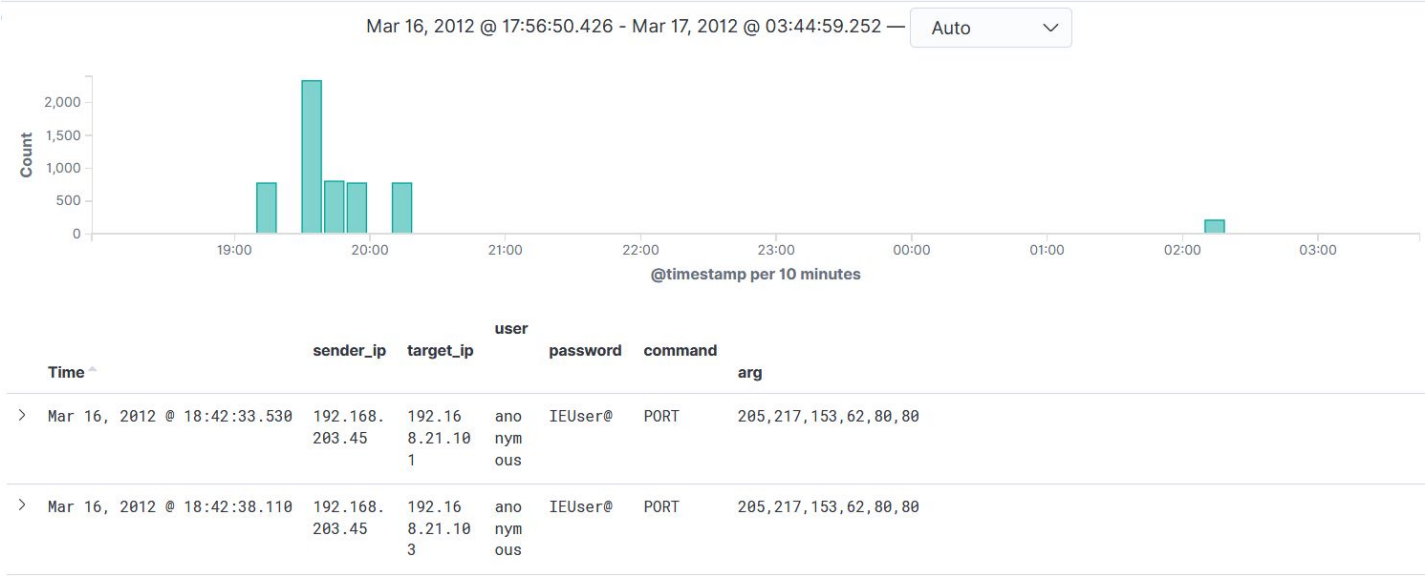
ATTACK DEFENSE

by PentesterAcademy

Name	Kibana : FTP Log Analysis
URL	https://attackdefense.com/challengedetails?cid=1231
Type	Log Analysis: Other Tools

