1. Log File Analyzer:

Create a script that analyzes web server logs (e.g., Apache, Nginx) for common patterns such as the number of 404 errors, the most requested pages, or IP addresses with the most requests. The script should output a summarized report.

Python

```
import re
import collections
from datetime import datetime

def analyze_log(log_file):

    error_count = 0
    request_counts = collections.Counter()
    ip_counts = collections.Counter()

    log_format = r'(\d+\.\d+\.\d+\.\d+) - - \[(.*?)\] "(.*?)" (\d+) (\d+) "(.*?)" "(.*?)"'
    with open(log_file, 'r') as f:
        for line in f:
            match = re.match(log_format, line)
            if match:
                ip, timestamp, request, status, size, referrer, user_agent = match.groups()
                if int(status) >= 400:
                    error_count += 1
                request_counts[request] += 1
                ip_counts[ip] += 1

    # Generating summary report

    print("Error Count:", error_count)
    print("Most Requested Pages:")
    for request, count in request_counts.most_common(10):
        print(f"  {request}: {count}")
    print("Top IP Addresses:")
    for ip, count in ip_counts.most_common(10):
        print(f"  {ip}: {count}")

if __name__ == "__main__":
    log_file = "access.log"
    analyze_log(log_file)
```

*Replacing the actual path or file of access log at the log_file declaration.

2. Automated Backup Solution:

Write a script to automate the backup of a specified directory to a remote server or a cloud storage solution. The script should provide a report on the success or failure of the backup operation.

Python

```
import boto3
import os
import datetime
import logging

def backup_to_s3(local_dir, bucket_name, prefix):

    s3 = boto3.client('s3')

    try:
        for root, dirs, files in os.walk(local_dir):
            for file in files:
                local_path = os.path.join(root, file)
                relative_path = os.path.relpath(local_path, local_dir)
                s3_path = os.path.join(prefix, relative_path)

                s3.upload_file(local_path, bucket_name, s3_path)

        return True
    except Exception as e:
        logging.error(f"Error backing up: {e}")
        return False

def main():
    local_dir = 'local/s3/dom/my_directory'
    bucket_name = 'mys3bucket'
    prefix = 'backup/'
    backup_success = backup_to_s3(local_dir, bucket_name, prefix)

    if backup_success:
        print(f"Backup to S3 bucket {bucket_name} successful at
       {datetime.datetime.now()}")
    else:
        print(f"Backup to S3 bucket {bucket_name} failed at {datetime.datetime.now()}")

if __name__ == "__main__":
    main()
```