Защищенные мультисервисные телекоммуникационные системы

№ 1 Модель Харрисона-Руззо-Ульмана

Задача 1. Построить матрицы и записать в виде команд сценарий атаки с помощью троянской программы в системах, функционирующих на основе модели Харрисона-Руззо-Ульмана (ХРУ)

> Дано: Пусть имеется два субъекта: s1 - доверенный пользователь, admin и s2 - рядовой пользователь, user; и два каталога o1 и o2, владельцами которых являются пользователи s1 и s2. В каталоге o1 имеется объект o3 с секретной информацией. Исходная матрица доступа имеет вид:

S	o1	o2	o3
s1	own, r, w, e	r, w, e	own, r, w, e
s2		own, r, w, e	

Решение:

Атакующий s2 должен создать в своем каталоге o2 файл с трояном и дать права rwe пользователю s1 на этот файл. Далее ожидает запуска доверенным пользователем s1 (или исп. команду sleep()) трояна из каталога o2. Главная цель убедить пользователя s1 запустить файл. Когда пользователь s1 запускает файл, троян скопирует секрет из o3 в o2, затем делегирует права доступа для атакующего s2, который теперь может работать с тем же уровнем доступа, что и s1. В итоге файл становится доступным для чтения s2, и атакующий может его прочитать из каталога o2.

```
command "создать файл" (s2, троян):
  if "write" принадлежит [s2, o2] then
    Создать объект троян;
    Ввести {"own", "read", "write", "execute"}
    в [s2, троян];
    end if
  if {"read", "write"} подмножество [s1, o2] then
    Ввести {"read", "write", "execute"} в [s1, троян];
    end if
  end command
```

S	o1 (секрет)	o2(секрета нет)	o3(секрет)	o4 (троян)
s1	own, r, w, e	r, w, e	own, r, w, e	r, w, e
s2		own, r, w, e		own, r, w, e

```
command "запустить файл" (s1,троян):
  if {"read","write","execute"} подмножество [s1,троян] then
    Создать субъект s(троян);
    Ввести { "read","write","execute"} в [s(троян),o2];
    Ввести { "read","write","execute"} в [s(троян),троян];
    end if
  if {"own", "read","write","execute"} подмножество [s1,o1] and {"own",
   ↪  "read","write","execute"} подмножество [s1,o3] then
    Ввести {"read","write","execute"} в [s(троян),o1];
    Ввести {"read","write","execute"} в [s(троян),o3];
    end if
  end command
```

S	o1 (секрет)	o2(секрета нет)	o3(секрет)	o4 (троян)
s1	own, r, w, e	r, w, e	own, r, w, e	r, w, e
s2		own, r, w, e		own, r, w, e
s(троян)	r, w, e	r, w, e	r, w, e	r, w, e

```
command "скопировать файл o3 программой s(троян) в o2" (s(троян),o3,o2):
  if "read" принадлежит [stp,o3] and "write" принадлежит [s(троян),o2] then
    Создать объект o';
    Ввести {"own", "read", "write", "execute"} в [s(троян),o'];
    Ввести "read" в [s2,o'];
    Читать (s(троян),o3);
    Записать (s(троян), o');
    end if
    Уничтожить субъект s(троян);
  end command
```

S	o1 (секрет)	o2(секрета нет)	o3(секрет)	o4 (троян)	o'= o3(секрет)
s1	own, r, w, e	r, w, e	own, r, w, e	r, w, e	
s2		own, r, w, e		own, r, w, e	r

Задача 2. Построить сценарий аналогичной атаки. Отобразить последовательности команд перехода и изменений матрицы доступа.

