О проблеме обоснования адекватности формальных моделей безопасности логического управления доступом и их реализации в компьютерных системах

д.т.н., доцент Девянин П.Н.
УМО ИБ, г. Москва
peter_devyanin@hotmail.com

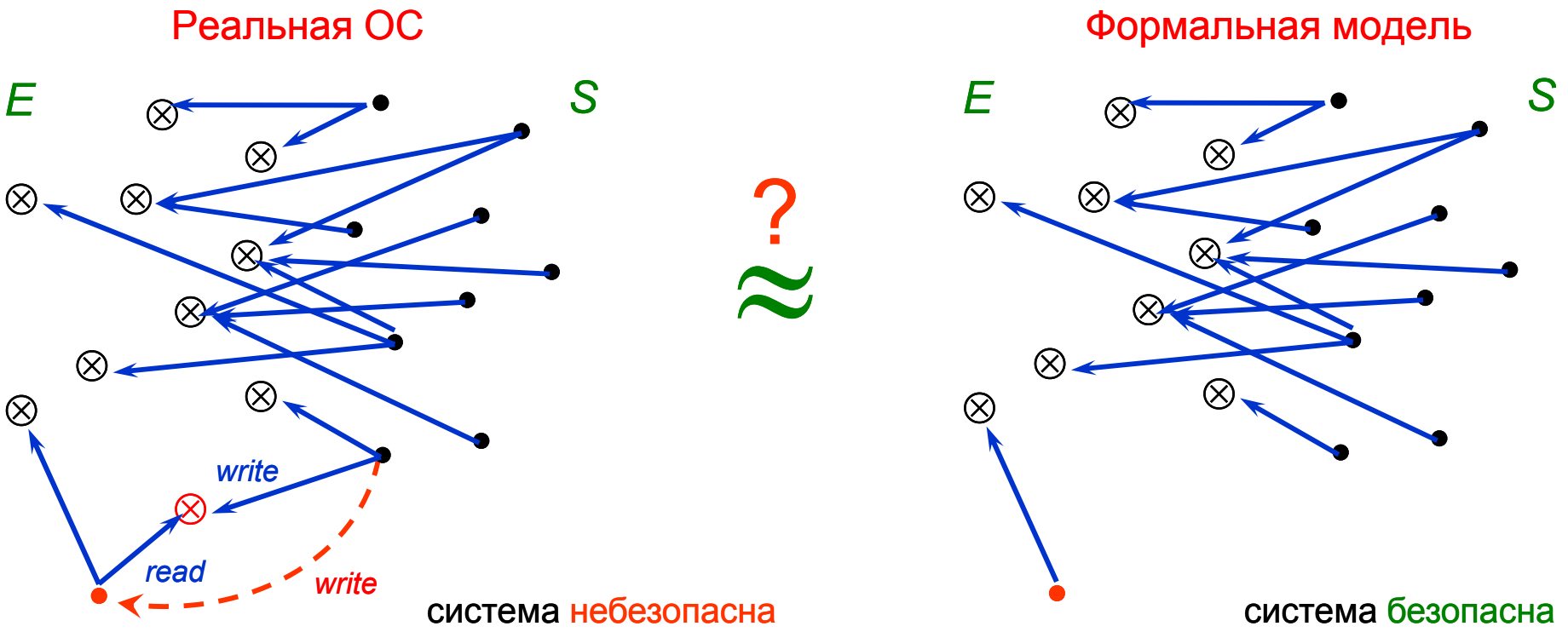
Модели безопасности управления доступом и их применение

- Классическая модель Белла-ЛаПадулы (мандатное управление доступом в ОС) и ее интерпретации, модель мандатной политики целостности информации Биба (механизм *MIC* в ОС *Windows Vista*/7/2008), модель систем военных сообщений (ЭПС);
- Модель *Take-Grant* и ее основные расширения;
- Модель Харрисона-Руззо-Ульмана и ее развитие — модель типизированной матрицы доступов (пакет *SELinux*);
- Субъектно-ориентированная модель ИПС (ИПС в защищенных ОС);
- Базовая модель ролевого управления доступом *RBAC* (управление доступом в СУБД) и ее расширение для КС с мандатным управлением доступом;
- Группоцентрические модели управления доступом (*g-SIS*), модели делегирования доверия;
- Модели безопасности логического управления доступом и информационными потоками — ДП-модели (анализ безопасности защищенных ОС).

