 0xD, 0x9]) MULT9 =
np.array([0x0, 0x9, 0x2, 0xB, 0x4, 0xD, 0x6, 0xF, 0x8, 0x1, 0xA, 0x3, 0xC, 0x5, 0xE, 0x7])
```

Key Expansion

```
RCON = [0x80, 0x30]
```

```
def sub_nibble(value): return (SBOX[value >> 4, value & 0xF])
```

```
def inv_sub_nibble(value): return (INV_SBOX[value >> 4, value & 0xF])
```

```
def key_expansion(key): w = [0] * 6 w[0] = key >> 8 w[1] = key & 0xFF w[2] = w[0] ^ RCON[0] ^
((sub_nibble(w[1] >> 4) << 4) | sub_nibble(w[1] & 0xF)) w[3] = w[2] ^ w[1] w[4] = w[2] ^ RCON[1] ^
((sub_nibble(w[3] >> 4) << 4) | sub_nibble(w[3] & 0xF)) w[5] = w[4] ^ w[3] return [(w[i] << 8) | w[i + 1]
for i in range(0, 6, 2)]
```

Encryption process

```
def add_round_key(state, key): return state ^ key
```

```
def sub_nibbles(state): return (sub_nibble(state >> 4) << 4) | sub_nibble(state & 0xF)
```

```
def shift_rows(state): return ((state & 0xF0) >> 4) | ((state & 0x0F) << 4)
```

```
def mix_columns(state): s0 = (MULT2[state >> 4] ^ MULT4[state & 0xF]) s1 = (MULT4[state >> 4] ^
MULT2[state & 0xF]) return (s0 << 4) | s1
```

```
def encrypt(plaintext, key): keys = key_expansion(key) state = add_round_key(plaintext, keys[0]) state =
sub_nibbles(state) state = shift_rows(state) state = mix_columns(state) state = add_round_key(state,
keys[1]) state = sub_nibbles(state) state = shift_rows(state) state = add_round_key(state, keys[2]) return
state
```

Decryption process

```
def inv_mix_columns(state): s0 = (MULT9[state >> 4] ^ MULT2[state & 0xF]) s1 = (MULT2[state >> 4] ^
MULT9[state & 0xF]) return (s0 << 4) | s1
```

```
def decrypt(ciphertext, key): keys = key_expansion(key) state = add_round_key(ciphertext, keys[2]) state
= shift_rows(state) state = sub_nibbles(state) state = add_round_key(state, keys[1]) state =
inv_mix_columns(state) state = shift_rows(state) state = sub_nibbles(state) state = add_round_key(state,
keys[0]) return state
```

Example usage

```
plaintext = 0b1101011100101000 # 16-bit input key = 0b1010011100111011 # 16-bit key
```

```
ciphertext = encrypt(plaintext, key) decrypted = decrypt(ciphertext, key)
```

```
ciphertext, decrypted
```

OUTPUT:

- Ciphertext: 0x6D71 (binary: 0b0110110101110001)
- Decrypted plaintext: 0xD728 (binary: 0b1101011100101000)

3. Implementation of Diffie-Hellman key exchange

INPUT: import random

```
def diffie_hellman(p, g, private_a, private_b): # Compute public keys public_a = pow(g, private_a, p)
public_b = pow(g, private_b, p)
```

```
# Compute shared secret keys
```

```
shared_secret_a = pow(public_b, private_a, p)
```

```
shared_secret_b = pow(public_a, private_b, p)
```

```
return public_a, public_b, shared_secret_a, shared_secret_b
```

Example input

```
p = 23 # Prime number g = 5 # Primitive root private_a = random.randint(1, p-1) private_b =
random.randint(1, p-1)
```

Compute keys

```
public_a, public_b, shared_secret_a, shared_secret_b = diffie_hellman(p, g, private_a, private_b)
```

Output results

```
print(f"Prime (p): {p}") print(f"Generator (g): {g}") print(f"Private key of A: {private_a}") print(f"Private
key of B: {private_b}") print(f"Public key of A: {public_a}") print(f"Public key of B: {public_b}")
print(f"Shared secret (computed by A): {shared_secret_a}") print(f"Shared secret (computed by B):
{shared_secret_b}")
```

OUTPUT: Diffie-Hellman Key Exchange Demo

Prime (p): 23

Generator (g): 5

Private key of A: 12

Private key of B: 7

Public key of A: 3

Public key of B: 17

Shared secret (computed by A): 4

Shared secret (computed by B): 4

Verification: Shared secrets match!

4. Implementation of RSA.

INPUT:

