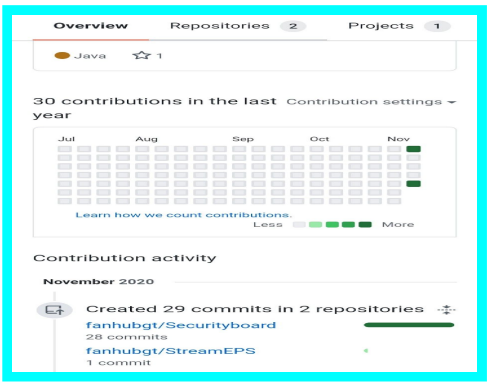
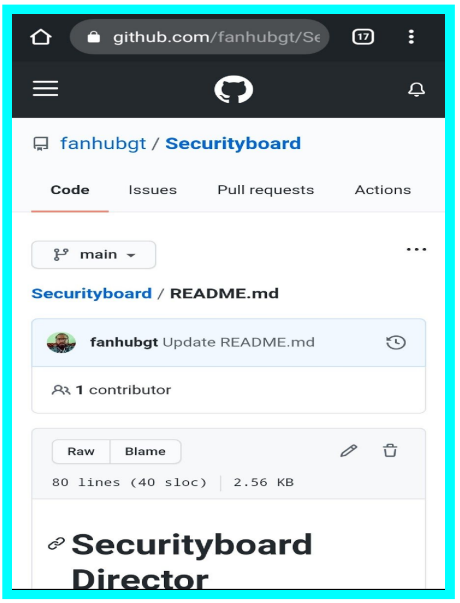
Poster on Security-board Director Research Project at Github.

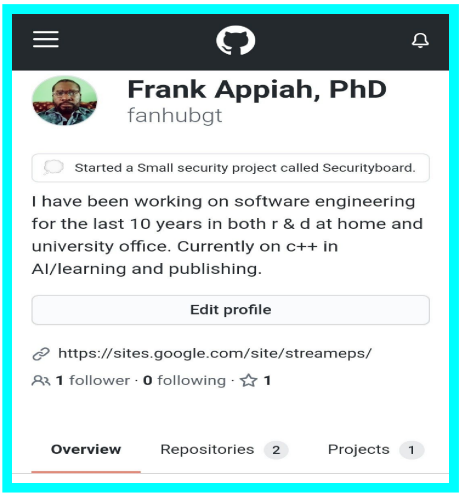




Prof Frank Appiah AKA FAEng PhD
Affiliate: King's College London, Strand, London, England, UK.
Email:appiahnsiahfrank@gmail.com
12.2020.11.

Abstract. A small prolog program[1] for security board director[2] used to make countermeasure check on vulnerability prevention is hosted at Github with name *fanhubgt stroke Securityboard* is the main focus of this poster.

Keywords. Prolog, logic program, security board, vulnerability prevention.



Introduction.

There are 3 main types of logic programming the same interpreter with methods like forward-backward chaining, case-based approach and predicate sentences on facts.
1. **director.pl** uses a forward-backward chaining method. This method has each rule having different head name and each rule calls the other after execution in chain with the main rule.

[1] **# Further Reading**
Search for author name Frank Appiah at easychair.org or Google scholar.
Four published articles are on display at the site.
The first is on security reasoning and the other three on the Prolog files.

[2] Appiah, Frank. "Security Controls or Countermeasures: Vunerabilities Prevention." Easychair Preprint 4410 (2020).

Appiah, Frank. "Open source project called Securityboard". Hosted at Github.com. Web access, https://github.com/fanhubgt/Securityboard/blob/main/README.md 2020.

After each differential rule calling a different headrule is invoked until the last rule is runs. Then the main, first headrule which is invoked by the Prolog Interpreter is called again in the process. As result causing all the headrules to be chained.

2. **directorcase.pl** is based on case approach. Each head rule has the same name as the other but with a differential increments of integer value starting from 1 to the the total number of headrules desired. Here, we are looking at 12 in total. The
Heads has the name cdd(1) to cdd(12). After each similar headrule runs, then the main is invoked again to bring a menu selection for any other head rule in any order.

Here the choices on selection are 1, 4, 7, 6. It runs as follows:

Cdd(1) can run after main menu shows up.
Cdd(4) can run after main menu shows up again.
Cdd(7) can run after main menu shows up.
Cdd(6) can run after main menu runs again.

3. **assertcounter.pl** is based on consult approach using a fact loader of predicate sentences. Consult is a builtin rule of Prolog. Using consult(assertcounter.pl), it will load the facts in the file into its database.
After, writing on the prompt of the Prolog Interpreter will invoke a Yes or no response in checking for assert of fact in the database. If yes then a fact is checked as member of the database.

```
?- cm_check(invalid,info).
yes.
? - cm_check(passby,riot).
yes.
?- cm_check(unlawful,entry).
yes
```

```
? - cm_check(intern,replacement).
yes
?- cm_check(access,system_codes)
yes
```

This is a typical way of getting yes responses from the prompt of Prolog execution.
Now no responses.

```
?- cm_check(interns,replacements).
no
%made arguments Plural throw no response.
?- cm_check(access,
error throws
 % incomplete with closing brackets and missing second argument[3].
```

```
? - cm_check(passthrough,riots).
no
% passby is now pass-through, also riot is riots.
```

[3] **#Reference**

(1) XProlog. Android IDE for Logic Programming. Playstore. 2020.
(2) Github. Online Repository. Web access at Github.com. 2012.

Appiah, Frank. "Open source project called Securityboard". Hosted at Github.com. Web access, https://github.com/fanhubgt/Securityboard/blob/main/README.md 2020.