Постановка проблемы

Применение классических методов теории моделирования (статистических, экспериментальных) для анализа адекватности моделей безопасности логического управления доступом и информационными потоками реальным КС затруднено.

Пример:

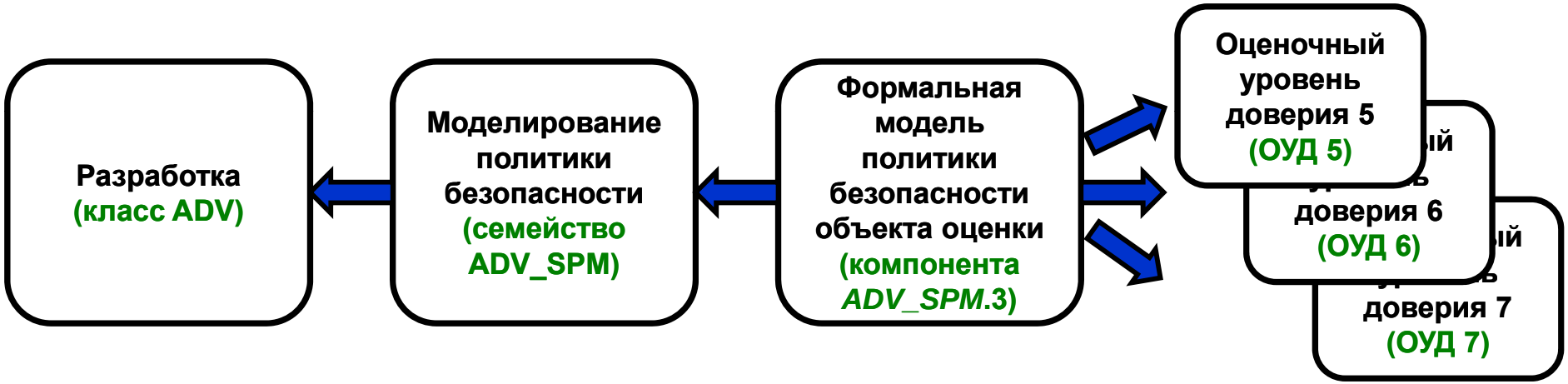


система небезопасна

система безопасна

Требования критериев оценки защищенности КС

«Критерии оценки безопасности информационных технологий» ГОСТ Р ИСО/МЭК 15408-2002:



«Как минимум, требуется моделировать политики управления доступом и информационными потоками (если они являются частью ПБО), так как в настоящее время это признается возможным».

Поэтапное решение

1. В соответствии с заданными для защищенной КС целями безопасности (например, на основе ГОСТ Р ИСО/МЭК 15408-2002) выбор направления формального моделирования безопасности логического управления доступом и информационными потоками (например, мандатное управление доступом).
2. Построение формальной модели. В соответствии с целями безопасности формулирование и обоснование условий безопасности КС.
3. Получение из формальной модели спецификаций (предусловия и постусловия) функций, реализующих в КС механизм управления доступом. Обоснование корректности реализации функций непосредственно в программном коде. Верификация программного кода — устранение «обходных путей» предоставления доступов.

2 этап — построение формальной модели. Пример — РОСА ДП-модель

$E = O \cup C$ — множество сущностей (O — множество объектов, C — множество контейнеров);

U — множество учетных записей пользователей; (L_U , N_U — учетные записи доверенных/недоверенных пользователей);

$]u[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с $u \in U$, и $UE = \{e \in]u[: u \in U\}$;

$S \subseteq E$ — субъект-сессии учетных записей пользователей (L_S , N_S — доверенные/недоверенные субъект-сессии);

R, AR — множество ролей/административных ролей ($AR \cap R = \varnothing$);

$]r[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с $r \in R \cup AR$, и $RE = \{e \in]r[: r \in R \cup AR\}$;

$R_r = \{read_r, write_r, execute_r, own_r\}$ — множество видов прав доступа; $R_a = \{read_a, write_a, own_a\}$ — множество видов доступа;
$R_f = \{write_m, write_t\}$ — множество видов информационных потоков;

$P \subset E \times R_r$ — множество прав доступа к сущностям; $A \subseteq S \times E \times R_a$ — множество доступов субъект-сессий к сущностям;

$F \subseteq E \times E \times R_f$ — множество информационных потоков;

$UA: U \to 2^R$, $AUA: U \to 2^{AR}$ — функции авторизованных ролей/административных ролей учетных записей пользователей;

