ASTRA LINUX®

# Обеспечение доверия к системному ПО на примере ОС Astra Linux

Девянин П.Н., научный руководитель ГК Astra Linux

# ОСНОВНЫЕ ФУНКЦИИ ВСТРОЕННЫХ СЗИ СЕРТИФИЦИРОВАННОЙ ПО 1 КЛАССУ ЗАЩИТЫ ОССН ASTRA LINUX SPECIAL EDITION



**РЕЖИМ «КИОСК»**
Белый список приложений

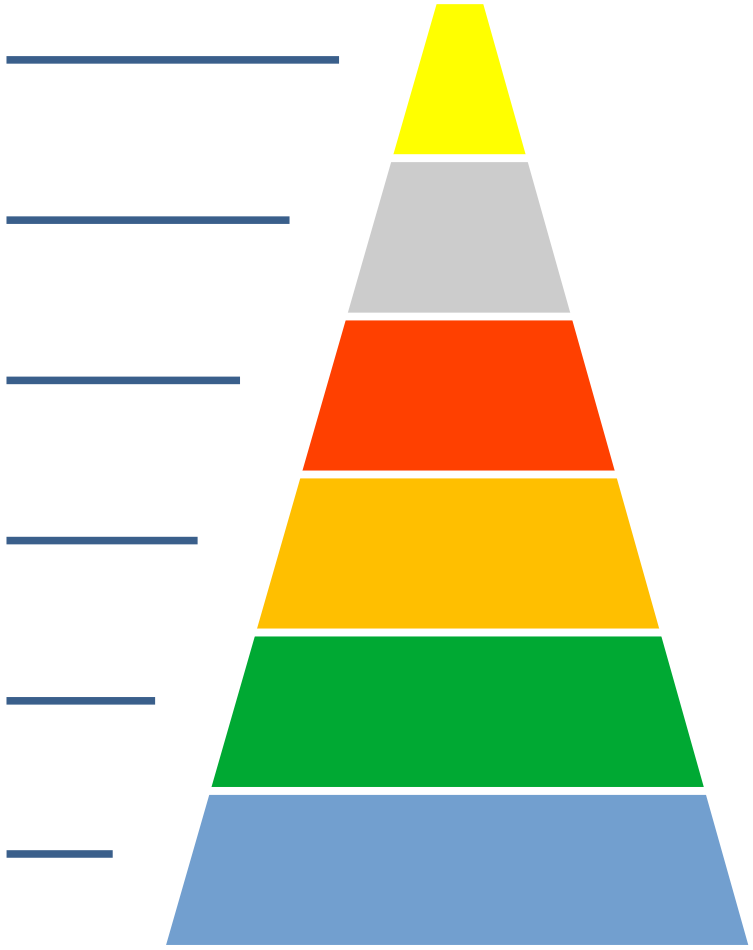
**ДИСКРЕЦИОННОЕ УПРАВЛЕНИЕ ДОСТУПОМ**
Защита информации одного уровня конфиденциальности

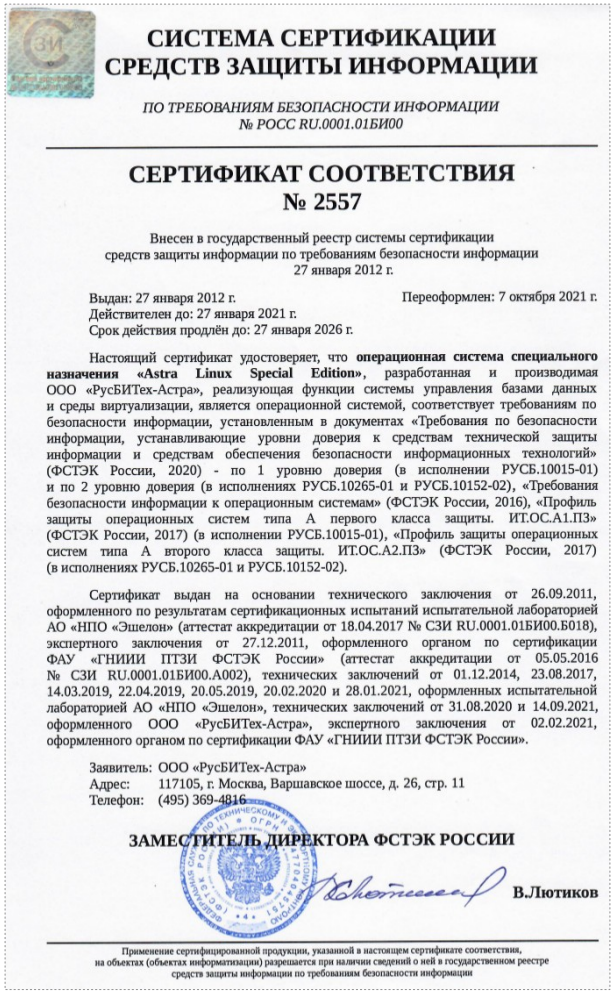
**МАНДАТНОЕ УПРАВЛЕНИЕ ДОСТУПОМ**
Защита информации различных уровней конфиденциальности