> Дано: Доверенный пользователь s1 в исходном состоянии имеет на каталог o2 только права чтения r. Исходная матрица доступа имеет вид:

S	o1 (секрет)	o2(секрета нет)	o3(секрет)
s1	own, r, w, e	r	own, r, w, e
s2		own, r, w, e	

Решение:

s2, являясь владельцем o2, дает на него недостающие права s1

```
command "дать права на каталог от владельца" (s1,s2,o2):
  if "own" принадлежит [s2,o2] then
    Ввести {"write","execute"} в [s1,o2];
    end if
  end command
```

S	o1 (секрет)	o2(секрета нет)	o3(секрет)
s1	own, r, w, e	r, w, e	own, r, w, e
s2		own, r, w, e	

№ 3 Модель с типизованной матрицей доступа

Задача 1. Построить в соответствии с командой A граф отношений наследственности.

> Дано: Система настроена в соответствии с типизованной моделью доступа. Пусть имеется три типа сущностей: a, b и c. Начальное состояние системы S0: (s1: A) - субъект s1 типа a. Система переходит в новое состояние S0 aS1 при которой создается объект o1 типа c, инициализируются новые субъекты s2 типа b и s3 типа A. Вновь созданным субъектам предоставляются права доступа r’ и r’’ на объект o1. Переход осуществляется по следующей команде:

```
A(s1: A; s2: b; o1: ⌧)
  Create object o1 of type c;
  Inter r into [s1, o1];
  Create object s2 of type b;
  Inter r' into [s2, o1];
  Create object s3 of type a;
  Inter r'' into [s3, o1];
end A
```

Решение:

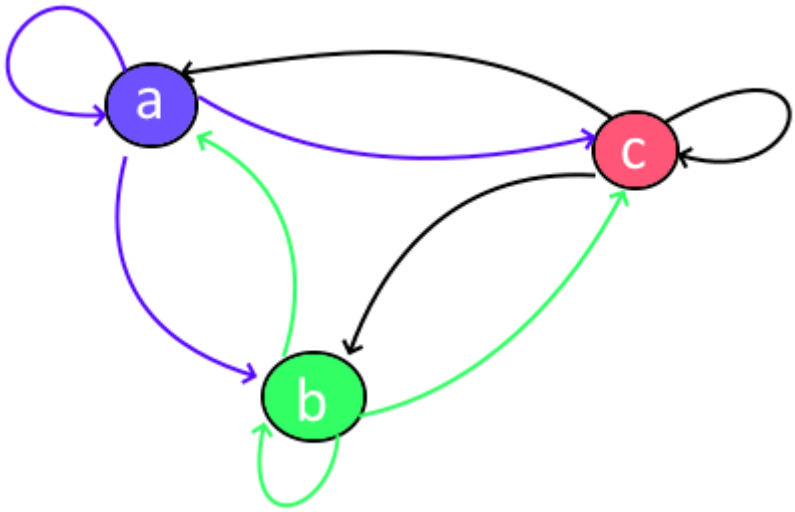


Figure 1: Граф отношений наследственности a, b, c

Задача 2. Реализовать сценарий атаки с помощью троянской программы при условии функционирования по типизованной модели доступа

> Дано: Пусть имеется два субъекта доступа: (s1: a) – субъект s1 типа a - доверенный пользователь admin (s2: b) - субъект s2 типа b – обычный пользователь user. Три объекта доступа: - Каталог (o1: v)sec – принадлежит пользователю (s1:a) “own”⊠ rs1,o1; - Несекретный каталог (o2: ⊠)non sec – принадлежит пользователю (s2:b) “own”⊠ r′s2,o2; - Секретный файл (o3: v)sec в каталоге (o1: v)sec – принадлежит пользователю (s1:a) “own”⊠ r′′s1,o3;
>
> В исходном состоянии граф наследственности имеет четыре вершины: a, b, v, n.
>
> Построить граф отношений наследственности по сценарию атаки троянским конем со стороны пользователя s2 на секретный файл o3 и записать команды перехода из состояния в состояние в нотации модели ТМД.