$PA: R \to 2^P \setminus \{\varnothing\}$ — функция прав доступа ролей; $user: S \to U$ — функция принадлежности субъект-сессии учетной записи пользователя; $roles: S \to 2^R \cup 2^{AR}$ — функция текущих ролей субъект-сессий;

$can\_manage\_rights: AR \to 2^R$ — функция администрирования прав доступа ролей;

$[s] \subset E \cup U$ — сущности, функционально ассоциированных с субъект-сессией s, $fa: U \times E \to 2^E \cup 2^U$ — функция, задающая множества сущностей, функционально ассоциированных с субъект-сессией при ее создании;

$]s[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с субъект-сессией, $fp: U \times E \to 2^E$ — функция, задающая множества сущностей, параметрически ассоциированных с субъект-сессией при ее создании;

$H_R: R \to 2^R$ — функция иерархии ролей; $H_{AR}: AR \to 2^{AR}$ — функция иерархии административных ролей.

Иерархия сущностей

H_E: $E \to 2^E$ — функцию иерархии сущностей (сопоставляющую каждой сущности $e \in E$ множество сущностей $H_E(e) \subset E$, непосредственно в ней содержащихся), удовлетворяющую условиям:

Условие 1. Если сущность $e \in H_E(c)$, то $e < c$, при этом, если $e \in C \cup S$, то не существует сущности-контейнера $d \in C$ такой, что $e < d$, $d < c$.

Условие 2. Для любых сущностей $e_1, e_2 \in E$, $e_1 \neq e_2$, по определению выполняются равенство $H_E(e_1) \cap H_E(e_2) \cap (C \cup S) = \varnothing$ и условия:

- если $o \in O$, то справедливо равенство $H_E(o) = \varnothing$;
- если $e_1 < e_2$, то или $e_1, e_2 \in E \setminus S$, или $e_1, e_2 \in S$;
- если $e \in E \setminus S$, то $H_E(e) \subset E \setminus S$;
- если $s \in S$, то $H_E(s) \subset S$.

execute_container: $S \times E \to \{true, false\}$ — функция доступа субъект-сессии к контейнеру такая, что по определению для субъект-сессии $s \in S$ и сущности $e \in E$ справедливо равенство *execute_container*$(s, e) = true$ тогда и только тогда, когда либо $e \in S$, либо $e \in E \setminus S$ и существует последовательность сущностей $e_1, \ldots, e_n$, где $n \geq 1$, $e = e_n$, удовлетворяющих следующим условиям:

- не существует сущности-контейнера $e_0 \in E \setminus S$ такой, что $e_1 \in H_E(e_0)$;
- $e_i \in H_E(e_{i-1})$, где $1 < i \leq n$;
- $(e_i, execute_r) \in PA(roles(s))$, где $1 \leq i < n$.

shared_container: $C \to \{true, false\}$ — функцию разделяемых контейнеров такую, что сущность-контейнер $c \in C \setminus S$ является разделяемой, когда *shared_container*$(c) = true$, в противном случае *shared_container*$(c) = false$.

Мандатный контроль целостности

$(LI, \leq)$ — линейная шкала двух уровней целостности данных, где $LI = \{i\_low, i\_high\}$, $i\_low < i\_high$;

$(i_u, i_e, i_r, i_s) \in I$ — четверка функций уровней целостности, при этом:

$i_u: U \to LI$ — функция уровней целостности субъект-сессий;

$i_e: E \setminus S \to LI$ — функция уровней целостности сущностей;

$i_r: R \cup AR \to LI$ — функция уровней целостности ролей;

$i_s: S \to LI$ — функция текущих уровней целостности субъект-сессий;

I — множества всех четверок функций заданного вида;

Предположение.

- для ролей $r, r' \in R \cup AR$, если $r \leq r'$, то $i_r(r) \leq i_r(r')$;
- для сущностей $e, e' \in E \setminus S$, если $e \leq e'$, то $i_e(e) \leq i_e(e')$;
- для субъект-сессий $s, s' \in S$, если $s \leq s'$, то $i_s(s) \leq i_s(s')$;
- для каждой сущности $e \in u$, где $u \in U$, справедливо равенство $i_e(e) = i_u(u)$;
- для субъект-сессии $s \in S$ верно неравенство $i_s(s) \leq i_u(user(s))$;
- для учетной записи пользователя $u \in U$ и роли $r \in R$, если $r \in UA(u)$, то $i_r(r) \leq i_u(u)$;
- для субъект-сессии $s \in S$ и роли $r \in R$, если $r \in roles(s)$, то $i_r(r) \leq i_s(s)$;
- для права доступа к сущности $(e, \alpha) \in P$, где $\alpha \in \{own_r, write_r\}$, и роли $r \in R$, если $(e, \alpha) \in PA(r)$, то $i_e(e) \leq i_r(r)$;
- для учетной записи доверенного пользователя $u \in L_U$ справедливо равенство $i_u(u) = i\_high$;
- для учетной записи недоверенного пользователя $u \in N_U$ справедливо равенство $i_u(u) = i\_low$;
- верно равенство $i_e(i\_entity) = i\_high$.