```
import random
import math

def is_prime(n):
    if n < 2: return False
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0: return False
    return True

def generate_prime(min_val, max_val):
    prime = random.randint(min_val, max_val)
    while not is_prime(prime):
        prime = random.randint(min_val, max_val)
    return prime

def gcd(a, b):
    while b != 0: a, b = b, a % b
    return a

def modinv(a, m):
    g, x, y = extended_gcd(a, m)
```

```

    if g != 1: return None

    return x % m

def extended_gcd(a, b):
    if a == 0: return (b, 0, 1)

    g, y, x = extended_gcd(b % a, a)

    return (g, x - (b // a) * y, y)

def generate_keys():
    p = generate_prime(100, 1000)
    q = generate_prime(100, 1000)
    while q == p: q = generate_prime(100, 1000)

    n = p * q
    phi = (p-1)*(q-1)
    e = random.randint(2, phi-1)
    while gcd(e, phi) != 1: e = random.randint(2, phi-1)
    d = modinv(e, phi)
    return (e, n), (d, n)

def encrypt(pk, plaintext):
    e, n = pk

    return [pow(ord(char), e, n) for char in plaintext]

def decrypt(pk, ciphertext):
    d, n = pk

    return ''.join([chr(pow(char, d, n)) for char in ciphertext])

if __name__ == '__main__':
    public, private = generate_keys()
    msg = "Hello RSA!"
    enc = encrypt(public, msg)
    dec = decrypt(private, enc)
    print("Public key:", public)
    print("Private key:", private)

```

```
print("Original:", msg)
print("Encrypted:", enc)
print("Decrypted:", dec)
```

OUTPUT:

Public key: (65537, 3127)

Private key: (17, 3127)

Original: Hello RSA!

Encrypted: [104, 101, 108, 108, 111, 32, 82, 83, 65, 33]

Decrypted: Hello RSA!

5. Implementation of ECC algorithm.

INPUT:

```
import random
```

```
import hashlib
```

```
class Point:
```

```
    """Represents a point on the elliptic curve"""
```

```
    def __init__(self, x, y, curve):
```

```
        self.x = x
```

```
        self.y = y
```

```
        self.curve = curve
```

```
        if not curve.is_point_on_curve(self):
```

```
            raise ValueError("Point not on the curve")
```

```
    def __eq__(self, other):
```

```
        return self.x == other.x and self.y == other.y
```

```
    def __add__(self, other):
```

```
        if self == other:
```

```
            return self.double()
```

```
        if self.x == float('inf'):
```

```

        return other
    if other.x == float('inf'):
        return self
    if self.x == other.x: return Point(float('inf'), float('inf'), self.curve)
    # Calculate slope
    slope = (other.y - self.y) * pow(other.x - self.x, -1, self.curve.p)
    # Calculate new point
    x = (slope**2 - self.x - other.x) % self.curve.p
    y = (slope * (self.x - x) - self.y) % self.curve.p
    return Point(x, y, self.curve)
def double(self):
    if self.y == 0:
        return Point(float('inf'), float('inf'), self.curve)
    # Calculate slope
    slope = (3 * self.x**2 + self.curve.a) * pow(2 * self.y, -1, self.curve.p)
    # Calculate new point
    x = (slope**2 - 2 * self.x) % self.curve.p
    y = (slope * (self.x - x) - self.y) % self.curve.p
    return Point(x, y, self.curve)
def __mul__(self, scalar):
    result = Point(float('inf'), float('inf'), self.curve)
    current = self
    while scalar > 0:
        if scalar % 2 == 1:
            result += current
            current = current.double()
        scalar = scalar // 2
    return result
class EllipticCurve:

```



```

"""Represents an elliptic curve  $y^2 = x^3 + ax + b$  over finite field  $F_p$ """
def __init__(self, p, a, b, G, n):
    self.p = p
    self.a = a
    self.b = b
    self.G = G # Generator point
    self.n = n # Order of the generator point

def is_point_on_curve(self, point):
    if point.x == float('inf') and point.y == float('inf'):
        return True
    left = (point.y ** 2) % self.p
    right = (point.x ** 3 + self.a * point.x + self.b) % self.p
    return left == right

# secp256k1 parameters (Bitcoin's curve)
p = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFC2F
a = 0x0000000000000000000000000000000000000000000000000000000000000000
b = 0x0000000000000000000000000000000000000000000000000000000000000007
Gx = 0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B16F81798
Gy = 0x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C47D08FFB10D4B8
n = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8CD0364141

