

# EC2 Recovery

This guide demonstrates how to recover access to an EC2 instance when both SSH and Serial Console access are unavailable. We'll use a Proxmox instance as an example, but this method works for any Linux-based EC2 instance.

## Prerequisites

- AWS Console access
- Basic understanding of Linux commands
- A working EC2 instance to use as rescue system (Amazon Linux 2 recommended)

## Recovery Steps

### 1. Create a Snapshot of the Affected Volume

1. Navigate to EC2 Dashboard
2. Go to Volumes and select the volume of the affected instance
3. Actions → Create Snapshot
4. Add descriptive name like "Pre-rescue-backup-[date]"
5. Click Create Snapshot
6. Wait for snapshot to reach 100% completion

### 2. Stop the Affected Instance

1. Navigate to EC2 Dashboard
2. Select the affected instance
3. Actions → Instance State → Stop
4. Wait until instance is fully stopped

### 3. Detach the Root Volume

1. Select the stopped instance
2. Scroll to 'Storage' tab
3. Note the volume ID of the root volume
4. Right-click the volume → Detach Volume
5. Confirm detach

## 4. Launch a Rescue Instance

1. Launch a new EC2 instance
2. Use Amazon Linux 2 AMI
3. Same availability zone as affected volume
4. Configure security group to allow SSH access

## 5. Attach Problem Volume to Rescue Instance

1. Select the detached volume
2. Actions → Attach Volume
3. Select rescue instance
4. Note the device name (e.g., /dev/sdb or /dev/xvdb)

## 6. Access and Mount the Volume

```
# Connect to rescue instance
ssh -i your-key.pem ec2-user@rescue-instance-ip

# List available disks to find attached volume
sudo fdisk -l

# or
lsblk

# Create mount point
sudo mkdir -p /mnt/rescue

# Mount the root partition
sudo mount /dev/xvdb1 /mnt/rescue # Adjust device name as needed
```

## 7. Troubleshoot and Fix Issues

### Common File Locations

```
# Network Configuration
sudo nano /mnt/rescue/etc/network/interfaces # Debian/Ubuntu/Proxmox
sudo nano /mnt/rescue/etc/sysconfig/network-scripts/ifcfg-eth0 # RHEL/CentOS

# SSH Configuration
```

```
sudo nano /mnt/rescue/etc/ssh/sshd_config

# System Logs
sudo less /mnt/rescue/var/log/syslog    # Debian/Ubuntu
sudo less /mnt/rescue/var/log/messages # RHEL/CentOS
```

## Example: Fixing Proxmox Network Configuration

```
# View current network config
sudo cat /mnt/rescue/etc/network/interfaces

# Edit if needed
sudo nano /mnt/rescue/etc/network/interfaces

# Example of working basic config:
auto lo
iface lo inet loopback

iface eth0 inet manual

auto vmbr0
iface vmbr0 inet dhcp
    bridge-ports eth0
    bridge-stp off
    bridge-fd 0
```

## 8. Cleanup and Restore

```
# Unmount volume
cd ~ # Ensure you're not in mounted directory
sudo umount /mnt/rescue
```

After unmounting:

1. Detach volume from rescue instance in AWS Console
2. Reattach to original instance as root volume
3. Start original instance
4. Test connectivity

# Common Issues and Solutions

## Network Configuration Issues

- Check for correct interface names (eth0, ens5, etc.)
- Verify gateway configuration
- Ensure no conflicting network bridges
- Check for valid IP addressing

## Boot Issues

- Check /boot partition isn't full
- Verify fstab entries are correct
- Check for kernel issues in grub configuration

## Permission Issues

- Verify SSH key permissions
- Check SELinux/AppArmor settings
- Validate root access configuration

## Prevention Tips

1. Always maintain a snapshot of working system
2. Document working network configuration
3. Use AWS Systems Manager Session Manager as backup access method
4. Keep serial console access enabled
5. Document network changes before implementing
6. Test changes in staging environment first

## Additional Resources

- [AWS EC2 User Guide](#)
- [Proxmox Network Configuration](#)
- [AWS Volume Management](#)

Remember: Always maintain current backups and document your system configuration to make recovery easier when needed.

---

Revision #2

Created 23 April 2025 02:53:23 by Jack Waterhouse

Updated 23 April 2025 02:57:11 by Jack Waterhouse