Фактическое владение. Фактические роли, права доступа, доступы

de_facto_own: $S \to S$ –– функция фактического владения субъект-сессий субъект-сессиями такая, что для $s, s' \in S$.

Предположение. Если субъект-сессия *s* реализовала информационный поток по памяти от себя к сущности, функционально ассоциированной с другой субъект-сессией *s'*, или субъект-сессия *s* реализовала информационный поток по памяти к себе от всех сущностей, параметрически ассоциированных с другой субъект-сессией *s'*, то субъект-сессия *s* получает фактическое владение субъект-сессией *s'* ($s' \in de\_facto\_own(s)$).

Предположение. Если субъект-сессия *s* имеет доступ владения own_a к субъект-сессии *s'*, то субъект-сессия *s* получает возможности:

- использовать роли из множества текущих ролей субъект-сессии *s'*;
- изменять множество текущих ролей субъект-сессии *s'*;
- использовать текущий уровень целостности субъект-сессии *s'*;
- использовать доступы субъект-сессии *s'* или удалить субъект-сессию *s'*;
- фактически владеть субъект-сессиями, которыми фактически владеет субъект-сессия *s'*;
- использовать административные роли субъект-сессии *s'*;
- использовать информационные потоки, которые реализует субъект-сессия *s'*.

Используем обозначения:

- *de_facto_roles*: $S \to 2^{R \cup AR}$ –– фактические текущие роли субъект-сессий;
- *de_facto_rights*: $S \to 2^{P}$ –– фактические текущие права доступа субъект-сессий;
- *de_facto_accesses*: $S \to 2^{A}$ –– фактические доступы субъект-сессий.

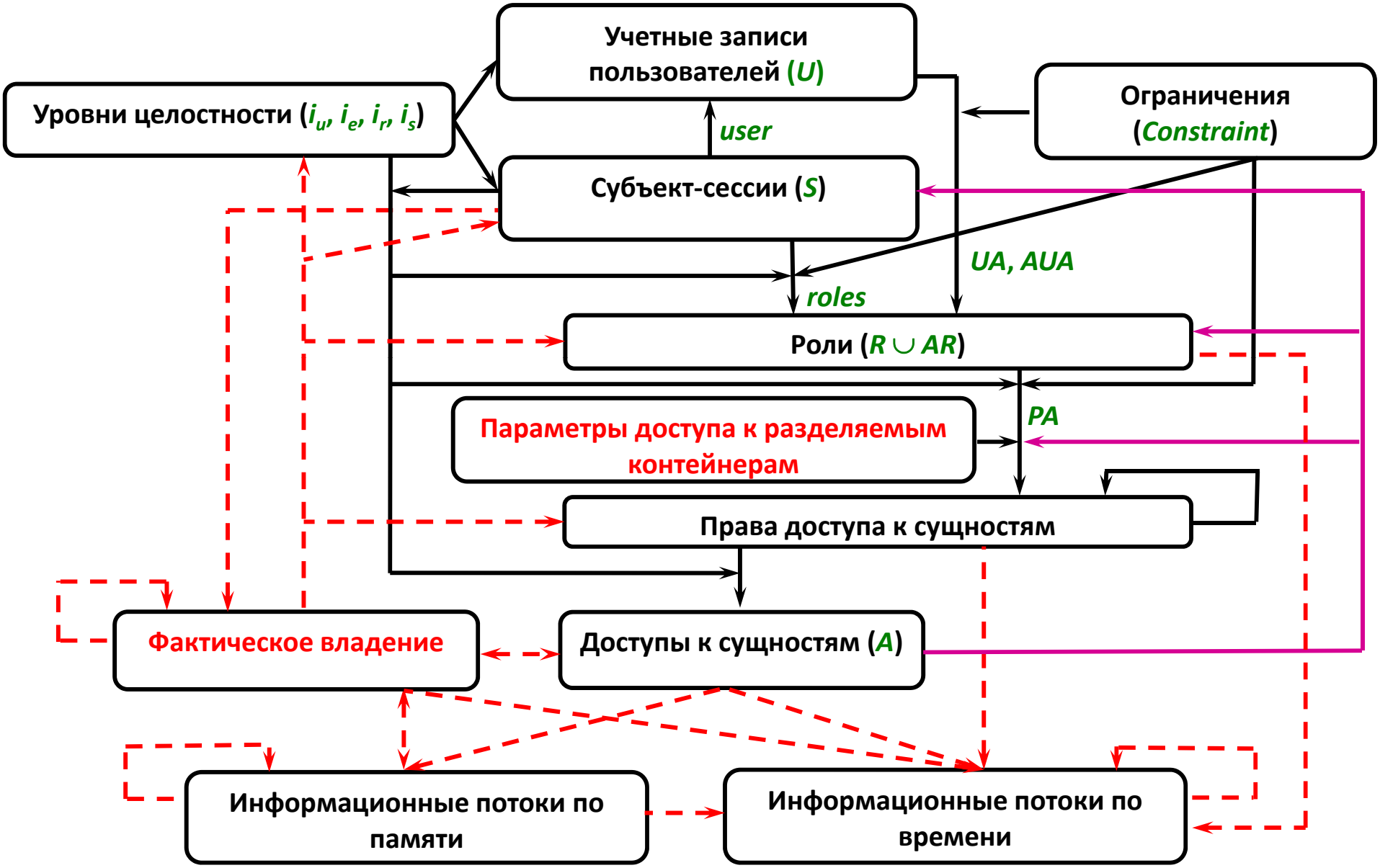
Примеры де-юре правил без информационных потоков по времени