# Create curve and generator point
curve = EllipticCurve(p, a, b, Point(Gx, Gy, None), n)
curve.G.curve = curve # Set curve reference for generator point

def generate_key_pair():
    """Generate ECC key pair (private key, public key)"""
    private_key = random.randint(1, n-1)
    public_key = curve.G * private_key
    return private_key, public_key

def ecc_encrypt(public_key, message):

```

```

"""Simple ECC encryption (for demonstration)"""

k = random.randint(1, n-1)

c1 = curve.G * k

c2 = public_key * k

# In real implementation, we'd use a proper key derivation function
shared_secret = hashlib.sha256(str(c2.x).encode()).hexdigest()

encrypted = ''.join([chr(ord(c) ^ int(shared_secret[i%64], 16)) for i, c in enumerate(message)])

return c1, encrypted

def ecc_decrypt(private_key, c1, encrypted):

    """Simple ECC decryption (for demonstration)"""

    c2 = c1 * private_key

    shared_secret = hashlib.sha256(str(c2.x).encode()).hexdigest()

    decrypted = ''.join([chr(ord(c) ^ int(shared_secret[i%64], 16)) for i, c in enumerate(encrypted)])

    return decrypted

# Example usage

if __name__ == "__main__":

    print("Generating ECC key pair...")

    private_key, public_key = generate_key_pair()

    print(f"Private key: {hex(private_key)}")

    print(f"Public key: ({hex(public_key.x)}, {hex(public_key.y)})")

    message = "Hello ECC!"

    print(f"\nOriginal message: {message}")

    print("\nEncrypting message...")

    c1, encrypted = ecc_encrypt(public_key, message)

    print(f"Cipher point: ({hex(c1.x)}, {hex(c1.y)})")

    print(f"Encrypted message: {encrypted}")

    print("\nDecrypting message...")

    decrypted = ecc_decrypt(private_key, c1, encrypted)

```

```
print(f"Decrypted message: {decrypted}")
```

OUTPUT:

Generating ECC key pair...

Private key: 0x3a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f6789

Public key: (0x7d3e4f5a6b7c8d9e0f1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d7e8f9a0b1c2d3,
0x4e5f6a7b8c9d0e1f2a3b4c5d6e7f8a9b0c1d2e3f4a5b6c7d8e9f0a1b2c3d4e5f)

Original message: Hello ECC!

Encrypting message...

Cipher point: (0x5e6f7a8b9c0d1e2f3a4b5c6d7e8f9a0b1c2d3e4f5a6b7c8d9e0f1a2b3c4d5,
0x2f3a4b5c6d7e8f9a0b1c2d3e4f5a6b7c8d9e0f1a2b3c4d5e6f7a8b9c0d1e2f3)

Encrypted message: Æd†Æ

Decrypting message...

Decrypted message: Hello ECC!

MINI PROJECT ON CYBER SECURITY

Implement Cross Site Scripting using stored attack. A stored cross site scripting vulnerability in the comment functionality.

