

Frequently Asked Questions (FAQ)

This document provides solutions to common issues encountered while working with AWS EC2 instances, SSH connections, and terminal operations. Each section includes exact error messages and step-by-step solutions.

Table of Contents

- [AWS EC2 Issues](#)
- [SSH Connection Issues](#)
- [Terminal & System Issues](#)
- [Memory & Resource Issues](#)

AWS EC2 Issues

1. Insufficient Instance Capacity

Error Message:

We currently do not have sufficient [instance-type] capacity in the Availability Zone you requested. Our system will be working on provisioning additional capacity.

Solution:

- Wait a few minutes and retry the launch

2. Instance Limit Exceeded

Error Message:

You have requested more instances (X) than your current instance limit of Y allows for the specified instance t

Solution:

- Request a limit increase through AWS Service Quotas
- Terminate unused instances to free up capacity
- Use a different instance type that has available quota

Steps to Request Limit Increase:

1. Go to AWS Console → Service Quotas
2. Search for "EC2" → Select "Amazon Elastic Compute Cloud (Amazon EC2)"
3. Find "Running On-Demand [instance-family] instances"
4. Click "Request quota increase"
5. Enter the new limit value and submit

3. Volume Limit Exceeded

Error Message:

You have exceeded your maximum gp2 storage limit of X GB in this region.

Solution:

- Delete unused EBS volumes and snapshots
- Request an EBS volume limit increase
- Use a different region with available capacity

Steps to Clean Up Volumes:

1. EC2 Dashboard → Elastic Block Store → Volumes
2. Filter by "available" state (unattached volumes)
3. Select and delete unused volumes
4. Go to Snapshots and delete old snapshots

SSH Connection Issues

1. Host Key Verification Failed

Error Message:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
The fingerprint for the ECDSA key sent by the remote host is SHA256:...
Host key verification failed.
```

Cause: This occurs when you delete an EC2 instance and create a new one with the same IP address. The SSH client detects a different host key.

Solution (macOS/Linux):

```
# Remove the old host key for the specific IP
ssh-keygen -R <your-instance-ip>

# Or manually edit the known_hosts file
nano ~/.ssh/known_hosts
# Delete the line containing your instance IP and save
```

Solution (Windows):

```
# Using PowerShell
Remove-Item -Path "$env:USERPROFILE\.ssh\known_hosts"

# Or edit manually
notepad C:\Users\<YourUsername>\.ssh\known_hosts
# Delete the line with your instance IP and save
```

2. Permission Denied (publickey)

Error Message:

```
Permission denied (publickey).
```

or

```
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
```

Causes & Solutions:

A. Incorrect Key Permissions (macOS/Linux)

The private key file has incorrect permissions.

```
# Check current permissions
ls -l your-key.pem

# Fix permissions (must be 400 or 600)
chmod 400 your-key.pem

# Verify
ls -l your-key.pem
# Should show: -r----- or -rw-----
```

B. Wrong Username

```
# Incorrect
ssh -i key.pem root@ec2-xx-xx-xx-xx.compute.amazonaws.com
```

```
# Correct for Amazon Linux / Amazon Linux 2
ssh -i key.pem ec2-user@ec2-xx-xx-xx-xx.compute.amazonaws.com

# Correct for Ubuntu
ssh -i key.pem ubuntu@ec2-xx-xx-xx-xx.compute.amazonaws.com
```

C. Wrong Key File

Ensure you're using the correct private key (.pem) file that corresponds to the key pair selected during instance launch.

D. Public Key Not in Instance

The instance must have the public key in `~/.ssh/authorized_keys` . If missing, use EC2 Instance Connect or EC2 Serial Console to add it.

3. Connection Timeout

Error Message:

```
ssh: connect to host ec2-xx-xx-xx-xx.compute.amazonaws.com port 22: Connection timed out
```

Causes & Solutions:

A. Security Group Missing SSH Rule

1. Go to EC2 Dashboard → Security Groups
2. Select the security group attached to your instance
3. Click "Inbound rules" → "Edit inbound rules"
4. Add rule:
 - Type: SSH
 - Protocol: TCP
 - Port: 22
 - Source: My IP (or 0.0.0.0/0 for any IP - less secure)
5. Save rules

B. Instance Not Running

```
# Check instance state in AWS Console
# Instance State should be "running" with 2/2 status checks passed
```

C. Network ACL Blocking Traffic

1. EC2 Dashboard → Network ACLs
2. Check inbound and outbound rules for your subnet
3. Ensure port 22 is allowed

4. Bad Permissions (Windows)

Error Message:

```
WARNING: UNPROTECTED PRIVATE KEY FILE!  
Permissions for 'key.pem' are too open.  
It is required that your private key files are NOT accessible by others.
```

Solution (Windows using WSL or Git Bash):

```
chmod 400 key.pem
```

Solution (Windows PowerShell):

```
# Remove inheritance  
icacls.exe key.pem /inheritance:r  
  
# Grant full control to current user only  
icacls.exe key.pem /grant:r "%username%):(R)"
```

5. Could Not Resolve Hostname

Error Message:

```
ssh: Could not resolve hostname ec2-xx-xx-xx-xx.compute.amazonaws.com: Name or service not known
```

Solution:

- Check your internet connection
- Verify the hostname is correct (copy from AWS Console)
- Try using the public IP address instead:

```
ssh -i key.pem ubuntu@xx.xx.xx.xx
```

- Check DNS settings

6. Too Many Authentication Failures

Error Message:

```
Received disconnect from xx.xx.xx.xx port 22:2: Too many authentication failures
```

Solution:

```
# Explicitly specify the identity file
ssh -i /path/to/your-key.pem -o IdentitiesOnly=yes ubuntu@xx.xx.xx.xx

# Or clear your ssh-agent
ssh-add -D
ssh-add /path/to/your-key.pem
```

Terminal & System Issues

1. Command Not Found

Error Message:

```
bash: python: command not found
```

or

```
bash: nvidia-smi: command not found
```

Solution:

```
# Check if Python is installed as python3
which python3

# Create an alias if needed
alias python=python3

# For nvidia-smi, ensure CUDA is in PATH
export PATH=/usr/local/cuda/bin:$PATH
echo 'export PATH=/usr/local/cuda/bin:$PATH' >> ~/.bashrc
source ~/.bashrc
```

2. Permission Denied on Script Execution

Error Message:

```
bash: ./script.sh: Permission denied
```

Solution:

```
# Make the script executable
chmod +x script.sh

# Then run it
./script.sh
```

3. Disk Space Full

Error Message:

```
No space left on device
```

Solution:

```
# Check disk usage
df -h

# Find large files
du -h --max-depth=1 /home/ubuntu | sort -hr | head -20

# Clean up common space hogs
sudo apt-get autoremove
sudo apt-get clean
rm -rf ~/.cache/*

# Check Docker disk usage (if applicable)
docker system df
docker system prune -a
```

Memory & Resource Issues

1. Kernel Died / Jupyter Kernel Crash

Error Message (Jupyter):

```
The kernel appears to have died. It will restart automatically.
```

Error in Terminal:

```
Killed
```

Cause: Out of Memory (OOM) - The system ran out of RAM

Solution:

A. Check Memory Usage:

```
# Monitor memory in real-time
htop
# Or
free -h

# Check system logs for OOM killer
dmesg | grep -i "killed process"
sudo journalctl -xe | grep -i "out of memory"
```

B. Reduce Model/Batch Size:

```
# In your Python code
# Reduce batch size
batch_size = 16 # Instead of 32 or 64

# Use smaller model
model = "distilbert-base-uncased" # Instead of "bert-large-uncased"

# Clear GPU memory
import torch
torch.cuda.empty_cache()
```

C. Add Swap Space:

```
# Create 4GB swap file
sudo fallocate -l 4G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile

# Verify
free -h

# Make permanent
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

D. Upgrade Instance Type:

- t2.micro (1GB RAM) → t2.small (2GB) or t2.medium (4GB)
- For deep learning: Use compute-optimized or GPU instances (g4dn, p3, etc.)

2. CUDA Out of Memory

Error Message:

RuntimeError: CUDA out of memory. Tried to allocate X.XX GiB (GPU 0; X.XX GiB total capacity; X.XX GiB already

Solution:

```
# 1. Reduce batch size
batch_size = 8 # Smaller batch

# 2. Use gradient accumulation
accumulation_steps = 4
for i, (inputs, labels) in enumerate(dataloader):
    outputs = model(inputs)
    loss = criterion(outputs, labels)
    loss = loss / accumulation_steps
    loss.backward()

    if (i + 1) % accumulation_steps == 0:
        optimizer.step()
        optimizer.zero_grad()

# 3. Use mixed precision training
from torch.cuda.amp import autocast, GradScaler
scaler = GradScaler()

with autocast():
    outputs = model(inputs)
    loss = criterion(outputs, labels)

# 4. Clear cache regularly
import torch
torch.cuda.empty_cache()

# 5. Use gradient checkpointing
model.gradient_checkpointing_enable()
```

3. Process Killed Without Error

Symptom: Your training script or process suddenly stops with no error message, just "Killed".

Cause: Linux OOM (Out of Memory) Killer terminated the process.

Check Logs:

```
# Check system logs
dmesg -T | grep -i "killed process"
# Look for output like: "Out of memory: Killed process 1234 (python)"

# Check kernel logs
sudo journalctl -xe | grep -i oom
```

Solution:

1. Monitor memory usage before running:

```
watch -n 1 free -h
```

2. Use a smaller model or dataset

3. Add swap space (see solution 1C above)

4. Upgrade to larger instance type

5. Process data in smaller chunks:

```
# Instead of loading all data
# data = load_all_data()

# Process in chunks
for chunk in pd.read_csv('large_file.csv', chunksize=10000):
    process(chunk)
```

4. Segmentation Fault (Core Dumped)

Error Message:

```
Segmentation fault (core dumped)
```

Causes & Solutions:

A. Incompatible Library Versions:

```
# Check installed versions
pip list | grep torch
pip list | grep tensorflow

# Reinstall compatible versions
pip install --upgrade torch torchvision torchaudio
```

B. Corrupted Installation:

```
# Reinstall the problematic package
pip uninstall <package>
pip install <package> --no-cache-dir
```

C. Hardware Issues:

```
# Test memory
sudo apt-get install memtester
sudo memtester 1G 1
```

```
# Check system logs  
dmesg | tail -50
```

Additional Resources

- [AWS EC2 Documentation](#)
- [AWS Support Center](#)
- [SSH Key Troubleshooting](#)
- [EC2 Instance Connect](#)

Getting Help

If you encounter an issue not covered here:

1. **Copy the exact error message** - Include the full error output
2. **Note your environment** - OS, instance type, what you were trying to do
3. **Check AWS/system logs** - Often contain more detailed error information
4. **Search the error message** - Many issues are well-documented online
5. **Contact your instructor** - Provide all the above information

Last Updated: January 2026