Решение:

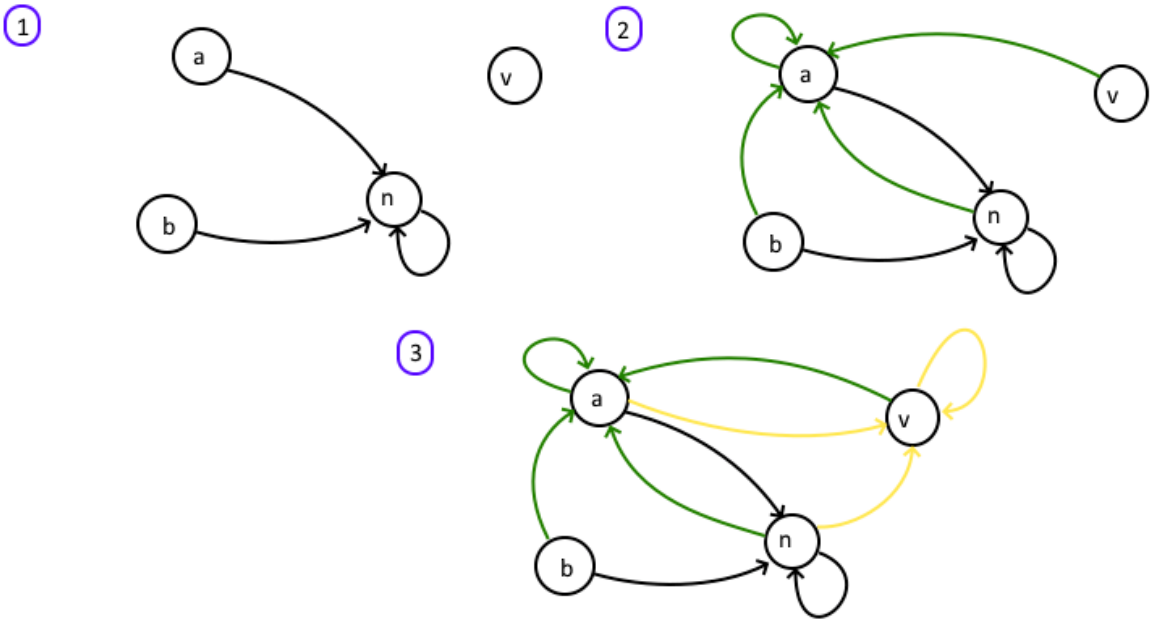
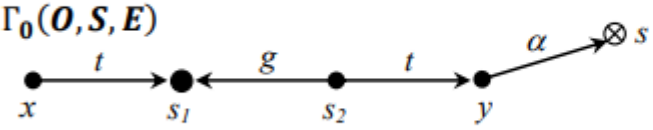


Figure 2: Граф отношений наследственности a, b, v, n

№ 4 Модель Take-Grant

Задача 1. Построить систему команд перехода передачи субъекту x прав доступа A на объект s от субъекта y

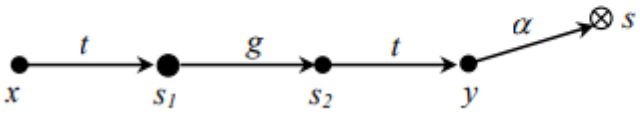
> Система субъектов и объектов представлена графом $\Gamma_0(\boldsymbol{O},\boldsymbol{S},\boldsymbol{E})$ вида
>
> $\Gamma_0(\boldsymbol{O},\boldsymbol{S},\boldsymbol{E})$
>
> 
>
>
> Дано: Сущности $\boldsymbol{x}$ и $\boldsymbol{y}$ связаны tg-путем.

Решение:

Объект S2 наследует у объекта Y право доступа к объекту S (у S2 есть на это права — t (take)). Затем объект S2 предоставляет объекту s1 свое право на объект S (у объекта S2 есть права grant для делегирования прав). Далее объект X берет право a на объект S у объекта S1, так как S1 имеет доступ к S, а X имеет право t (take) прав у объекта S1.

