1. Проблема сетевых атак
   1. Уязвимости
   2. Виды уязвимостей, приводящих к проникновению на узел
   3. Методы сканирования узлов
   4. Основные способы эксплуатации уязвимостей
2. Графы атак в задачах оценки защищенности сети
   1. Описание графа атак
   2. Способы построения графов атак
      1. Ручное построение графа
      2. Автоматизация построения графа (metasploit + nmap, Ш.1)
   3. Анализ графа атак (Ш. 2-3)
      1. Методики оценки защищенности сети
         1. Метрики защищенности узлов (CVSS, Научные работы(есть ссылки)
      2. Методики выбора защитных мер
         1. –
         2. –
         3. Недостаточность существующих методов
      3. Метод выбора контрмер на основе удаления узла, максимизирующего риски (СВОЙ МЕТОД)
      4. Оптимизация процесса выбора наиболее критического узла ( Ш.4)
3. Разработка системы автоматизированного тестирования на проникновение и итерированного выбора контрмер
   1. Архитектура системы
   2. Оценка сложности системы до оптимизации
   3. Оценка сложности системы после оптимизации
4. Тестирование разработанной системы
   1. Оценка влияния введенного метода выбора контрмер на безопасность системы

Ш.1.

В процессе автоматизированного тестирования на проникновение логируем все достижимые узлы, способ проникновения и подсчитанный уровень уязвимости узла, для чего используем различные метрики (cvss, есть научная работа по метрикам уязвимости узлов)

Ш.2.

После выполнения Ш.1. строим граф, состоящий из узлов-устройств (D), узлов-уязвимостей (V) и ребер E, где E – подмножество V x D (данном графе не будет существовать ребер типа D x D). (Возможно стоит рассмотреть ребра V x V -> переход от уязвимости к уязвимости).

Считаем угрозу, создаваемую каждым узлом:

*,* где – уязвимость устройств j, достижимых из i.

Ш.3.

Считаем уязвимость всей исследуемой системы:

Ш.4.

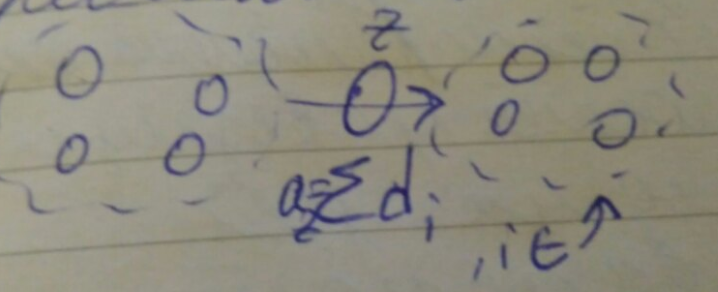
Анализируем существующий граф с целью нахождения такого узла-уязвимости (V), при удалении которого значение максимально уменьшится.

Решение задачи без оптимизаций:

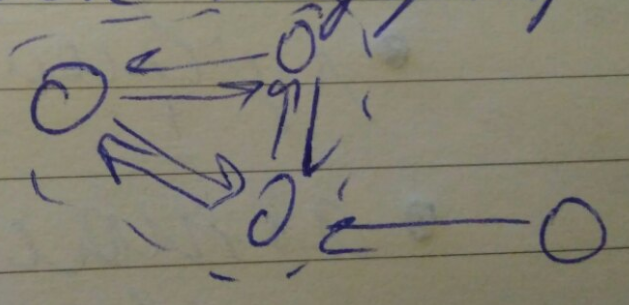
По очереди удаляем каждый из узлов-уязвимостей и пересчитываем угрозу попадания на каждый из узлов и уязвимость всей системы Ищем такой узел, при котором изменение уязвимости системы максимально.

Оптимизации:

* Для мостов (единственный путь между двумя подграфами) нет необходимости пересчитывать пути.

**

* Значения уязвимости всех «листов» графа можно проссумировать с родителями до тех пор, пока не натыкаемся на узел с несколькими дочерними объектами (считаем рекурсивно, проблема возникает только при подходе к полным графам (там куча циклов, по идее не сможем просчитать однозначно угрозу маршрута через них);
* Находим полные подграфы и считаем для них , так как при попадании в любой из узлов такого подграфа мы гарантированно попадаем на все остальные (или решаем задачу нахождения максимально длинного цикла!, попав на экземпляр цикла гарантированно достигаем всех элементов)

**