توضیحات پروژه EvaluatingCSA

ابتدا ساخت مدل را توضیح می دهیم سپس کد ها را روی مدل ها اعمال می کنیم.

مدل

مدل از سه بخش BPMN، Network و Attack تشکیل شده است.

BPMN

در این بخش باید فرآیند ها در یک کسب و کار مشخص شود همچنین ارتباط بین آن ها و ماموریت های مربوطه به همراه میزان اهمیت هر کدام مشخص شود.

بخش BPMN به 4 بخش ResourcePools، Processes، WorkFlows و Missions تقسیم می شود.

ResourcePools

در این بخش مشخص می شود که به چند Resource Pool نیاز داریم اسم هر کدام از آن ها چیست و هر کدام از آن ها چند Resource دارند و آیا بین Resource Pool ها وابستگی وجود دارد یا خیر و همینطور هر Resource Pool به کدام Subnet در بخش Network متصل می شود.

در شکل زیر مثال این بخش را می بینید.


model1.yml
BPMN:
ResourcePools:
ResourcePoolNumbers: 4
ResourcePool1:
Name: "DB"
ResourceNumbers: 1
Resource1:
Name: "DB"
Dependencies: []
SubnetNumber: 4
ResourcePool2:
Name: "Servers"
ResourceNumbers: 2
Resource1:
Name: "Server1"
Resource2:
Name: "Server2"
Dependencies: [DB]
SubnetNumber: 2
ResourcePool3:
Name: "NginxServer"
ResourceNumbers: 1
Resource1:
Name: "NGINX"
Dependencies: []
SubnetNumber: 1
ResourcePool4:
Name: "SMTPServer"
ResourceNumbers: 1
Resource1:
Name: "SMTP"
Dependencies: [Servers]
SubnetNumber: 3
Processes: <6 keys>
WorkFlows: <2 keys>
Missions: <4 keys>

Processes

در این بخش تمام فرآیند ها معرفی می شوند و این که هر کدام از آن ها مربوط به کدام Resource Pool هستند نیز معلوم می شود.

WorkFlows

در این بخش ارتباط فرآیند ها با یکدیگر مشخص می شود. می توان چندین Path داشته باشیم. در هر Path می توان لیستی از Processes ها و همینطور GateWays ها داشته باشیم. باید ترتیب اجرا را نیز با کلید OrderedKeys داشته باشیم. در هر Path باید مشخص باشد که Priority که میزان اهمیت آن Path را نشان می دهد وجود دارد یا خیر. در GateWays ها نیز می توان چندین Path داشت.

Priority باعث می شود که میزان اهمیت آن Path برای صاحب کسب و کار مشخص شود.

در شکل زیر مثالی از بخش های Processes و WorkFlows قرار داده شده است.

model1.yml

```yaml
BPMN:
  ResourcePools: <5 keys>
  Processes:
    ProcessNumbers: 5
    Process1:
      Name: "ReceiveRequest"
      ResourcePool: NginxServer
    Process2:
      Name: "Authenticate"
      ResourcePool: Servers
    Process3:
      Name: "EmailUser"
      ResourcePool: SMTPServer
    Process4:
      Name: "ProcessData"
      ResourcePool: Servers
    Process5:
      Name: "SendResponse"
      ResourcePool: NginxServer
  WorkFlows:
    PathNumbers: 1
    Path1:
      HasPriority: False
      OrderedKeys: [Processes1, GateWays, Processes2]
      Processes1: [ "Start", ReceiveRequest, Authenticate ]
      GateWays:
        Type: "Parallel" #Exclusive and Inclusive
        Condition: None
        PathNumbers: 2
        Path1:
          HasPriority: True
          OrderedKeys: [Processes1]
          Processes1: [ ProcessData ]
          Priority: 5
        Path2:
          HasPriority: True
          OrderedKeys: [Processes1]
          Processes1: [ EmailUser ]
          Priority: 2
      Processes2: [ SendResponse, "End" ]
```

Missions

در این بخش به هر ماموریت نامی را می دهیم و هر ماموریت از تعدادی فرآیند تشکیل شده است. هر ماموریت نیز Priority دارد.

مجموع تمام Priority این بخش به همراه Priority های بخش WorkFlows عدد Business Importance را تشکیل می دهند.

می توان اهمیت هر فرآیند را نیز حساب کرد بدین صورت که اهمیت هر ماموریت بین فرآیند های مربوط به آن تقسیم می شود. این عمل برای Path های درون WorkFlows نیز تکرار می شود یعنی اهمیت هر Path بین فرآیند های مربوطه تقسیم می شود. در نتیجه ما Process Importance را به ازای هر فرآیند داریم. * (در بخش اصلاحات توضیح بیشتر آورده شده است)