Итоговая цепочка атаки: TAKE(S2 Y => S) GRANT(S2 S1 => S) TAKE(X S1 => S).

Задача 2. Построить систему команд перехода передачи субъекту x прав доступа A на объект s от субъекта y и оценить возможность передачи прав доступа по tg-пути независимо от направления прав t и g

> Система субъектов и объектов представлена графом $\Gamma_0(\boldsymbol{O},\boldsymbol{S},\boldsymbol{E})$ вида
>
> $\Gamma_0(\boldsymbol{O},\boldsymbol{S},\boldsymbol{E})$
>
> 
>
>
> Дано:

Решение:

Аналогично задаче 1, объект S2 берет право у Y на доступ к объекту S. Затем S1 создает новый объект ABOBA со следующими правами: TAKE + GRANT, и дает права GRANT для объекта S2,

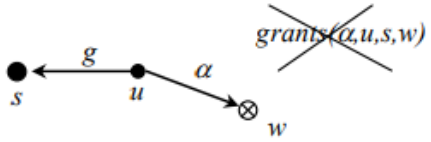
чтобы тот мог работать с объектом ABOBA. Объект S2, имея права на объект S и права GRANT для ABOBA, предоставляет объекту ABOBA права на объект S. S1, будучи владельцем ABOBA, наследует права на S у объекта ABOBA.

Итог: S1 получил доступ к S независимо от направления прав.

Задача 3. Построить систему команд получения субъектом s прав доступа A на объект w от субъекта u при условии того, что команда grants(A,u,s,w) не может быть задействована

Дано:

Система субъектов и объектов представлена графом $\Gamma_0(\boldsymbol{O},\boldsymbol{S},\boldsymbol{E})$ вида



Политика безопасности системы запрещает любым субъектам предоставлять право **α** на «свои» объекты другим субъектам, но не запрещает субъектам, которые владеют правами на какие-либо субъекты брать у них права на их объекты.

Кроме субъекта s, субъект u может быть связан ***tg***-путем с другими субъектами.

Решение:

Пусть объект NEW имеет права на объект U, и объект U также имеет права на объект NEW. Тогда:

1. Объект U предоставляет права на S объекту NEW.
2. Объект S берет у объекта NEW право на объект U.
3. Объект S берет у U права на W.

Итог: Объект S имеет права на W.

№ 5 Модель ь Белла-Ла Падуллы

Задача 1. Составить систему уровней допусков пользователей, грифов секретности объектом доступа и матрицу доступа

Дано:

Пусть имеется мандатная модель системы доступа $\Gamma(\boldsymbol{\nu_0}, \boldsymbol{Q}, \boldsymbol{F_r})$. Решетка уровней безопасности $\boldsymbol{\Lambda_L}$ линейна и имеет три уровня l_1, l_2, l_3; причем $l_1 > l_2 > l_3$; $l_1 > l_3$.

Субъекты системы:

u_1 - администратор системы (admin)
u_2 – директор компании
u_3 – делопроизводитель
u_4 - пользователь (user)

Объекты доступа:

o1 - системное ПО;
o2 – документ верхнего уровня («Стратегия компании»)
o3 - приказ о формировании рабочей группы
o4 - информационная система предприятия (СДО)

Решение:

NRU (Нельзя читать вверх) — Пользователь может читать только свой каталог и более низкие по правам доступа. Все, что выше прав пользователя, недоступно.

NWD (Нельзя писать вниз) — Пользователь может писать только в свой уровень доступа и выше. Если он попытается записать данные в файлы более низкого уровня доступа, запись будет запрещена.

S	o1 (l3)	o2 (l1)	o3 (l2)	o4 (l3)
u1 (l2)	r	w	rw	r
u2 (l1)	r	rw	r	r
u3 (l2)	r	w	rw	r
u4 (l3)	rw	w	w	rw

Задача 2.

> Дано:

Решение:

Задача 3.

> Дано:

Решение: