Question 1

This is a test. Using any publicly available dataset, write a script that will take data from that source and output it via http in json format to either one of the following inputs: Elastic, Splunk http input, Cribl HTTP input, or s3. We should be able to run the script while passing along the required arguments to make it work with your choice of destination.

The script I wrote required a dataset. A really interesting one that I found was data.gov. If you simply click the search bar with no search context it pulls random datasets published by various government departments. Seeing how the data is formatted you can start to think about how you might pull multiple datasets to correlate. In this example the Economic Research Service (ERS) used Circana (formerly Information Resources Inc. [IRI]) between 2013 and 2020 tracking all items sold in grocery stores nationwide. The data is originally in .csv format. My script can take the URL of the specific dataset, convert it to json, and send it to a test elasticsearch instance that I was able to get for free. If we could imagine this on an elastic dash board or something like a Grafana! Think about how big level data driven decisions could be made to improve our future! It would revolutionize government spending and bureaucracy! The URLs to the respective endpoints are all listed below:

https://catalog.data.gov/dataset/fruit-and-vegetable-prices
https://www.ers.usda.gov/data-products/fruit-and-vegetable-prices.aspx
Fruit Prices 2020.csv (local download)

In addition to describing the simple functionality of my script I did want to provide you with a bit of "where my head was at" as I put it together. Hopefully, this demonstrates how I think a bit better. I start by naming a variable as one of the endpoint data sources from the ERS. I do this with [Invoke-RestMethod -uri] which will connect via http/https. Once the data was pulled down it wasn't too hard to switch it over to json. It took two tries because originally my syntax just said simply convert TO json there was a struggle with the intake and the data. Once I added a convert FROM .csv it seemed to standardize the input therefore I could do a conversion to json with a uniform input.

I want to explain the last part openly and honestly because I think this is one of my attributes that can help any team! I did not have an elasticsearch instance of my own before this paper, I got one. While the pulling down of data is something sort of in my skill set already; sending it to a a third party like elasticsearch was completely new to me. I went to work reading the following links on Elastic's Documentation. That is how I got the instance running and therefore built my endpoint. Elastic provided me with those credentials. I think this is important because it shows that I don't understand something; I will make it my only business to learn it! Check out those links below:

https://zacktestenvironment.kb.us-central1.gcp.cloud.es.io:9243/api/security/saml/callback
https://discuss.elastic.co/t/load-json-to-elasticsearch/195235

See the full script below:

```
$TestData = "https://www.ers.usda.gov/webdocs/DataFiles/51035/Fruit%20Prices%202020.csv?v=499.9"
$response = Invoke-RestMethod -Uri $TestData

if ($response) {
    # Success
    $json_data = $response | ConvertFrom-Csv | ConvertTo-Json
    Write-Output $json_data
}
else {
    echo "Double Check The Link You Provided, It's Not Working."
}

{
Install-Module -Name "Elasticsearch" -Force

    $elasticUrl = "https://zacktestenvironment.kb.us-central1.gcp.cloud.es.io:9243/api/security/saml/callback"
    $elasticUsername = "elastic"
    $elasticPassword = "zPRscsnMd0tfCzk9bQEnNe2g"


    $credentials = [System.Text.Encoding]::UTF8.GetBytes("$($elasticUsername):$($elasticPassword)")
    $base64AuthInfo = [System.Convert]::ToBase64String($credentials)
    $headers = @{ Authorization = "Basic $base64AuthInfo" }


    $result = Invoke-RestMethod -Uri $elasticUrl -Method Post -Headers $headers -Body $json_data -ContentType
"application/json"

    if ($result.created -eq $true) {
        Write-Output "Data sent successfully to Elastic!"
    }

    else {
        Write-Output "Failed to send data to Elastic. Error: $($result.error.type)"
    }
}
```