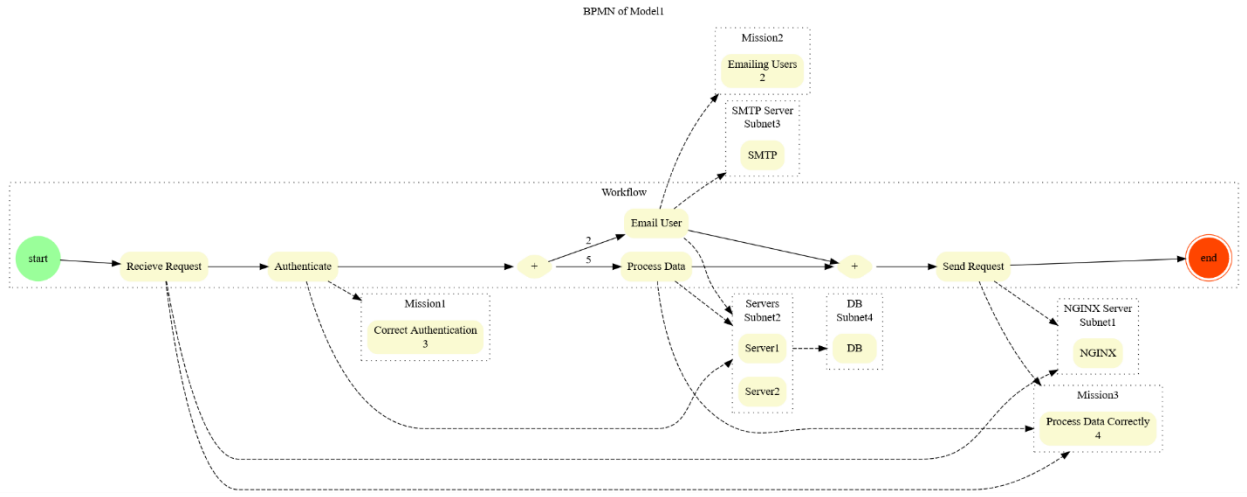
دو بخش Business Importance و Process Importance درون کد ها محاسبه می شوند.

در شکل زیر مثالی از بخش Missions را مشاهده می کنید.

model1.yml

```yaml
BPMN:
  ResourcePools: <5 keys>
  Processes: <6 keys>
  WorkFlows: <2 keys>
  Missions:
    MissionNumbers: 3
    Mission1:
      Name: "Correct Authentication"
      Processes: [Authenticate]
      Priority: 3
    Mission2:
      Name: "Emailing Users"
      Processes: [ EmailUser ]
      Priority: 2
    Mission3:
      Name: "Process data correctly"
      Processes: [ ReceiveRequest, ProcessData, SendResponse]
      Priority: 4
```

در شکل زیر بخش BPMN را به صورت تبدیل شده به گراف مشاهده می کنید. (این بخش در پروژه لازم نیست و فقط برای مشاهده بهتر ایجاد شده است)



Network

در این بخش تعداد Subnet ها و اینکه هر Subnet چند Host دارد و اینکه بین کدام Subnet ها ارتباط وجود دارد (ارتباط از اینترنت به کدام Subnet وجود دارد) بیان می شود.

اینکه هر کدام از Host ها چه آدرسی دارند (عدد اول شماره Subnet و عدد دوم شماره Host)، چه سیستم عاملی در آن وجود دارد و چه سرویس ها و Process هایی در آن اجرا می شوند نیز آورده شده است.

این تنظیمات با توجه به ارتباطاتی که در بخش BPMN داریم باید نوشته شود.

به ازای هر Host عددی را به عنوان SecurityFactor بین صفر تا یک قرار می دهیم که نشاندهنده میزان امن بودن سیستم در برابر تهدید است. عدد نزدیک به یک به معنای مقاوم تر بودن است.

می توان فرمولی به عنوان نسبت تعداد آسیب پذیری هایی که Host می داند و می تواند جلوی خرابی ناشی از آن را بگیرد تقسیم بر تمام آسیب پذیری هایی که وجود دارد را برای این عدد پیشنهاد دهیم اما با این کار باعث می شویم که عواملی که در مدل به آن پرداخته نشده مانند تاثیر آنتی ویروس ها، IDS ها و ... را در نظر نگیریم.

به ازای هر Host با توجه به اهمیت هر فرآیند و ارتباط آن فرآیند ابتدا با ResourcePool مربوطه و سپس با Subnet مربوطه، می توان اهمیت هر کدام را به دست آورد. این کار در کد انجام می شود. * (در بخش اصلاحات توضیح بیشتر آورده می شود)

برای هر Host چهار متغییر دودویی در نظر گرفته ایم.

1- IsCompromised که در ابتدا False است.
2- IsCompromisedCompletely که در ابتدا False است. اگر حمله کننده بتواند به دسترسی root یا administration دسترسی پیدا کند. این متغیر True می شود.

3- isDataLeaked که در ابتدا False است. اگر اطلاعاتی از Host برداشته شود True می شود. می تواند در دو حالتی که IsCompromised، True باشد و زمانی که IsCompromisedCompletely هم True باشد در نظر گرفته شود.
4- IsTerminated اگر حمله کننده تصمیم بگیرد که کلا کارایی یک Host را نابود کند آنگاه این متغیر True می شود. البته ابتدا IsCompromisedCompletely باید True باشد.

به ازای 5 حالتی که برای هر Host وجود دارد ضرایبی را داریم که از اهمیت آن Host برای ارزیابی کم می کند.

1- در حالتی که Host، Terminate شده باشد، تمام ارزش آن Host از بین می رود پس ضریب 1 را در نظر می گیریم.
2- زمانی که Host کاملا Compromised شده باشد و دیتایی نیز استخراج شده باشد ضریب 0.8 است. یعنی بعد از آن حمله ارزش آن Host، 0.2 حالت اصلی می شود.
3- زمانی که Host، Compromised شده باشد و دیتایی نیز استخراج شده باشد ضریب 0.6 است.
4- زمانی که Host کاملا Compromised شده باشد ضریب 0.5 است.
5- زمانی که Host، Compromised شده باشد ضریب 0.3 است.