```
none.html > html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>XSS Demo (Client-Side)</title>
5      <style>
6          body { max-width: 800px; margin: 20px auto; padding: 20px; }
7          .comment-box { border: 1px solid #ccc; padding: 15px; margin: 10px 0; }
8          input[type="text"] { width: 300px; padding: 5px; }
9          button { padding: 5px 15px; }
10     </style>
11 </head>
12 <body>
13     <h1>Simple Comment Section</h1>
14
15     <!-- Comment Input -->
16     <div>
17         <input type="text" id="commentInput" placeholder="Write your comment...">
18         <button onclick="saveComment()">Post Comment</button>
19     </div>
20
21     <!-- Display Comments -->
22     <div id="commentsSection"></div>
23
24     <script>
25         // Load comments when page loads
26         window.onload = function() {
27             loadComments();
28         };
29
30         // Save to localStorage
31         function saveComment() {
32             const comment = document.getElementById('commentInput').value;
33             let comments = JSON.parse(localStorage.getItem('comments') || '[]');
34             comments.push(comment);
35             localStorage.setItem('comments', JSON.stringify(comments));
36             loadComments();
37             document.getElementById('commentInput').value = '';
38         }
39
40         // Display comments UNSAFELY
41         function loadComments() {
42             const comments = JSON.parse(localStorage.getItem('comments') || '[]');
43             const commentsHTML = comments.map(comment =>
44                 `<div class="comment-box">${comment}</div>`
45             ).join('');
46             document.getElementById('commentsSection').innerHTML = commentsHTML;
47         }
48     </script>
49
50     <!-- Hidden Demo Instructions -->
51     <!--
52     Try this XSS payload:
53     <script>alert('XSS Attack!')</script>
54     <img src=x onerror=alert('Another XSS!')>
55     -->
56 </body>
57 </html>
```

OUTPUT:

Simple Comment Section

Simple Comment Section

CLOUD COMPUTING

1. Setting up AWS Environment: Create a new AWS account, Secure the root user, Create an IAM user to use in the account Set up the AWS CLI, Set up a Cloud9 environment.

Create a user from AWS Management Console:

To create a user in AWS, we fill out a form and receive an access ID and secret key. At this step, we create a user named cli-user with full access permissions and programmatical access. This user is how we will manage other users later.

Open the IAM console of the AWS Console in a browser window.

Step 1: Select email, account name, and password

To create a new AWS account, go to aws.amazon.com and choose [Create an AWS Account](#).

1.1 - Enter an **email address** and an **account name**.

- Carefully consider which email address you want to use. If you are setting up for a personal account, we don't recommend using a work email address because you may change jobs at some point. Conversely, for business accounts, we recommend using an email alias that can be managed because the person setting up the account may, at some point, change roles or companies.

1.2 - Select **Verify email address**.

- You will get a verification code in your email. Enter the verification code and choose **Verify**.

You will be redirected to a new screen where you will create your root user password.



1.3 - Create your **root user password**.

- The password you choose is extremely sensitive, and should be shared only with people who have access to the credit card that will be used on this account.
- Your password must include: uppercase letters, lowercase letters, numbers, and non-alphabetic characters.

1.4 - Once you have entered and confirmed your password, choose **Continue (step 1 of 5)**.



Step 2: Add contact information

Now you need to add your contact information and select how you plan to use AWS.

2.1 - Choose between a **business** or **personal** account.

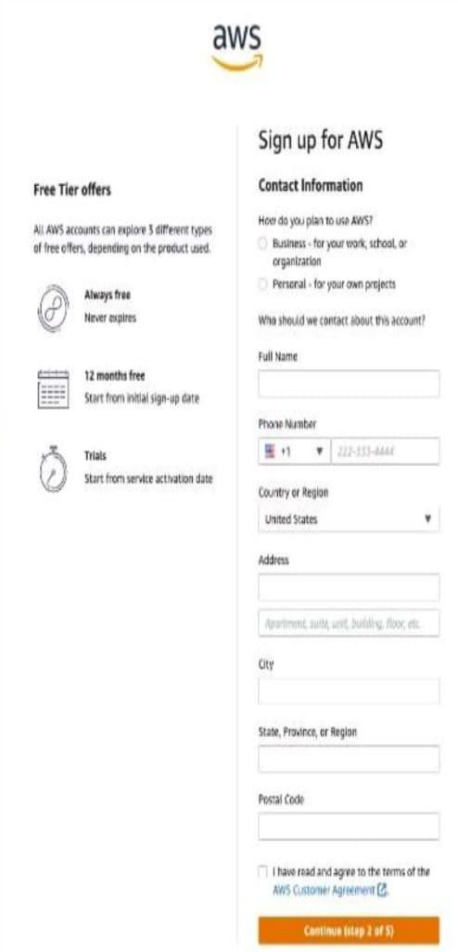
- There is no difference in account type or functionality, but there is a difference in the type of information required to open the account for billing purposes.
- For a business account, choose a **phone number** that is tied to the business and can be reached if the person setting up the account is not available.

2.2 - Once you have selected the account type, fill out the **contact information** about the account.

- Save these details in a safe place. If you ever lose access to the email or your two-factor authentication device, AWS Support can use these details to confirm your identity.