**ЗАМКНУТАЯ ПРОГРАММНАЯ СРЕДА**
Защита приложений от подмены

**МАНДАТНЫЙ КОНТРОЛЬ ЦЕЛОСТНОСТИ**
Защита целостности программной среды, в т.ч. от вирусов, закладок и типовых атак

**ОПЕРАЦИОННАЯ СИСТЕМА**
Настройка состава устанавливаемого ПО и расширенный аудит событий безопасности





СИСТЕМА СЕРТИФИКАЦИИ СРЕДСТВ ЗАЩИТЫ ИНФОРМАЦИИ

*ПО ТРЕБОВАНИЯМ БЕЗОПАСНОСТИ ИНФОРМАЦИИ*
*№ РОСС RU.0001.01БИ00*

СЕРТИФИКАТ СООТВЕТСТВИЯ
№ 2557

Внесен в государственный реестр системы сертификации средств защиты информации по требованиям безопасности информации 27 января 2012 г.

Выдан: 27 января 2012 г. Переоформлен: 7 октября 2021 г.
Действителен до: 27 января 2021 г.
Срок действия продлён до: 27 января 2026 г.

Настоящий сертификат удостоверяет, что **операционная система специального назначения «Astra Linux Special Edition»**, разработанная и производимая ООО «РусБИТех-Астра», реализующая функции системы управления базами данных и среды виртуализации, является операционной системой, соответствует требованиям по безопасности информации, установленным в документах «Требования по безопасности информации, устанавливающие уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий» (ФСТЭК России, 2020) - по 1 уровню доверия (в исполнении РУСБ.10015-01) и по 2 уровню доверия (в исполнениях РУСБ.10265-01 и РУСБ.10152-02), «Требования безопасности информации к операционным системам» (ФСТЭК России, 2016), «Профиль защиты операционных систем типа А первого класса защиты. ИТ.ОС.А1.ПЗ» (ФСТЭК России, 2017) (в исполнении РУСБ.10015-01), «Профиль защиты операционных систем типа А второго класса защиты. ИТ.ОС.А2.ПЗ» (ФСТЭК России, 2017) (в исполнениях РУСБ.10265-01 и РУСБ.10152-02).

Сертификат выдан на основании технического заключения от 26.09.2011, оформленного по результатам сертификационных испытаний испытательной лабораторией АО «НПО «Эшелон» (аттестат аккредитации от 18.04.2017 № СЗИ RU.0001.01БИ00.Б018), экспертного заключения от 27.12.2011, оформленного органом по сертификации ФАУ «ГНИИИ ПТЗИ ФСТЭК России» (аттестат аккредитации от 05.05.2016 № СЗИ RU.0001.01БИ00.А002), технических заключений от 01.12.2014, 23.08.2017, 14.03.2019, 22.04.2019, 20.05.2019, 20.02.2020 и 28.01.2021, оформленных испытательной лабораторией АО «НПО «Эшелон», технических заключений от 31.08.2020 и 14.09.2021, оформленного ООО «РусБИТех-Астра», экспертного заключения от 02.02.2021, оформленного органом по сертификации ФАУ «ГНИИИ ПТЗИ ФСТЭК России».