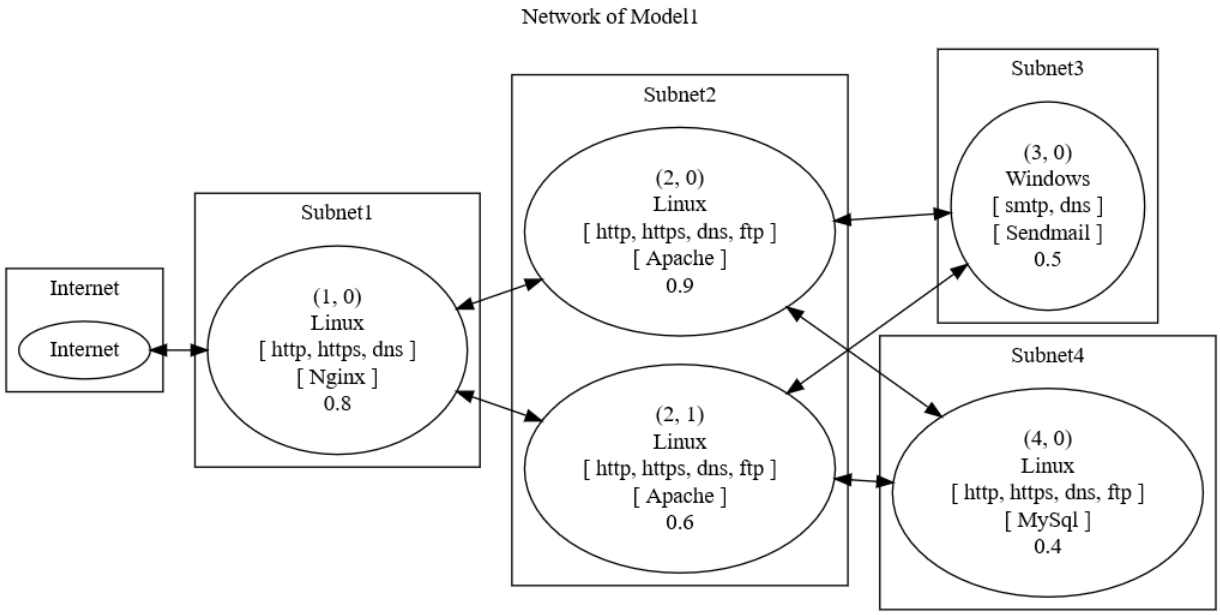
در شکل زیر مثالی از بخش Network را مشاهده می کنید.

model1.yml

```yaml
BPMN: <4 keys>

Network:
  SubnetsNumbers: 4
  Subnets: [ 1, 2, 1, 1 ]
  Topology:
    Internet: [Subnet1]
    Subnet1: [Internet, Subnet2]
    Subnet2: [Subnet1, Subnet3, Subnet4]
    Subnet3: [Subnet2]
    Subnet4: [Subnet2]
  HostConfiguration:
    (1,0):
      Os: linux
      Services: [ http, https, dns ]
      Processes: [ Nginx ]
      SecurityFactor: 0.8
    (2,0):
      Os: linux
      Services: [ http, https, dns, ftp ]
      Processes: [ Apache ]
      SecurityFactor: 0.9
    (2,1):
      Os: linux
      Services: [ http, https, dns, ftp ]
      Processes: [ Apache ]
      SecurityFactor: 0.6
    (3,0):
      Os: windows
      Services: [ smtp, dns ]
      Processes: [ Sendmail ]
      SecurityFactor: 0.5
    (4,0):
      Os: linux
      Services: [ http, https, dns, ftp ]
      Processes: [ MySQL ]
      SecurityFactor: 0.4
```

در شکل زیر گراف مربوط به Network را مشاهده می کنید.



Attack

این بخش از یک گراف تشکیل شده است. به این صورت که در هر Node اطلاعات زیر موجود است.

1- نام Exploit
2- نام آسیب پذیری
3- نام سیستم عامل مربوطه (در غیر این صورت None)
4- نام سرویس مربوطه (در غیر این صورت None)
5- نام Process مربوطه (در غیر این صورت None)
6- آدرس Host ای که مورد حمله قرار گرفته
7- احتمال موفقیت
8- Node بعدی در صورت موفقیت حمله (در صورت اتمام None)
9- Node بعدی در صورت شکست حمله (در صورت اتمام None)

10- حمله در چه مرحله ای قرار دارد از جمله (Initial Compromise، Data Exfiltration، Terminate Node)

نرخ احتمال موفقیت را می توان با فرمول نسبت تعداد Exploit هایی که حمله کننده می داند به تمام Exploit هایی که برای آسیب پذیری های آن Host وجود دارد.

اما بهتر است که فرمول را در نظر نگیریم چرا که مهارت حمله کننده، حمله های روز صفر و ... را با فرمول در نظر نگرفته ایم.