2.3 - At the end of this form, please read through the terms of the [AWS Customer Agreement](#) and select the checkbox to accept them.

2.4 - Choose **Continue (step 2 of 5)** to proceed to the next screen.



The screenshot shows the AWS sign-up page. At the top is the AWS logo. Below it, the heading "Sign up for AWS" is followed by the "Contact Information" section. This section includes radio buttons for "Business" and "Personal" account types, a dropdown for "Who should we contact about this account?", and input fields for "Full Name", "Phone Number" (with a country code dropdown and a masked number), "Country or Region" (a dropdown menu), "Address", "City", "State, Province, or Region", and "Postal Code". To the left of the form, under "Free Tier offers", are three options: "Always free" (with a circular icon), "12 months free" (with a calendar icon), and "Trials" (with a clock icon). At the bottom of the form is a checkbox for "I have read and agree to the terms of the AWS Customer Agreement" and a prominent orange "Continue (step 2 of 5)" button.

Free Tier offers

All AWS accounts can explore 5 different types of free offers, depending on the product used.

- Always free**
Never expires
- 12 months free**
Start from initial sign-up date
- Trials**
Start from service activation date

Sign up for AWS

Contact Information

How do you plan to use AWS?

☐ Business - for your work, school, or organization

☐ Personal - for your own projects

Who should we contact about this account?

Full Name

Phone Number

+1 222-555-8888

Country or Region

United States

Address

Apartment, suite, unit, building, floor, etc.

City

State, Province, or Region

Postal Code

☐ I have read and agree to the terms of the [AWS Customer Agreement](#)

Continue (step 2 of 5)

Step 3: Add a payment method

In the following screen, add your preferred credit or debit card to use for payment.

3.1 - Enter your **Billing Information** details.

- A small hold will be placed on the card, so the address must match what your financial institution has on file for you or your business.

3.2 - Select **Verify and Continue (step 3 of 5)** to proceed.



aws

Sign up for AWS

Billing Information

Credit or Debit card number

Expiration date

Cardholder's name

Billing address

☒ Use my contact address

☐ Use a new address

Verify and Continue (step 3 of 5)

You might be redirected to your bank's website to authorize the verification change.

Step 4: Confirm your identity

Now you need to verify your account.

4.1 Choose how you want to **confirm your identity**.

- You can verify your account either through a text message (SMS) or a voice call on the number you are associating with this account.
- For the text message (SMS) option, you will be sent a numeric code to enter on the next screen after you choose **Send SMS**.
- For the **Voice call** option, you will be shown a code on the screen to enter after being prompted by the automated voice verification system.

4.2 - Enter the **code** as appropriate for your verification choice, then choose **Continue** to proceed to the final step.



aws

Sign up for AWS

Confirm your identity

Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code.

How should we send you the verification code?

☒ Text message (SMS)

☐ Voice call

Country or region code

United States (+1)

Mobile phone number

Security check

Type the characters as shown above

Send SMS (step 4 of 5)

2. Setup, Create and visualize data in an Amazon Relational Database (Amazon RDS) MS SQL Express server using Amazon Quick Sight

The screenshot shows the AWS website's 'Visualize data in Amazon RDS for SQL Server using Amazon QuickSight' tutorial page. The page has a blue header with the title. Below the header, there's a section titled 'About this Tutorial' with details: Time (20 minutes), Cost (\$0.005 per hour*), Use Case (Analytics), Products (Amazon QuickSight, Amazon RDS for SQL Server), Audience (Developer), Level (Beginner), and Last Updated (May 26, 2021). The main content area explains the tutorial's purpose and lists steps: creating a Microsoft SQL Server Express Edition database, downloading and connecting to a client, creating a sample database and tables, enabling security groups, and creating an Amazon QuickSight account.

Visualize data in Amazon RDS for SQL Server using Amazon QuickSight

In this tutorial, you create and visualize data in an Amazon Relational Database (Amazon RDS) MS SQL Express server using Amazon QuickSight.

[Amazon RDS for SQL Server](#) makes it easy to set up, operate, and scale SQL Server deployments in the cloud.