Заявитель: ООО «РусБИТех-Астра»
Адрес: 117105, г. Москва, Варшавское шоссе, д. 26, стр. 11
Телефон: (495) 369-4816

ЗАМЕСТИТЕЛЬ ДИРЕКТОРА ФСТЭК РОССИИ В.Лютиков

Применение сертифицированной продукции, указанной в настоящем сертификате соответствия, на объектах (объектах информатизации) разрешается при наличии сведений о ней в государственном реестре средств защиты информации по требованиям безопасности информации

# ТРЕБОВАНИЯ ПО БЕЗОПАСНОСТИ ИНФОРМАЦИИ, УСТАНАВЛИВАЮЩИЕ 6 УРОВНЕЙ ДОВЕРИЯ (ПРИКАЗ ФСТЭК РОССИИ ОТ 02.06.2020 № 76)



**4 УРОВЕНЬ ДОВЕРИЯ (ОБЪЕКТЫ КИИ 1 КАТЕГОРИИ, ГИС 1 КЛАССА ЗАЩИЩЕННОСТИ)**

- модель безопасности, включая реализуемые политики управления доступом;
- идентификация и анализ скрытых каналов по памяти.

**3 УРОВЕНЬ ДОВЕРИЯ (ИС, В КОТОРЫХ ОБРАБАТЫВАЕТСЯ ИНФОРМАЦИЯ, СОДЕРЖАЩАЯ СЕКРЕТНЫЕ СВЕДЕНИЯ)**

- верификация модели безопасности с использованием инструментальных средств;
- формальное (математическое) описание подсистем, реализующих функции безопасности;
- идентификация и анализ скрытых каналов по времени.

**2 УРОВЕНЬ ДОВЕРИЯ (ИС, В КОТОРЫХ ОБРАБАТЫВАЕТСЯ ИНФОРМАЦИЯ, СОДЕРЖАЩАЯ СОВЕРШЕННО СЕКРЕТНЫЕ СВЕДЕНИЯ)**

- формальное (математическое) описание модулей, реализующих функции безопасности.

**1 УРОВЕНЬ ДОВЕРИЯ (ИС, В КОТОРЫХ ОБРАБАТЫВАЕТСЯ ИНФОРМАЦИЯ, СОДЕРЖАЩАЯ СВЕДЕНИЯ ОСОБОЙ ВАЖНОСТИ)**

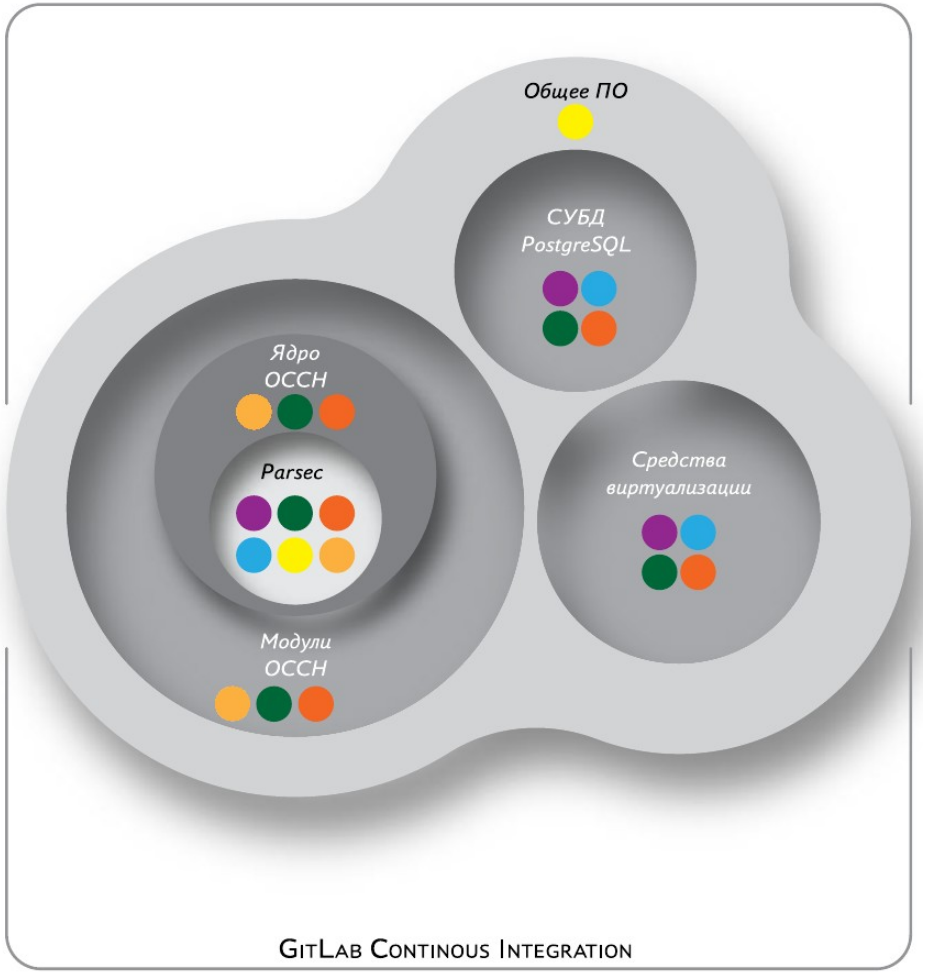
- идентификация и анализ скрытых статистических каналов.