take_roles$(x, x', \{r_j: 1 \le j \le k\})$	$x, x' \in S$, $r_j \in UA(user(x)) \cup AUA(user(x))$, $\{(e, x, read_a): e \in]r_j[$, где $1 \le j \le k\} \subset A$, $i_r(r_j) \le i_s(x)$, $Constraint_S(roles') = true$, [если $i_r(r_j) = i\_high$, то $(x', i\_entity, write_a) \in A$, где $1 \le j \le k$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H_E' = H_E$, $roles'(x) = roles(x) \cup \{r_j: 1 \le j \le k\}$ и для $s \in S \setminus \{x\}$ выполняется равенство $roles'(s) = roles(s)$, $F' = F$
grant_rights$(x, x', r, \{(y_j, \alpha_{rj}): 1 \le j \le k\})$	$x, x' \in S$, $y_j \in E$, $r \in can\_manage\_rights(roles(x) \cap AR)$, $(x, y_j, own_a) \in A$, $i_r(r) \le i_s(x)$, [если $y_j \in S$, то $\alpha_{rj} = own_r$ и $i_s(y_j) \le i_r(r)$], [если $y_j \in E \setminus S$ и $\alpha_{rj} \in \{own_r, write_r\}$, то $i_e(y_j) \le i_r(r)$], $[Constraint_P(PA') = true]$, [если $i_e(y_j) = i\_high$, то $(x', i\_entity, write_a) \in A$, где $1 \le j \le k$]	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H_E' = H_E$, $PA'(r) = PA(r) \cup \{(y_j, \alpha_{rj}): 1 \le j \le k\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, $F' = F$
create_hard_link(x, x', y, z)	$x, x' \in S$, $y \in O \setminus S$, $z \in C \setminus S$, $y \notin UE \cup RE$, $(x, z, write_a) \in A$, $i_e(y) \le i_r(r) \le i_s(x)$, $i_e(y) \le i_e(z) \le i_s(x)$, [если $i_e(z) = i\_high$, то $(x', i\_entity, write_a) \in A$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H_E'(z) = H_E(z) \cup \{y\}$, для $e \in E \setminus \{z\}$ выполняется равенство $H_E'(e) = H_E(e)$, $F' = F$
create_first_session(x, x', u, r, y, z, zi)	$x, x' \in S$, $u \in U$, $y \in E$, $z \notin E$, $(y, execute_r) \in PA(roles(x))$, $execute\_container(x, y) = true$, $r \in can\_manage\_rights(roles(x) \cap AR)$, $zi \le i_u(u)$, $zi \le i_r(r)$, $\{(e, x, read_a): e \in]u[\} \subset A$, $Constraint_P(PA') = true$, $Constraint_S(roles') = true$, [если $zi = i\_high$, то $(x', i\_entity, write_a) \in A$]	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A \cup \{(x, z, own_a)\}$, $i_s'(z) = zi$, $user'(z) = u$, для $s \in S$ выполняется равенство $user'(s) = user(s)$, $roles'(z) = \varnothing$, для $s \in S$ выполняется равенство $roles'(s) = roles(s)$, $[z] = fa(u, y)$, $]z[= fp(u, y)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, $H_E'(z) = \varnothing$, для $e \in E$ выполняется равенство $H_E'(e) = H_E(e)$, $F' = F$
access_write(x, x', y)	$x, x' \in S$, $y \in E \setminus S$, $(y, write_r) \in PA(roles(x))$, $execute\_container(x, y) = true$, $i_e(y) \le i_s(x)$, [если $i_e(y) = i\_high$, то $(x', i\_entity, write_a) \in A$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H_E' = H_E$, $A' = A \cup \{(x, y, write_a)\}$, $F' = F \cup \{(x, y, write_m)\}$