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

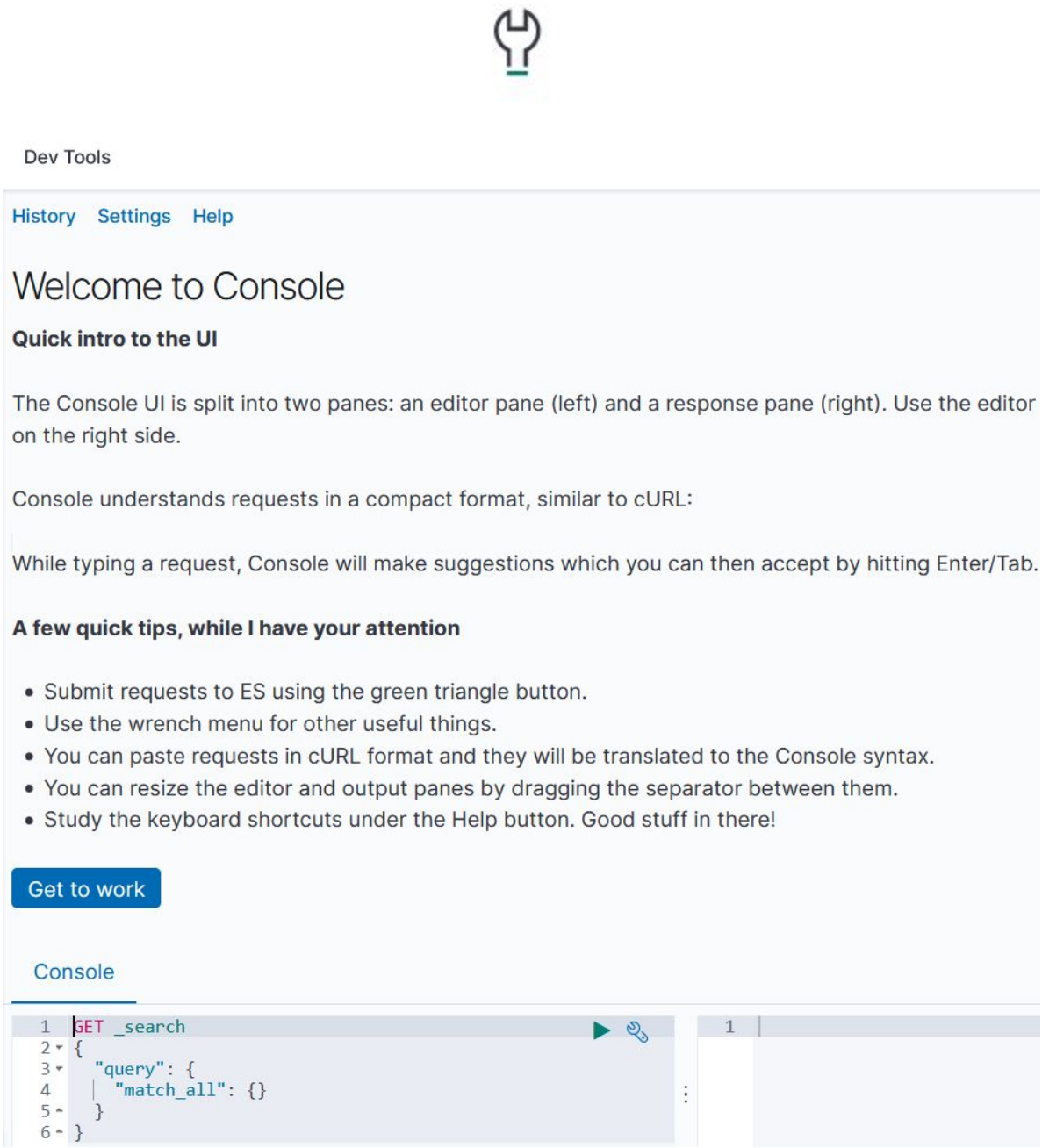
Kibana Dashboard:



Note: All the questions are solved using elasticsearch queries and aggregations.

Kibana provides a console to interact with elasticsearch and perform queries and aggregations.

The console is available under the "Dev Tools" section on the left panel.



The screen shown above appears when the Dev Tools section is accessed for the first time.

The "Console" window on this page could be used to interact with elasticsearch and issue queries and aggregations.

Q1. Determine the count of the client machines.

Answer: 15

Solution:

Step 1: Use the following query to get the count of client machines.

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "aggs": {
          "dintinct_client": {
          "cardinality": {
          "field": "sender_ip.keyword"
          }
          }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "aggs": {
    "dintinct_client": {
      "cardinality": {
        "field": "sender_ip.keyword"
      }
    }
  }
}
```

Response:


```
{
  "took" : 5,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 5796,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "dintinct_client" : {
      "value" : 15
    }
  }
}
```


There were 15 client machines.

Q2. Determine the count of the FTP servers.

Answer: 21

Solution:

Step 1: Use the following query to get the count of the FTP servers.

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
          "match": {
          "target_port": "21"
          }
  },
  "aggs": {
          "dintinct_target": {
          "cardinality": {
          "field": "target_ip.keyword"
          }
          }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
    "match": {
      "target_port": "21"
    }
  },
  "aggs": {
    "dintinct_target": {
      "cardinality": {
        "field": "target_ip.keyword"
      }
    }
  }
}
```

Response:


```
{
  "took" : 108,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 5796,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "dintinct_target" : {
      "value" : 21
    }
  }
}
```


There were 21 FTP servers.

Q3. What was the IP address of the most active client machine?

Answer: 192.168.202.102

Solution:

Step 1: Use the following query to get the IP address of the most active client machine.

