Pilot Log Anonymization in GlideinMonitor

Mirica Yancey, Valparaiso University – SIST Intern

Introduction

GlideinWMS is a pilot-based resource provisioning tool for distributing HTCondor

- Provides reliable and uniform virtual clusters
- Provisions computers to run scientific computations (analysis, simulations, reconstructions etc.)

GlideinMonitor

Web application to view GlideinWMS's log files

- User interface tool
- Useful for quick searches and decoding log content
- Contains an efficient, managed archive of log files

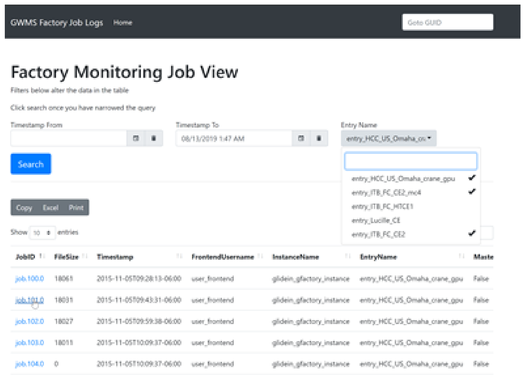

Figure 1 – GlideinMonitor Homescreen

Problem

- Log files contain sensitive information that makes the logs less secure and decreases the ease of distribution amongst developers.

Objective

Create a script that:

- Locates the job submitter's IP Address
- Locates any personal identifiers of the job submitter
- Filters/Replaces those values with non-identifiable information

In order to:

- Increase log use diversity for developers
- Protect user information.

Methods

- Identified what the personal information was and then cross referenced files to ensure there was a consistent reference for this information in order to search and retrieve it.

```
def findEmail(filename): #CONDOR SPECIFIC finds and returns user email (before @symbol)
    lis = ''
    with open(filename, 'rb', 0) as file, \
        mmap.mmap(file.fileno(), 0, access=mmap.ACCESS_READ) as s:
        if s.find(b'x509UserProxyEmail') != -1:
            x = s.find(b'x509UserProxyEmail')
            end = s.find(b'@',x)
            lis = (((s[x: end].split(b'='))[1].decode("utf-8")).replace('"', '')).replace(' ', '')
    return lis
```

Figure 2 – Email Identifier Script

```
def findUserIds(filename): #CONDOR SPECIFIC finds and returns user email (befo
    lis = ''
    with open(filename, 'rb', 0) as file, \
        mmap.mmap(file.fileno(), 0, access=mmap.ACCESS_READ) as s:
        if s.find(b'x509UserProxyFQAN = "') != -1:
            x = s.find(b'x509UserProxyFQAN = "')
            end = s.find(b'Role=',x)
            lis = s[x: end].decode("utf-8")#
            lis = lis.split('CN=')
            lis.pop(0)
            lis = " ".join(lis).replace("/","").replace(",cms","").split(' ')
    return lis
```

Figure 3 – User Information Identifier Script

- Created a method to replace this information and ran unit tests against multiples files of varying file types to insure quality.

```
def cleanLogs():
    parser = argparse.ArgumentParser(description="GlideinMonitor's Filtering")    # Parse command
    parser.add_argument('-i', help="Input Directory", required=True)
    parser.add_argument('-o', help="Output Directory", required=True)
    args = parser.parse_args()
    input_directory = args.i
    output_directory = args.o

    input_directory_files = [file for file in os.listdir(input_directory)
                             if os.path.isfile(os.path.join(input_directory, file))]

    for file_name in input_directory_files:
        path = os.path.join(input_directory, file_name)
        if os.path.splitext(file_name)[1] == '.out' or os.path.splitext(file_name)[1] == '.err' :
            print("Cleaning Glidein!")
            cleanGlidein(path)
        elif condorFile(file_name) == True:
            print("Cleaning Condor!")
            cleanCondor(path)

        out_path = os.path.join(output_directory, file_name)

        f_in = open(path, "r")
        with open(out_path, 'w') as file:
            output_file_contents = f_in.read()
            f_in.close()
            file.write(output_file_contents)
            print("Wrote to outfile!")

        os.remove(os.path.join(input_directory, file_name))
        print("Removed Old File!")

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

Figure 4 – Main Filtering Method

```
You, a few seconds ago | 1 author (You)
class TestFilter(unittest.TestCase):
    def test_glidein(self): #Glidein Logs
        file1 = "regular_logs/overall_test_in.txt"
        user_ids2 = findGlideinUserIDs(file1)
        cleanGlidein(file1)
        file = open(file1, "r", encoding="utf8")
        line = file.read()
        file.close()

        self.assertNotRegex(line, IP_REGEX[0])
        self.assertNotRegex(line, IP_REGEX[1])
        self.assertNotRegex(line, rf"{user_ids2[0]}[\w \W]{user_ids2[1]}")
        for x in range(0, len(user_ids2)):
            self.assertFalse(line.find(user_ids2[x]) != -1)
```

Figure 5 – Glidein Unit Test

Results

- Filtered logs contain no personal identifiers
- Script integrated into GlideinMonitor for further testing

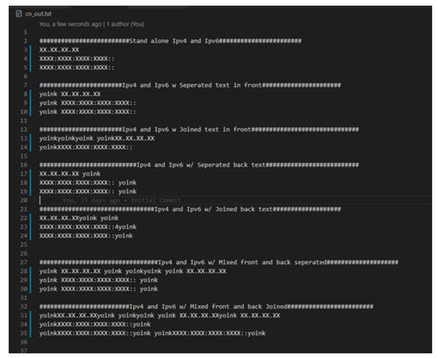

Figure 6 – Filtered Test File

Figure □ – Filtered I ddress in Log File

Figure □ – Filtered User Identifiers in Log File

Acknowledgements

Thank you to my supervisor Marco Mambelli , the GlideinWMS team, and my SIST Mentoring team for consistently making me feel welcome and heard. Thank you for the belief in my skill set and the time you dedicated towards helping me grow that set. In addition to that, thank you Fermilab for the amazing opportunity you provided to me and the other interns this summer.