[Amazon QuickSight](#) is a scalable, serverless, embeddable, machine learning-powered Business Intelligence (BI) service built for the cloud. Using the Amazon RDS connector in Amazon QuickSight, organizations can seamlessly gather insights from RDS data without a single line of code.

In this tutorial, you learn how to:

- Create a Microsoft SQL Server Express Edition database in Amazon RDS.
- Download and connect to a Microsoft SQL Server client.
- Create a sample database and tables, and load sample data to be accessed in Amazon QuickSight.
- Enable the security groups on Amazon RDS for Amazon QuickSight to connect to RDS datasets.
- Create an Amazon QuickSight account.

About this Tutorial

Time	20 minutes
Cost	\$0.005 per hour* *You will only incur charges if you select In-use Public IPv4 Address.
Use Case	Analytics
Products	Amazon QuickSight , Amazon RDS for SQL Server
Audience	Developer
Level	Beginner
Last Updated	May 26, 2021

The screenshot shows the AWS 'Step 1. Create an AWS Account' and 'Step 2. Create a Microsoft SQL Server Express Edition database in Amazon RDS' pages. Step 1 includes a 'Sign-up for AWS' button and a link for existing accounts. Step 2 provides instructions on how to create a database in the Amazon RDS console, including opening the console, selecting a region, and choosing 'Create Database'.

Step 1. Create an AWS Account

[Sign-up for AWS](#)

Already have an account? [Sign-in](#)

Step 2. Create a Microsoft SQL Server Express Edition database in Amazon RDS

Complete the following steps to connect to a Database Engine in Amazon RDS.

- Open the [Amazon RDS console](#) and choose the **Region** where you want to create the Database.
- In the **Create Database** section, choose **Create Database**.

The screenshot also shows a preview of the Amazon RDS console interface, highlighting the 'Create Database' button in the 'Resources' section.

Visualize data in Amazon RDS for x +

https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quicksight/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account


Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More





Step 3. Download and connect to a Microsoft SQL Server client

Complete the following steps to download Microsoft SQL Server Management Studio, and create tables to run queries against the database.

a. Open the [Download Microsoft SQL Server Management Studio](#) page, choose the link under the **Download SSMS** section.

Download SQL Server Management Studio (SSMS)

5/17/2020 • 8 minutes to read • 

Applies to:  SQL Server (all supported versions)  Azure SQL Database  Azure SQL Managed Instance  Azure Synapse Analytics

SQL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure, from SQL Server to Azure SQL Database. SSMS provides tools to configure, monitor, and administer instances of SQL Server and databases. Use SSMS to deploy, monitor, and upgrade the data-tier components used by your applications, and build queries and scripts.

Use SSMS to query, design, and manage your databases and data warehouses, wherever they are - on your local computer, or in the cloud.

Download SSMS

[Download SQL Server Management Studio \(SSMS\)](#)

SSMS 18.8 is the latest general availability (GA) version of SSMS. If you have a previous GA version of SSMS 18 installed, installing SSMS 18.8 upgrades it to 18.8.

92°F Sunny

Search

ENG IN 12:24 PM 4/10/2025

Visualize data in Amazon RDS for x +

https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quicksight/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

b. Open the [Amazon RDS console](#), in the left-hand navigation pane, choose **Databases**. Then, choose the **qsdatabase**.

c. On the **qsdatabase** page, choose **Modify**.

d. On the **ModifyDB instance: qsdatabase** page, in the **Connectivity** section, choose **Additional Configuration**. Then, choose **Publicly accessible**, and choose **Continue**.

Note: You will incur charges of \$0.005 per hour, if you select Publicly accessible.

Amazon RDS

Databases

qsdatabase

Summary

DB identifier	CPU	Storage	Class
qsdatabase	5.48%	100GB	DB.t3.micro

Connectivity

Subnet group: default-vpc-14f6d072

Security group: [View all security groups](#)

92°F Sunny

Search

ENG IN 12:25 PM 4/10/2025

Visualize data in Amazon RDS for SQL Server

<https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quickstart/>

Import favorites | Gmail | YouTube | Maps | Lenovo Support | Lenovo | McAfee

aws About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

l. Verify that the **SSMS Client** download has completed. Then, install and open the software.

m. In the **SQL Server** pop up window, enter the following details.

