Network Classifier

You are going to develop a system that will receive two input files (‘classification_rules.csv’, communications.csv’’),
and produce an output file (‘classifications.csv’) that will contain the classification of the network devices.

The Classification Process

The system will read the communications and try to match them for existing rules. if a rule is matched, the rule type will be applied to the communication ‘device_id’

Data Models

Classification Rules (csv)
id (int), type (text), argument (nullable text), classification (text)

- id - the rule id
- type - the rule’s pre-defined type (this will be explained later on)
- argument - argument relevant to the current rule. Could be empty for rule types that don't require any arguments.
- classification - the type of network device that will match this rule

Communications (csv)
id (int), timestamp (int), device_id (text), protocol_name (text), host (text)

- id - the event id
- timestamp - seconds since epoch
- device_id - the device that the event talks about
- protocol_name - the protocol that triggered this event
- host - a domain or an IP address related to the event. Could be empty for event types that don’t need it

Classifications (csv)
id (int), device_id (text), classification (text)

- id - the sequential ID of the line, beginning with 1
- device_id - the device that the classification is about
- classification - the chosen classification

Tasks

1. Add support for rule type ‘communicating_protocol’ that receives the protocol_name to match as an argument in the ‘argument’ field
    a. Example:
        i. Rule: ‘1,communicating_protocol,http,user endpoint’
        ii. On Communication: ‘1,1578455846,aaaa,http, 10.0.0.1’
        iii. Will Output: ‘1,aaaa,user endpoint’

2. Add support for rule type ‘communicating_with’ that receives in the ‘argument’ field the IP address to match as an argument
    a. Example:
        i. Rule: ‘1,communicating_with,10.1.1.1,ct’
        ii. On Communication: ‘1,1578455846,aaaa,ssl, 10.1.1.1’
        iii. Will Output: ‘1,aaaa,ct’

3. Add support for rule type ‘communicating_with_subnet’ that receives the subnet to match as an argument in the ‘argument’ field
    a. Example:
        i. Rule: ‘1,‘communicating_with_subnet’,10.1.1.0/24,ct’
        ii. On Communication: ‘1,1578455846,aaaa,ssl, 10.1.1.8’
        iii. Will Output: ‘1,aaaa,ct’

4. Add support for rule type ‘communicating_with_domain’ that receives the domain to match as an argument in the ‘argument’ field
    a. Example:
        i. Rule: ‘1,communicating_with_domain,www.apple.com,iphone’
        ii. On Communication: ‘1,1578455846,aaaa,ssl, 10.1.1.8’ (www.apple.com lookup returned 10.1.1.8)
        iii. Will Output: ‘1,aaaa,iphone’

Bonus

5. Process communications concurrently (!)
6. Add support for rule type ‘multi_rules’ that receives a list of rule ids (separated with ‘-’) to match as an argument in the ‘argument’ field
    a. Example:
        i. Rule:
        ‘1,communicating_with,10.1.1.1,ct
        ‘2,communicating_protocol,ssl,user endpoint’’
        3,multi_rules,1-2,mri’
        ii. On Communications:
        ‘1,1578455846,aaaa,http, 10.1.1.1
        2,1578455846,aaaa,ssl, 20.1.1.1’
        iii. Will Output: ‘1,aaaa,mri’

Notes:

1. The system should be always on and process the communications line by line
2. If a single device_id received more than one classification, the last one (by communication id and than by rule id) will be the one selected
3. If a device will have no classification we should output ‘unknown’

General Guidelines:

1. Pay attention to the quality of your code (design, test, clean code)
2. You can make any assumptions you understand, add comments explaining your thoughts
3. You can use the internet
4. You can assume the input is valid