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "aggs": {
          "most_active_client": {
          "terms": {
          "field": "sender_ip.keyword",
          "size": 1
          }
          }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "aggs": {
    "most_active_client": {
      "terms": {
        "field": "sender_ip.keyword",
        "size": 1
      }
    }
  }
}
```

Note 1: The size parameter is used to limit the number of results returned by the query.

Note 2: By default, if no query is specified, all the documents are returned, so setting the size parameter to zero prevents displaying the returned document.

Note 3: The size parameter in the aggregation is set to 1 to get the IP address of the most active client machine. In case the size parameter is omitted, the aggregation would return the active client IP addresses in the descending order based on their number of occurrences in the logs.

Response:


```
{
  "took" : 2,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 5796,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "most_active_client" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 318,
      "buckets" : [
        {
          "key" : "192.168.202.102",
          "doc_count" : 5478
        }
      ]
    }
  }
}
```


The machine having IP address "192.168.202.102" was the most active client.

Q4. In how many log instances did any client machine logged into the FTP servers anonymously?

Answer: 5743

Solution:

Step 1: Use the following query to get the count of the log instances where the client either used "ftp" or "anonymous" as the username.

Query:

```
GET ftp-logs/_count
{
  "query": {
          "bool": {
          "should": [
          {
          "match": {
          "user.keyword": "ftp"
          }
          },
          {
          "match": {
          "user.keyword": "anonymous"
          }
          }
          ]
          }
  }
}
```

Console

```
GET ftp-logs/_count
{
  "query": {
    "bool": {
      "should": [
        {
          "match": {
            "user.keyword": "ftp"
          }
        },
        {
          "match": {
            "user.keyword": "anonymous"
          }
        }
      ]
    }
  }
}
```

Note: The query makes use of the count API (/_count) instead of the search API (/_search).

Response:

```
1  {
2    "count" : 5743,
3    "_shards" : {
4      "total" : 1,
5      "successful" : 1,
6      "skipped" : 0,
7      "failed" : 0
8    }
9  }
10
```

There were 5743 log instances where any client machine logged into the FTP servers anonymously.

Q5. A client machine logged into an FTP server using a password ending with '.edu'. What was the username used by that client?

Answer: anonymous

Solution:

Step 1: Use the following query to list the usernames of the clients whose passwords end with ".edu".

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
        "wildcard": {
        "password": {
        "value": "*.edu"
        }
        }
```

```
  },
  "aggs": {
        "usernames": {
        "terms": {
        "field": "user.keyword"
        }
        }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
    "wildcard": {
      "password": {
        "value": "*.edu"
      }
    }
  },
  "aggs": {
    "usernames": {
      "terms": {
        "field": "user.keyword"
      }
    }
  }
}
```

Note: Instead of using a wildcard query, a regexp query could also be used.

Response:


```
{
  "took" : 14,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "usernames" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "anonymous",
          "doc_count" : 4
        }
      ]
    }
  }
}
```


The client having the password ending with ".edu" used "anonymous" as the username.

Q6. Which client machine sent the maximum number of append commands to the FTP servers? Analyze all such requests and try to figure out if there was something odd about those requests.

Answer: 192.168.202.102

Solution:

Step 1: Use the following query to get the IP address of the client that had sent a maximum number of "APPE" commands to the FTP server.

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
          "match": {
          "command.keyword": "APPE"
          }
  },
  "aggs": {
          "clients": {
          "terms": {
          "field": "sender_ip.keyword",
          "size": 1
          }
          }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
    "match": {
      "command.keyword": "APPE"
    }
  },
  "aggs": {
    "clients": {
      "terms": {
        "field": "sender_ip.keyword",
        "size": 1
      }
    }
  }
}
```

Response:


```
{
  "took" : 2,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 72,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "clients" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "192.168.202.102",
          "doc_count" : 72
        }
      ]
    }
  }
}
```


Note: There were 72 document hits for the query and same number of hits for the aggregation. That means that there was only one client machine, having IP address "192.168.202.102", that sent all the "APPE" commands to the FTP server.

Step 2: Use the above query and set the size parameter to 72, which was the number of document hits and analyze the returned documents.

Query:

```
GET ftp-logs/_search
{
  "size": 72,
  "query": {
            "match": {
            "command.keyword": "APPE"
            }
  },
  "aggs": {
            "clients": {
            "terms": {
            "field": "sender_ip.keyword",
            "size": 1
            }
            },
            "commands": {
            "terms": {
            "field": "command.keyword",
            "size": 1
            }
            }
  },
  "sort": [
            {
            "@timestamp": {
            "order": "asc"
            }
            }
  ]
}
```

Note 1: The above query also applies a "terms" aggregation on the FTP commands used by the client.

Note 2: The documents returned by the above query are sorted by the timestamp field.

Console

```
GET ftp-logs/_search
{
  "size": 72,
  "query": {
    "match": {
      "command.keyword": "APPE"
    }
  },
  "aggs": {
    "clients": {
      "terms": {
        "field": "sender_ip.keyword",
        "size": 1
      }
    },
    "commands": {
      "terms": {
        "field": "command.keyword",
        "size": 1
      }
    }
  },
  "sort": [
    {
      "@timestamp": {
        "order": "asc"
      }
    }
  ]
}
```

Response:


```
{
  "took" : 26,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 72,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [
      {
        "_index" : "ftp-logs",
        "_type" : "_doc",
        "_id" : "ZfcSkGwBWANxU_WfrGYa",
        "_score" : null,
        "_source" : {
          "arg" : """ftp://192.168.21.152/./\x83\xc7
            <\xbe\xf0]\xbd\xde\x87\xf73\xc9\xb1\xc8\xf3\xa4\x8b\xdf\x8b|$L\xb8Buno\xa
            \x90\x90\x90\x90\x90\x90Auno\xeb^C_W\xc3\xe8\xf8\xff\xc7^KGGGG\x817\xb1\x
            8c\x83\x8c\xd1\xd3\xdd\x82\xf2\xda\xc5\xf2\xc7\xd0\xd2\xe3\xd1\xdb\xc3\xc
            xc6\xd1\x87\xe0\x82\x90\x9b\xdb\xdd\xce\xb5\xe2\xd5)\xa8\x8f\xb1\xa2\xb54
            <P\xdb\xaf\xec\xd9\x1c\xae\x95^Q\xc4KWH
            :\xd1\xad2\xf4\xaf\xe5\xd5\xbfJ\xad\xbf\xb1\xa2\xe52\xf4\xb9\xe5\xd5\xbdJ
            2\x84^?\x80y<\xfaA)\xf3\x83:\xe6\xb3\xc7\xb0R6^?\xa9)\xa56\xe4V<M:\xea\xb
            :\xda\xbdAAAAAAAA\xbb@@\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x
            \x90""",
          "target_ip" : "192.168.21.152",
          "sender_port" : "2487",
          "conn_id" : "Cvul862RaUW7CABdOl",
          "command" : "APPE",
          "sender_ip" : "192.168.202.102",
          "reply_code" : "530",
          "target_port" : "21",
          "@timestamp" : "2012-03-16T13:59:40.280Z",
          "user" : "anonymous",
          "password" : "Cuno",
          "reply_msg" : "Login or password incorrect!"
```