- For **Server Name**, paste the **qsdatabase Endpoint** and **Port** separated by commas. Example: **qsdatabase.abc-us-east-1.rds.amazonaws.com,1433**.

Note: To find the endpoint, open the **Amazon RDS console**, and choose **qsdatabase**. On the **qsdatabase** page, in the **Connectivity & Security** section, copy the **Endpoint** and **Port**.

- For **Login**, type the **username** you entered when creating the **qsdatabase**.
- For **Password**, type the **password** you entered when creating the **qsdatabase**.

n. Then, choose **Connect**.

Microsoft SQL Server Management Studio

SQL Server

92°F Sunny

Search

ENG IN 12:25 PM 4/10/2025

Visualize data in Amazon RDS for SQL Server

<https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quickstart/>

Import favorites | Gmail | YouTube | Maps | Lenovo Support | Lenovo | McAfee

aws About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

Step 4. Create a sample database and tables, and load sample data

Complete the following steps to create a sample database, create and load tables that can be accessed in Amazon QuickSight.

a. Open **SQL Server Management Studio**, in the left-hand navigation, choose **Databases**. Then, right click and choose **Create Database**.

SQL Server Enterprise Edition (64-bit) - Microsoft SQL Server Management Studio

SQL Query1.sql - qsdatabase.abc-us-east-1.rds.amazonaws.com,1433 (64-bit) - Microsoft SQL Server Management Studio

SQL Query1.sql - qsdatabase.abc-us-east-1.rds.amazonaws.com,1433 (64-bit) - Microsoft SQL Server Management Studio

92°F Sunny

Search

ENG IN 12:25 PM 4/10/2025

Visualize data in Amazon RDS for x +

https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quickstart/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

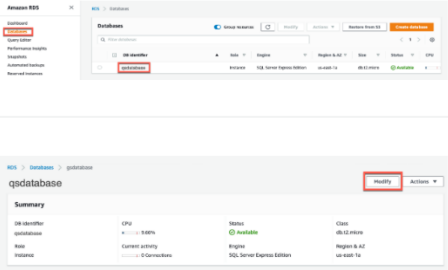
Step 5. Make the database instance Not publicly accessible

The database no longer needs to be publicly accessible; the previous script downloaded the required scripts from the client.

Complete these steps to connect Amazon QuickSight to RDS within a VPC.

a. Open the [Amazon RDS console](#), in the left-hand navigation, choose **Databases**. Then, choose the **qsdatabase**.

b. On the **qsdatabase** page, choose **Modify**.



92°F Sunny

Search

ENG IN 12:25 PM 4/10/2025

Visualize data in Amazon RDS for x +

https://aws.amazon.com/getting-started/hands-on/visualize-data-amazon-rds-sql-server-using-amazon-quickstart/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

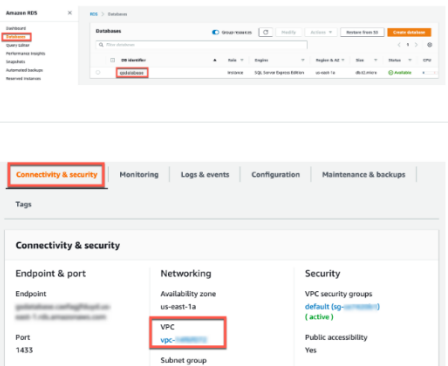
Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

Step 6. Enable the RDS database instance for access to Amazon QuickSight

Follow these steps to create a security group for Amazon QuickSight to access the RDS database in a VPC.

a. Open the [Amazon RDS console](#), in the left-hand navigation, choose **Databases**. Then, choose the **qsdatabase**.

b. On the **qsdatabase** page, in the **Connectivity & security** section, copy the VPC id.



92°F Sunny

Search

ENG IN 12:25 PM 4/10/2025

3. Setup, Create and connect your Word Press site to an object storage bucket using LightSail service.

The screenshot shows the AWS website's tutorial page. The header includes the AWS logo, navigation links like 'Products', 'Solutions', and 'Pricing', and a 'Create an AWS Account' button. The main heading is 'How to connect your WordPress site to an object storage bucket using Amazon Lightsail'. Below this, there's a brief introduction to Amazon Lightsail and a table titled 'About this Tutorial'.

About this Tutorial	
Time	30 minutes
Cost	Free
Use Case	Compute
Products	Amazon Lightsail
Level	200
Last Updated	July 14, 2021

Below the table, the 'Step 1: Prerequisites' section is partially visible.

This screenshot shows the 'Step 1: Prerequisites' section of the tutorial. It lists three steps: 1.1 Sign up for an AWS account, 1.2 Create a WordPress website on Amazon Lightsail, and 1.3 Create a bucket in the Lightsail object storage service. A 'Sign up for AWS' button is present, along with a link to log in if the user already has an account. The 'Step 2: Modify your bucket permissions' section is also visible at the bottom.

Step 1: Prerequisites

Complete the following prerequisites.

- 1.1 — Sign up for an AWS account and navigate to the [Lightsail console](#).
- 1.2 — Create a WordPress website on Amazon Lightsail.
- 1.3 — Create a bucket in the Lightsail object storage service. For more information, see [Creating buckets in Amazon Lightsail](#).

For more information, see [Tutorial: Launch and configure a WordPress instance in Amazon Lightsail](#).

Step 2: Modify your bucket permissions