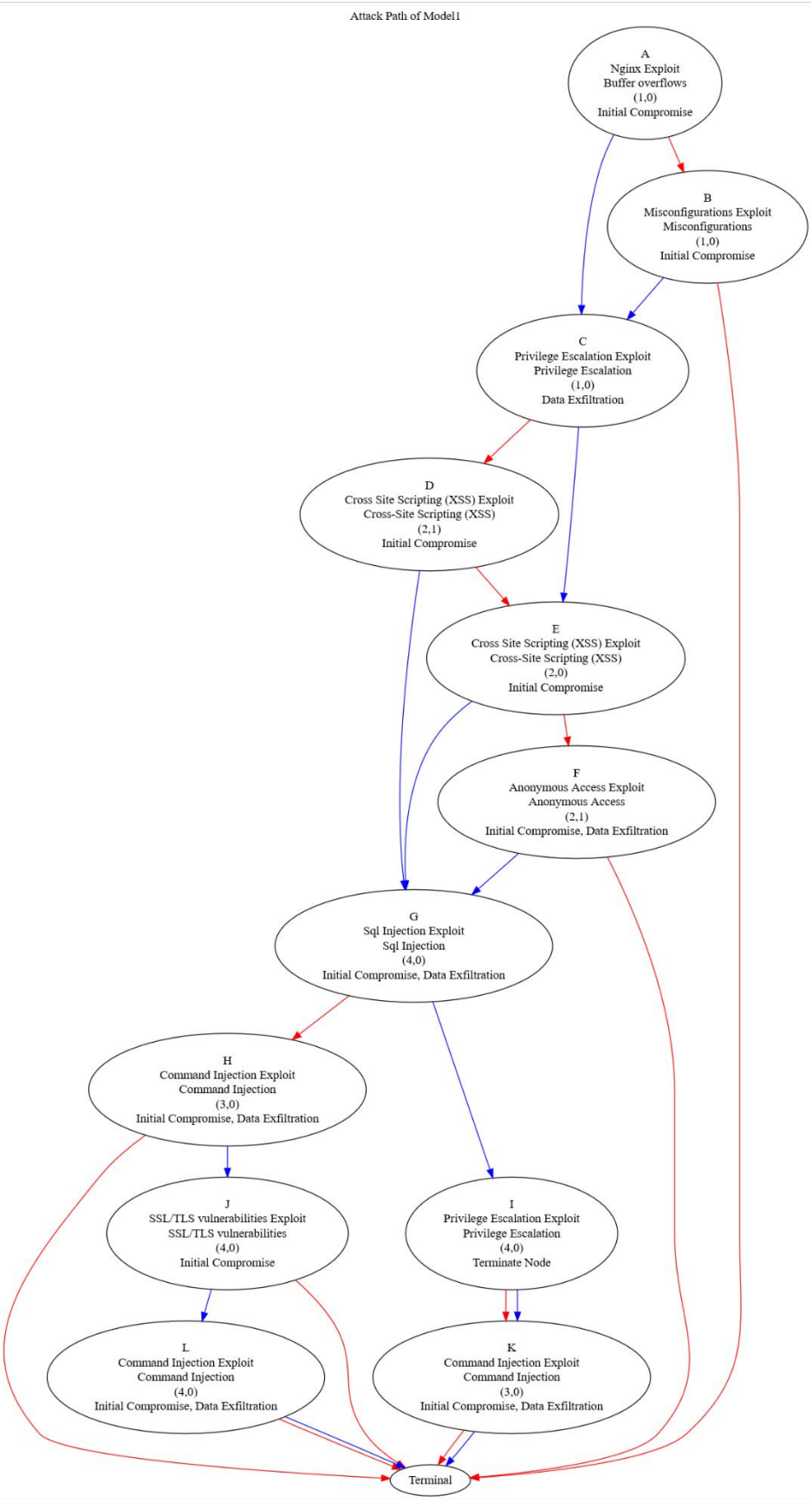
مثالی از چند Node یک Attack را در شکل زیر می بینید.

در شکل بعد از آن مثالی از گراف حمله را مشاهده می کنید.

model1.yml

```yaml
Attack:
  A:
    ExploitName: Nginx Exploit
    Vulnerability: Buffer overflows
    Os: None
    Service: None
    Process: Nginx
    SuccessRate: 0.8
    Target: (1,0)
    AttackStage: [Initial Compromise]
    SuccessPath: C
    FailurePath: B
  B:
    ExploitName: Misconfigurations Exploit
    Vulnerability: misconfigurations
    Os: None
    Service: None
    Process: Nginx
    SuccessRate: 0.7
    Target: (1,0)
    AttackStage: [Initial Compromise]
    SuccessPath: C
    FailurePath: None
  C:
    ExploitName: Privilege Escalation Exploit
    Vulnerability: privilege escalation
    Os: linux
    Service: None
    Process: None
    SuccessRate: 0.6
    Target: (1,0)
    AttackStage: [Data Exfiltration]
    SuccessPath: E
    FailurePath: D
  D:
    ExploitName: Cross-Site Scripting (XSS) Exploit
    Vulnerability: Cross-Site Scripting (XSS)
    Os: None
    Service: http
    Process: Apache
```

Attack Path of Model1



اینکه حمله ای در یک Node رخ می دهد یا خیر با توجه به فرمول زیر مشخص می شود.