```
656 {
657   "_index" : "ftp-logs",
658   "_type" : "_doc",
659   "_id" : "KvcSkGwBWANxU_Wfr245",
660   "_score" : null,
661   "_source" : {
662     "arg" : """ftp://192.168.22.152
        /./\x81\xecX\xfb\xffAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAA\xda^@\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90
        0\x90\x90\x90\x90\x90\xe8\xff\xff\xc3_\x83\xc7^EWYIIIIIIIIIQZVTX30VX4AP0A3HH0A00ABA
        p2dnpdplpla0pGpeptpPprpopcaQpdpdprpepspspLpopapdpLpipbprpaprpyaQpwpsp2aOp3p2dnpdplpl
        1hLlKpsp0k9hopjdMpYpflmdLa0lnpuhihbhwlKpsb8lMpEdMpPpjdNhhb8c0c0c0pPlMpEeKpPpjdLhhdLc
        pEhplKpFellKpDc6pxc1hpk3iPb8lKb0k9pUhtk9hrlKpHc4c1hQk9pMillKpxc8lKaPdLc1hPc1hWk9pEhx
        flKc4pZdOmohPlKc4j1c1hpk9pEhphkc6pCekaMhtprhTk9hlpalKpDa4hPhBc8c0apk9hepRpVpWpjc2pjp
        phpPprpopcphptpipopnphpoprpmpaphptpIpnpfphpNptpSpepTpPiopTa4p8pPlMpLa4p0pjc4pQpja2pj
        0dmi2c5c0c0pPpRpVpWpjc0phpepspsc0phpPprpopcphpOpppepnpTiopta4b8iopTa4b8pPphpEpxc0c0p
        pPphprpyc0c0phpMpepmpophpcpepspsphpepPprpophpWprpiptpTiopta4pDiopTa4pDpPphpapdc0c0ph
        LiopTa4aLpPpjc0phpdplpec0phpepHpapnphpCplpopspTiopta4ppiopTa4pppPpjc2pjc0pjc0pjc1pjc
        pPpjaQphpaplpupephpepgpepVphpipvpiplphpupppPprphpLpopopkpTpPioj4a4h0c0c0c0pPpjc0phpl
        HpRpQpjc0iohPpPpjc0phpopkpepnphpepspspTphpPprpopcphpOpppepnpTiopta4pLioj4a4ipc0c0c0p
        phpkpepnpPphpsptpTpophaQpdpjpupTiopta4phioj4a4ilc0c0c0pPlMpLa4alpjc0pjc0pjdLpQpjc0io
        0c0pPk5iPpuc4lKhepahCphpeaQc0c0phpepNpapmphpepBpapsphpopdpuplphpGpeptpMpTiopta4b8ioj
        c1c0c0pPpjc0phpipcpmppphaOpsptprpTpPioj4a4pDc1c0c0pPpjc0k9hgk1hlc0b0c0c0k9hgp1hIpfh9
        upmpTioh4a4pTb0c0c0ioj4a4apb1c0c0pPlMpLa4b8lMpTa4b4pRphc0b0c0c0pQiohPlMpta4b8k3ala4b
        a4pXb1c0c0pPk1hlc0b0c0c0k9ipphc0b0c0c0pPpjc0ioh4a4dLb0c0c0ioj4a4pTa0c0c0k5iPptaNpjc0
        0aneolKpDa4c4lKc0lMpLa4eLpQpPioj4a4pHa0c0c0k3hlhxk5iPpub8lKk4a4a0b0c0c0lKpTa4c0hwhZk
        k3hlhdioh4a4c0b0c0c0ioj4a4ila0c0c0lMk4a4eLb0c0c0k3a8c4k1hlhxhoioiohielioioiolMh4a4b8
        4c0pjc0pjdjioj4a4apb1c0c0pPk5iPpuc5hihbc0c0c0pjaPphc0b0c0c0phlJc2c0c0pjc0iopta4b0ioj
        c0c0ioh4a4k8b1c0c0iopta4b0iopta4b8ioj4a4pHb1c0c0pjc0lKpDa4c8c5pBc1c0c0k9hapQphpHc1c0
        3pjc0pjc0pjc0iopta4b4pjc0pjc0iopta4a4ioj4a4elb1c0c0pPk3ala4c0c0ptdOlMpTa4dLp1iPk9c2lM
        pepepEphpupaplpFphpVpiprptpTioh4a4k8b1c0c0ioj4a4k8b1c0c0pPphc0k0c0c0pjc0iopta4eLiopta
        0phpPprpopcphpEpxpiptpTioh4a4k4b1c0c0ioj4a4k4b1c0c0pPpjc0iohPk3hlhliopLa4dLk3hlhthil
        ptphpWpSaQpSphpeptc0c0phpspopcpkphpepcptc0phpcpopnpnlKiLpSpQk3hCc8pSpQk3hCc8pSpQlKiL
        jc6pjc1pjc2iohPlKhhiohWpjc0pjc0phiPi8hJpfphc2c0iQpzlKiLpjb0pSpUiohPk3hDdlj5pYpZpQpRp
        hWj3iohWj5k3hDb0pXpZpYpSpUpPlKhlpPpWpRpVpjc4pZhkdNaKhhpUc0c0c0c1hSaNpZaOpXiohchhhmio
        hplplpopcphpupaplaQphpVpiprptpTpViohWk3hDb0pZpRpjaPphc0b0c0c0pRpjc0iohPk9pDa4a4pZj3h
        p3iPhhdJc0c0c0aKpZc3hXdkhPpuhopahCp3hIpQpRpSiopuc0iopTc5c4k5iPanc1hCphpapdc0c0phpTph
663     "target_ip" : "192.168.22.152",
664     "sender_port" : "4855",
665     "conn_id" : "CmeBJ43753uLXmjkke",
666     "command" : "APPE",
```

```
1595   "aggregations" : {
1596     "clients" : {
1597       "doc_count_error_upper_bound" : 0,
1598       "sum_other_doc_count" : 0,
1599       "buckets" : [
1600         {
1601           "key" : "192.168.202.102",
1602           "doc_count" : 72
1603         }
1604       ]
1605     },
1606     "commands" : {
1607       "doc_count_error_upper_bound" : 0,
1608       "sum_other_doc_count" : 0,
1609       "buckets" : [
1610         {
1611           "key" : "APPE",
1612           "doc_count" : 72
1613         }
1614       ]
1615     }
1616   }
1617 }
```


Note: In all the log events returned, "APPE" was command used.

The "arg" field of the returned documents contains hexadecimal characters in the request (probably some shellcode was used) to perform buffer overflow attack on the FTP servers. This could further be confirmed by checking the length of the successive "arg" values. The length of the payload varies in the successive requests and also, the second screenshot above contains a very large string, that was possibly used for identifying the correct offset value.

Q7. For how many connection requests the username or password was incorrect?

Answer: 9

Solution:

Step 1: Use the following query to get the logs instances where the reply message field contained "incorrect" keyword. The returned documents are aggregated on the basis of the client IP addresses and the reply message field.

Query:

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
            "match": {
            "reply_msg": "incorrect"
            }
  },
  "aggs": {
            "clients": {
            "terms": {
            "field": "sender_ip.keyword"
            }
            },
            "reply_msg": {
            "terms": {
            "field": "reply_msg.keyword"
            }
            }
  }
}
```

Console

```
GET ftp-logs/_search
{
  "size": 0,
  "query": {
    "match": {
      "reply_msg": "incorrect"
    }
  },
  "aggs": {
    "clients": {
      "terms": {
        "field": "sender_ip.keyword"
      }
    },
    "reply_msg": {
      "terms": {
        "field": "reply_msg.keyword"
      }
    }
  }
}
```

Response:

```
{
  "took" : 8,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 9,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
```

```
18    "aggregations" : {
19      "clients" : {
20        "doc_count_error_upper_bound" : 0,
21        "sum_other_doc_count" : 0,
22        "buckets" : [
23          {
24            "key" : "192.168.202.102",
25            "doc_count" : 9
26          }
27        ]
28      },
29      "reply_msg" : {
30        "doc_count_error_upper_bound" : 0,
31        "sum_other_doc_count" : 0,
32        "buckets" : [
33          {
34            "key" : "Login or password incorrect!",
35            "doc_count" : 9
36          }
37        ]
38      }
39    }
40  }
```


There were 9 connection requests where the username or password was incorrect.

Q8. An image was downloaded from an FTP server. Provide the complete URL of that image.

Answer: ftp://192.168.25.101/dept/qdept/static/images/biohazard.jpg

Solution:

Step 1: Use the following query to get the documents having MIME type: "image/*".

Query:

```
GET ftp-logs/_search
{
  "size": 1,
  "query": {
            "regexp": {
            "mime_type": "image/*"
            }
  },
  "_source": "arg"
}
```

Console

```
GET ftp-logs/_search
{
  "size": 1,
  "query": {
    "regexp": {
      "mime_type": "image/*"
    }
  },
  "_source": "arg"
}
```

Note: The query uses the "_source" parameter to retrieve only the "arg" field of all the matched documents.

Response:


```
{
  "took" : 8,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 1,
      "relation" : "eq"
    },
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "ftp-logs",
        "_type" : "_doc",
        "_id" : "F_cSkGwBWANxU_WfsHbD",
        "_score" : 1.0,
        "_source" : {
          "arg" : "ftp://192.168.25.101/dept/qdept/static/images/biohazard.jpg"
        }
      }
    ]
  }
}
```


There was only one request in which an image file was downloaded from an FTP server.

The image was downloaded from the URL:
"ftp://192.168.25.101/dept/qdept/static/images/biohazard.jpg".

References:

1. ELK Stack (https://www.elastic.co/elk-stack)
2. Elasticsearch Reference (https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html)
3. https://www.secrepo.com/maccdc2012/ftp.log.gz
   (Security Repo maintained by Mike Sconzo and licensed under Creative Commons Attribution 4.0 International License)