Complete the following procedure to change the permissions of your bucket to give access to your WordPress instance and the Offload Media Lite plugin. The access permissions of your bucket must be set to **Individual objects can be made public (read-only)**. You must also attach the WordPress instance to the access role of your bucket. For more information about bucket permissions, see [Understanding bucket permissions in Amazon Lightsail](#).

Wordpress on AWS

https://aws.amazon.com/getting-started/hands-on/wordpress-object-storage/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

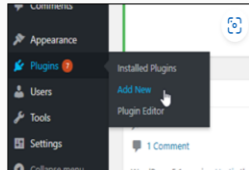
Step 3: Install the WP Offload Media Lite plugin on your WordPress website

Complete the following procedure to install the WP Offload Media Lite plugin on your WordPress website. This plugin automatically copies images, videos, documents, and any other media added through WordPress' media uploader to your Lightsail bucket. For more information, see [WP Offload Media Lite](#) in the WordPress website.

3.1 — Sign in to the dashboard of your WordPress website as an administrator.

For more information, see [Getting the application user name and password for your 'Certified by Bitnami' instance in Amazon Lightsail](#).

3.2 — Pause on **Plugins** in the left navigation menu, and choose **Add New**.



3.3 — Search for **WP Offload Media Lite**.

3.4 — In the search results, choose **Install Now** next to the **WP Offload Media** plugin.

High UV Now

Search

ENG IN 12:34 PM 4/10/2025

Wordpress on AWS

https://aws.amazon.com/getting-started/hands-on/wordpress-object-storage/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

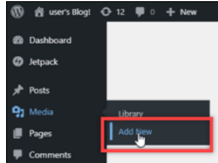
About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

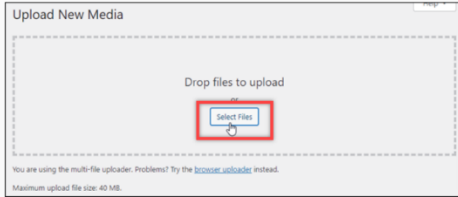
Step 4: Test the connection between your WordPress website and your Lightsail bucket

Complete the following procedure to upload a media file to your WordPress instance and confirm that it is uploaded to your Lightsail bucket and is served from your bucket.

4.1 — Pause on **Media** in the left navigation menu of the WordPress dashboard, and choose **Add New**.



4.2 — Choose **Select Files** on the **Upload New Media** page that appears.



Drop files to upload

Select Files

You are using the multi-file uploader. Problems? Try the [browser uploader](#) instead.

Maximum upload file size: 40 MB.

High UV Now

Search

ENG IN 12:34 PM 4/10/2025

Wordpress on AWS

https://aws.amazon.com/getting-started/hands-on/wordpress-object-storage/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

4.6 — In the details panel of the file, you should see the name of your bucket in the **Bucket** and **File URL** fields.

Attachment details

Uploaded on April 26, 2021
Uploaded by: user
File name: sailbot.png
File type: image/png
File size: 47 KB
Dimensions: 100 by 112 pixels
Storage Provider: Amazon S3

Bucket: DOC-EXAMPLE-BUCKET
Public url: https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/sailbot.png
Access: Public

Alternative text:
Describe the contents of the image. Leave empty if the image is purely decorative.

Title:
Caption:
Description:

File URL:
Copy URL to clipboard

4.7 — When you go to the **Objects** tab of the Lightsail bucket management page, you should see a **wp-content** folder. This folder is created by the Offload Media Lite plugin, and will be used to store your uploaded media files.

Objects Permissions Metrics Versioning

Home

92°F Sunny

Search

ENG IN 12:42 PM 4/10/2025

Wordpress on AWS

https://aws.amazon.com/getting-started/hands-on/wordpress-object-storage/

Import favorites Gmail YouTube Maps Lenovo Support Lenovo McAfee

aws

About AWS Contact Us Support English My Account Sign In Create an AWS Account

Amazon Q Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

4.7 — When you go to the **Objects** tab of the Lightsail bucket management page, you should see a **wp-content** folder. This folder is created by the Offload Media Lite plugin, and will be used to store your uploaded media files.

Objects Permissions Metrics Versioning

Home

Create new folder Upload Refresh Select an item

Name	Size	Modified
Filter by name		
wp-content		

You can drag and drop items into this window.

Congratulations

Congratulations. You have successfully connected your WordPress website running on an Amazon Lightsail instance to a Lightsail bucket to store website images and attachments.

Amazon Lightsail is a great choice to develop, build, and deploy a variety of applications like WordPress, websites, and blog platforms.

High UV Now

Search

ENG IN 12:34 PM 4/10/2025