احتمال موفقیت در آن Node ضربدر یک منهای Security Factor آن Host مربوطه

حال اگر برای مثال 10000 حمله را شبیه سازی کنیم با توجه به احتمالات حمله به مثال شکل زیر می رسیم. * (در مورد تعداد حمله در بخش اصلاحات توضیحاتی آورده می شود)

model1.yml | AttackPath.txt


```
{'["A", "B", "C", "D", "E", "F", "G", "H", "J", "L", "None:F"]': 5,
 '["A", "B", "C", "D", "E", "F", "G", "H", "J", "L", "None:S"]': 1,
 '["A", "B", "C", "D", "E", "F", "G", "H", "J", "None"]': 25,
 '["A", "B", "C", "D", "E", "F", "G", "H", "None"]': 46,
 '["A", "B", "C", "D", "E", "F", "G", "I", "K:F", "None:F"]': 26,
 '["A", "B", "C", "D", "E", "F", "G", "I", "K:F", "None:S"]': 15,
 '["A", "B", "C", "D", "E", "F", "G", "I", "K:S", "None:F"]': 20,
 '["A", "B", "C", "D", "E", "F", "G", "I", "K:S", "None:S"]': 16,
 '["A", "B", "C", "D", "E", "F", "None"]': 514,
 '["A", "B", "C", "D", "E", "G", "H", "J", "L", "None:F"]': 1,
 '["A", "B", "C", "D", "E", "G", "H", "J", "L", "None:S"]': 1,
 '["A", "B", "C", "D", "E", "G", "H", "J", "None"]': 7,
 '["A", "B", "C", "D", "E", "G", "H", "None"]': 16,
 '["A", "B", "C", "D", "E", "G", "I", "K:F", "None:F"]': 11,
 '["A", "B", "C", "D", "E", "G", "I", "K:F", "None:S"]': 7,
 '["A", "B", "C", "D", "E", "G", "I", "K:S", "None:F"]': 6,
 '["A", "B", "C", "D", "E", "G", "I", "K:S", "None:S"]': 3,
 '["A", "B", "C", "D", "G", "H", "J", "L", "None:F"]': 6,
 '["A", "B", "C", "D", "G", "H", "J", "L", "None:S"]': 2,
 '["A", "B", "C", "D", "G", "H", "J", "None"]': 65,
 '["A", "B", "C", "D", "G", "H", "None"]': 80,
 '["A", "B", "C", "D", "G", "I", "K:F", "None:F"]': 39,
 '["A", "B", "C", "D", "G", "I", "K:F", "None:S"]': 22,
 '["A", "B", "C", "D", "G", "I", "K:S", "None:F"]': 40,
 '["A", "B", "C", "D", "G", "I", "K:S", "None:S"]': 20,
 '["A", "B", "C", "E", "F", "G", "H", "J", "None"]': 4,
 '["A", "B", "C", "E", "F", "G", "H", "None"]': 9,
 '["A", "B", "C", "E", "F", "G", "I", "K:F", "None:F"]': 8,
 '["A", "B", "C", "E", "F", "G", "I", "K:F", "None:S"]': 1,
 '["A", "B", "C", "E", "F", "G", "I", "K:S", "None:F"]': 4,
 '["A", "B", "C", "E", "F", "G", "I", "K:S", "None:S"]': 3,
 '["A", "B", "C", "E", "F", "None"]': 98,
 '["A", "B", "C", "E", "G", "H", "J", "None"]': 1,
 '["A", "B", "C", "E", "G", "H", "None"]': 1,
 '["A", "B", "C", "E", "G", "I", "K:F", "None:F"]': 3,
 '["A", "B", "C", "E", "G", "I", "K:S", "None:F"]': 1,
 '["A", "B", "None"]': 7238,
 '["A", "C", "D", "E", "F", "G", "H", "J", "L", "None:F"]': 9,
 '["A", "C", "D", "E", "F", "G", "H", "J", "L", "None:S"]': 1,
 '["A", "C", "D", "E", "F", "G", "H", "J", "None"]': 36,
 '["A", "C", "D", "E", "F", "G", "H", "None"]': 83,
```


TODO | Python Console | Problems | Terminal | Services

به دلیل آنکه بتوان از روی Attack Path بتوان موفقیت آمیز بودن حمله در Node ها را تشخیص دهیم، زمانی که در صورت شکست و موفقیت در یک Node، Node بعدی یکسان باشد در اسم Node بعدی دو نقطه و F به معنی شکست و S به معنی پیروزی را اضافه کرده ایم.

کد

در کد برای شبیه سازی CSA به عددی با عنوان CSA Correctness نیاز داریم به این معنا که چند درصد مواقع پیشبینی قطعا درست است و چند درصد مواقع از روش دیگری استفاده شده است.

در کد از چهار CSA استفاده کرده ایم که اولی 0.5 و دومی 0.7 و سومی 0.9 Correctness دارند. برای چهارمین CSA از حالت رندوم استفاده کرده ایم به این معنا که برای هر پیشبینی یکبار این عدد به صورت تصادفی انتخاب می شود.

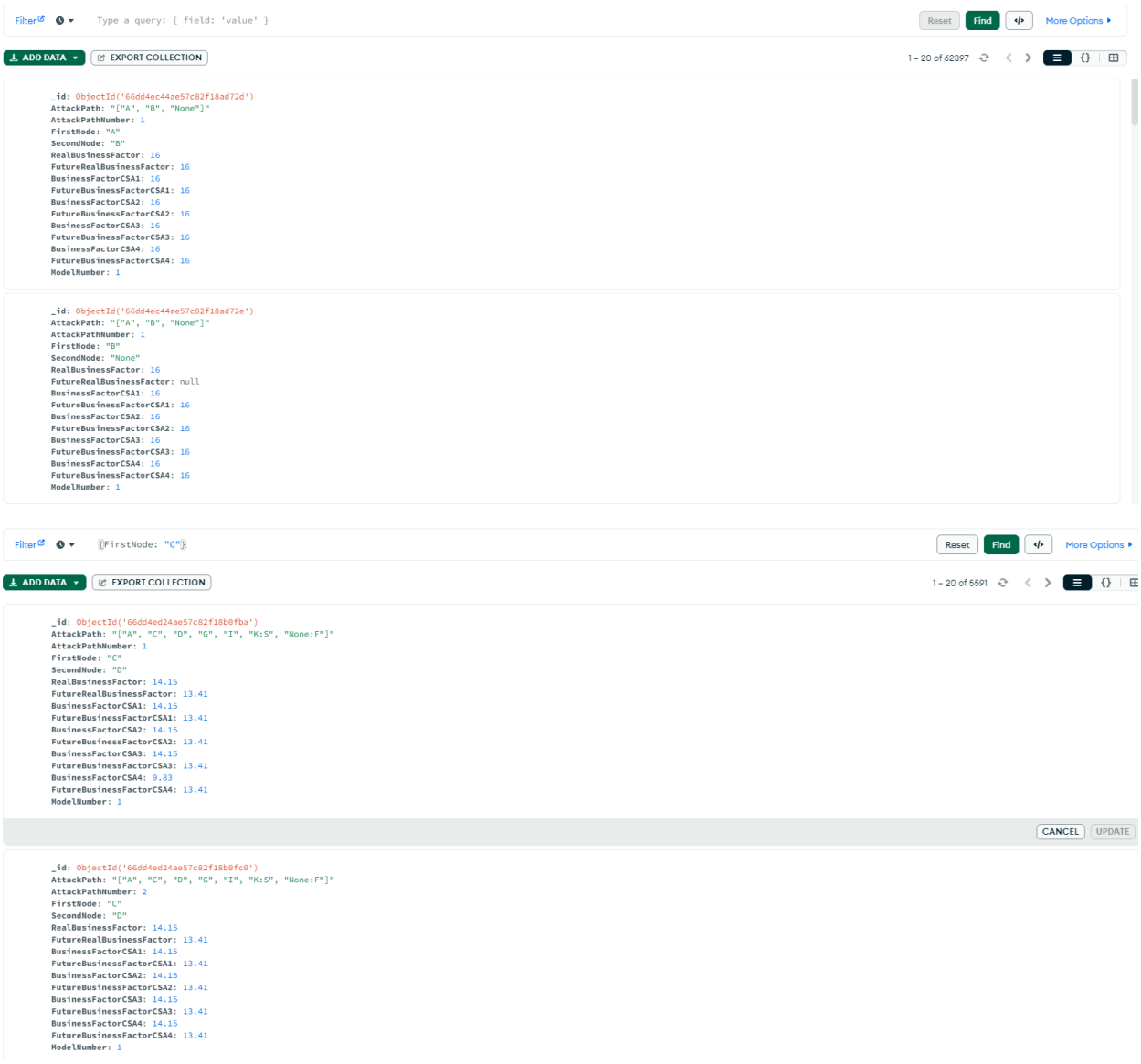
هر CSA در هر حمله دو پیشبینی را انجام می دهد. 1- هنگامی که حمله انجام شده است مشخص می کند که چه تغییراتی در وضعیت Host ها ایجاد شده است. 2- پیشبینی می کند که در صورت حمله بعدی چه تغییراتی در وضعیت Host ها ایجاد می شود.

به هر CSA چهار عدد را به عنوان احتمال می دهیم به صورتی که اولی احتمال رخ دادن حمله دومی احتمال رخ دادن CompleteCompromised و سومی احتمال رخ دادن DataLeakation و چهارمی احتمال Terminate شدن node (اگر از قبل CompleteCompromised شده باشد) را می دهد. که ما برای همه آن ها 0.5 را در نظر گرفته ایم.

در واقع CSA در هر پیشبینی ابتدا با استفاده از اعداد تصادفی مشخص می کند که باید جواب درست را برگرداند یا خیر. اگر عدد تصادفی کمتر از Correctness بود وضعیت درست شبکه را بر می گرداند. در غیر این صورت Host مربوطه را تشخیص داده و با احتمالات بالا وضعیت را بر می گرداند.

بعد از مشخص شدن هر پیشبینی وضعیت Business Importance فعلی شبکه را اعلام می کند.

در شکل زیر دو مثال از دیتاست که این اعداد برای هر چهار CSA آورده شده اند را مشاهده می کنید. ❋ (در مورد فیلد های دیتاست شکل زیر در اصلاحات توضیحاتی ارائه می شود)



حال برای ارزیابی عملکرد یک CSA باید میانگین اختلاف عدد های پیشبینی شده را با اعداد واقعی به دست بیاوریم.

در شکل زیر این اعداد را می بینیم.

model1.yml | CSAResult.txt

```
([{'csa_number': 1,
   'evaluate_future': 0.654498691410897,
   'evaluate_now': 0.7475814928260519},
  {'csa_number': 2,
   'evaluate_future': 0.4042969307637402,
   'evaluate_now': 0.47231178462034495},
  {'csa_number': 3,
   'evaluate_future': 0.13862669521770166,
   'evaluate_now': 0.16109044010962437},
  {'csa_number': 4,
   'evaluate_future': 0.673604568165596,
   'evaluate_now': 0.7617620506206674}],
```

اگر این اعداد را به درصد تبدیل کنیم داریم.

```
[{'csa_number': 1,
  'evaluate_future': 4.090616821318107,
  'evaluate_now': 4.672384330162824},
 {'csa_number': 2,
  'evaluate_future': 2.526855817273376,
  'evaluate_now': 2.9519486538771558},
 {'csa_number': 3,
  'evaluate_future': 0.8664168451106353,
  'evaluate_now': 1.0068152506851522},
 {'csa_number': 4,
  'evaluate_future': 4.210028551034975,
  'evaluate_now': 4.761012816379171}],
```

حال اگر زمان هایی را که اختلاف صفر است را در نظر نگیریم به اعداد زیر می رسیم.

```
[{'csa_number': 1,
  'evaluate_future': 2.3866545202151657,
  'evaluate_now': 2.7210702969135077},
 {'csa_number': 2,
  'evaluate_future': 2.373931265716681,
  'evaluate_now': 2.7319563595673255},
 {'csa_number': 3,
  'evaluate_future': 2.3550848827809214,
  'evaluate_now': 2.764925290536801},
 {'csa_number': 4,
  'evaluate_future': 2.3992881355932205,
  'evaluate_now': 2.7459379358437936}],
```

اگر آن ها را به درصد تبدیل کنیم داریم.

```
[{'csa_number': 1,
  'evaluate_future': 14.916590751344785,
  'evaluate_now': 17.006689355709423},
 {'csa_number': 2,
  'evaluate_future': 14.837070410729256,
  'evaluate_now': 17.074727247295783},
 {'csa_number': 3,
  'evaluate_future': 14.719280517380758,
  'evaluate_now': 17.280783065855008},
 {'csa_number': 4,
  'evaluate_future': 14.99555084745763,
  'evaluate_now': 17.16211209902371}])
```

اصلاحات

محاسبه اهمیت یک فرآیند

اگر فرآیند Process Data را در نظر بگیریم، این فرآیند در مسیری با ضریب اهمیت 5 قرار دارد و در این مسیر 4 فرآیند وجود دارد پس اهمیت این فرآیند برابر با (5/4=1.25) است.

این فرآیند در ماموریت Process Data Correctly نیز وجود دارد که ضریب اهمیت آن برابر 4 است و در آن 3 فرآیند وجود دارد پس اهمیت این فرآیند در بخش ماموریت ها برابر با (4/3=1.3) است.

ضریب اهمیت یک فرآیند از مجموع اهمیت آن فرآیند در بخش Workflow و بخش Mission بدست می آید که در اینجا برابر با (1.25+1.3=2.55) است.

در شکل زیر محاسبه برای همه فرآیند ها را در کد مشاهده می کنید.

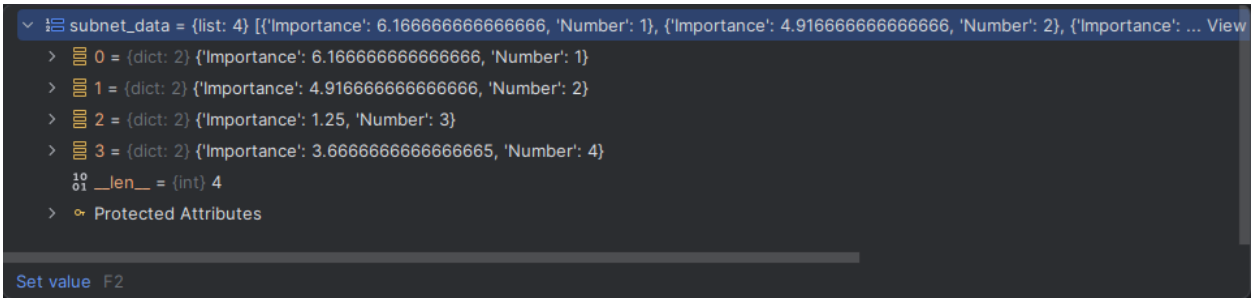
محاسبه اهمیت یک host

برای محاسبه میزان اهمیت یک host ابتدا باید میزان اهمیت subnet مربوطه را بدست آوریم سپس اهمیت بدست آمده را تقسیم بر تعداد host های درون subnet کنیم.

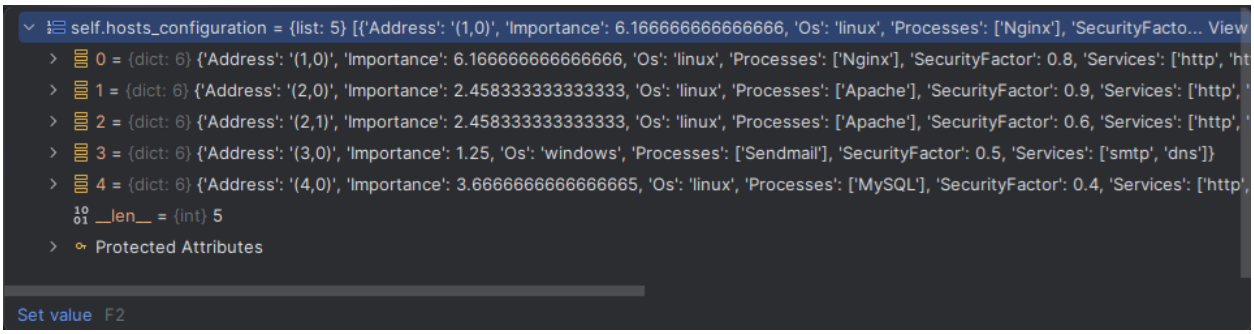
برای بدست آوردن اهمیت یک subnet از اهمیت فرآیند ها استفاده می کنیم بدین صورت که می دانیم هر فرآیند به کدام ResourcePool مربوط است و هر ResourcePool مربوط به کدام subnet است. اگر ResourcePool وابستگی به ResourcePool های دیگر نداشته باشد همان اهمیت فرآیند به اهمیت subnet اضافه می شود در غیر این صورت میزان اهمیت بین این

ResourcePool (و به طبع آن subnet مربوطه) و ResourcePool های وابسته تقسیم می شود.

در شکل زیر اعداد اهمیت هر subnet محاسبه شده است.



در شکل زیر اعداد اهمیت هر host محاسبه شده است.



بدست آوردن تعداد شبیه ساز حمله

به جای استفاده از عدد ثابت ده هزار تا می آییم و احتمال رخ دادن موفقیت آمیزترین مسیر حمله را به دست می آوریم. در این مدل مسیر حمله زیر است

["A", "C", "E", "G", "I", "K:S", "None:S"]

اگر احتمال موفقیت ها را در هم ضرب کنیم، احتمال رخ دادن آن مسیر حمله به دست می آید.

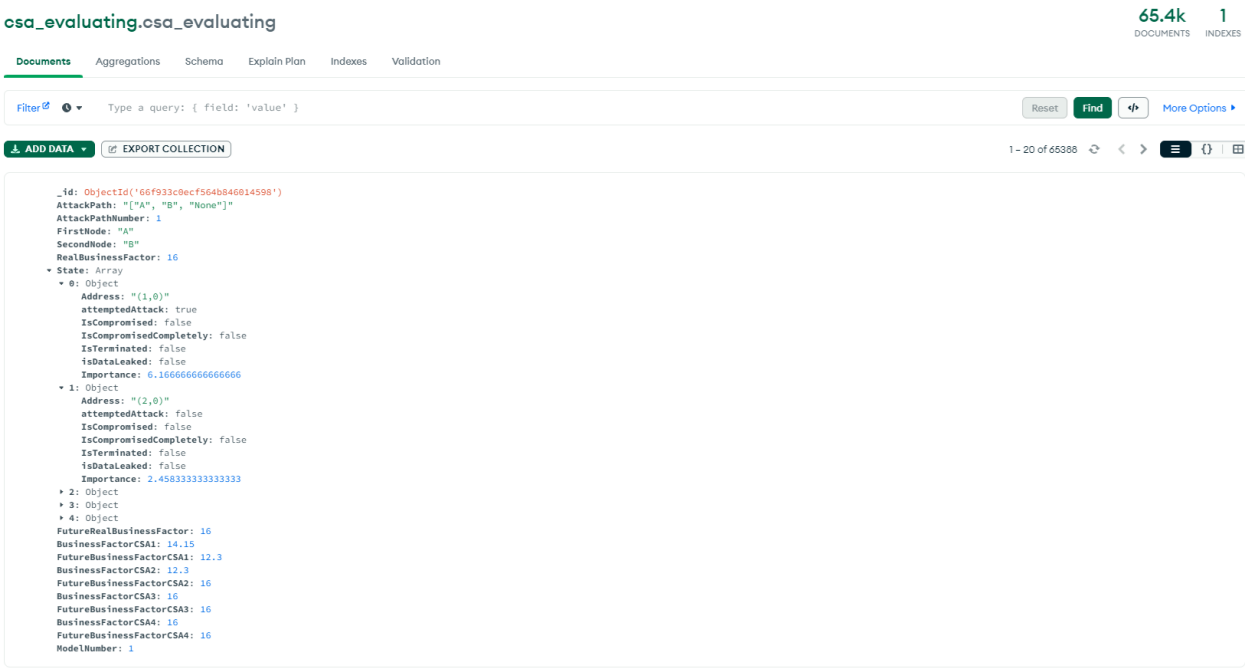
(0.8 * 0.2) * (0.6 * 0.2) * (0.7 * 0.1) * (0.7 * 0.6) * (0.7 * 0.6) * (0.8 * 0.5) = 0.00009483264

اگر عدد بدست آمده را ضربدر x کنیم و مقدار برابر با 1 شود آنگاه تعداد لازم را به دست می آوریم.

عدد بدست آمده را با صدگان آن رند می کنیم. یعنی 10,544.89256019868 برابر با 10500 می شود.

دیتاست

برای آن که وضعیت شبکه را در هر داده داشته باشیم فیلدی به نام state را اضافه کردیم که در آن وضعیت هر کدام از host ها آورده شده است.



در هنگام تمام شدن باید فیلد های مربوط به CSA را حذف کنیم چرا که آن ها برای مقایسه استفاده شده اند و نیازی به حضور در دیتاست ندارند.