Де-факто правила без информационных потоков по времени

$de\_facto\_op(x, op(y, y', ...))$	$x, y, y' \in S, y, y' \in de\_facto\_own(x)$, выполняются условия применения де-юре правила преобразования состояний $op(y, y', ...)$	Соответствуют результатам применения правила $op(y, y', ...)$
$control(x, y, z)$	$x, y \in S, x \neq y, z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$, или $z \in S$ и $z \in de\_facto\_own(x)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E, A' = A, F' = F, de\_facto\_own'(x) = de\_facto\_own(x) \cup \{y\}$
$know(x, y)$	$x, y \in S, x \neq y$, и для каждой $e \in]y[$, существует $(e, x, write_m) \in F$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E, A' = A, F' = F, de\_facto\_own'(x) = de\_facto\_own(x) \cup \{y\}$
$take\_access\_own(x, y, z)$	$x, y, z \in S, y \in de\_facto\_own(x), z \in de\_facto\_own(y)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E, A' = A, F' = F, de\_facto\_own'(x) = de\_facto\_own(x) \cup \{z\}$
$flow\_memory\_access(x, y, \alpha_a)$	$x \in S, y \in E, (y, \alpha_a) \in de\_facto\_accesses(x)$, где $\alpha_a \in \{read_a, write_a\}$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E,$ если $\alpha_a = read_a$, то $F' = F \cup \{(y, x, write_m)\}$, если $\alpha_a = write_a$, то $F' = F \cup \{(x, y, write_m)\}$
$find(x, y, z)$	$x, y \in S, z \in E, x \neq z, (x, y, write_m) \in F$, [или $(z, write_a) \in de\_facto\_accesses(y)$, или $(y, z, write_m) \in F$]	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E, F' = F \cup \{(x, z, write_m)\}$
$post(x, y, z)$	$x, z \in S, y \in E, x \neq z, (y, read_a) \in de\_facto\_accesses(z)$, [или $(y, write_a) \in de\_facto\_accesses(x)$, или $(x, y, write_m) \in F$]	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E, F' = F \cup \{(x, z, write_m)\}$
$pass(x, y, z)$	$y \in S, x, z \in E, x \neq z,$ $(x, read_a) \in de\_facto\_accesses(y)$ [или $(z, write_a) \in de\_facto\_accesses(y)$, или $(y, z, write_m) \in F$]	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E, F' = F \cup \{(x, z, write_m)\}$
$take\_flow(x, y)$	$x, y \in S, x \neq y, y \in de\_facto\_own(x)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E, F' = F \cup \{(x, e, write_m): (y, e, write_m) \in F, e \in E\}$

Зависимость условий и результатов правил



3 этап — обоснование корректности программной реализации формальной модели

Правила вывода — $V\ \{S\}\ P$ (в том числе):

- усиления предусловия и ослабления постусловия;
- оператора присваивания;
- условного оператора *if*;
- последовательности операторов;
- цикла с условием продолжения;
- подпрограмма или сегмент программы.

Если

$V \Rightarrow V1$ и $\{V1\}\ OP\ \{P1\}$ и

$P\ 1 \Rightarrow P$

то

$\{V\}\ OP\ \{P\}$

Если

$\{V1\ and\ B\}\ OP1\ \{P\}$ и

$\{V2\ and\ not\ B\}\ OP2\ \{P\}$

то

$\{V1$ and $V2\}$

if B

then $OP1$

else $OP2$

endif $\{P\}$

Спасибо за внимание!