# ВЕРИФИКАЦИЯ И АНАЛИЗ КОДА ОССН В СООТВЕТСТВИИ С МЕТОДИКОЙ ВЫЯВЛЕНИЯ УЯЗВИМОСТЕЙ И НЕДЕКЛАРИРОВАННЫХ ВОЗМОЖНОСТЕЙ



Для верификации программного кода компонент ОССН в зависимости от степени их важности для обеспечения безопасности (от основы **поверхности атаки — модуля PARSEC**) применяются:

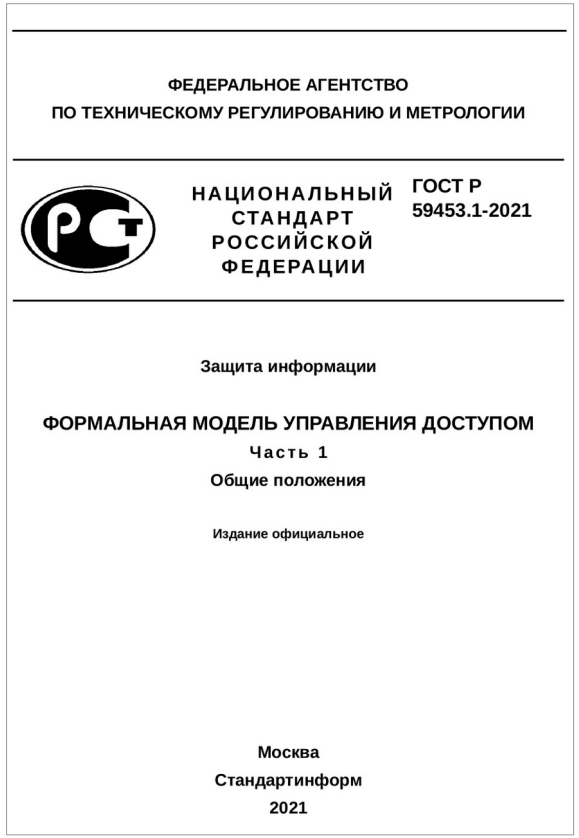
- **Иерархическая МРОСЛ ДП-модель** – основа реализованных в ОССН мандатных управления доступом и контроля целостности;
- Комплекс инструментальных средств **верификации** МРОСЛ ДП-модели и её реализации в программном коде ОССН (Rodin, ProB, Frama-C);
- Инструментальные средства **статического анализа** программного кода (Svace, Clang Static Analyzer, cppcheck, АК-ВС);
- Инструментальные средства **динамического анализа** (фаззинга) программного кода (Crusher, AFL, Syzkaller4Astra);
- Инструментальное средство **сбора трасс и анализа помеченных данных** (Блесна);
- Система **непрерывной разработки и интеграции** GitLab.




- МРОСЛ ДП-модель
- Rodin, ProB, Frama-C
- Svace, Clang SA, cppcheck, АК-ВС
- Syzkaller4Astra
- Блесна
- Crusher, AFL

# ГОСТ Р 59453.1,2-2021 «ЗАЩИТА ИНФОРМАЦИИ. ФОРМАЛЬНАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ»





ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 59453.1-2021

Защита информации

ФОРМАЛЬНАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ

Часть 1

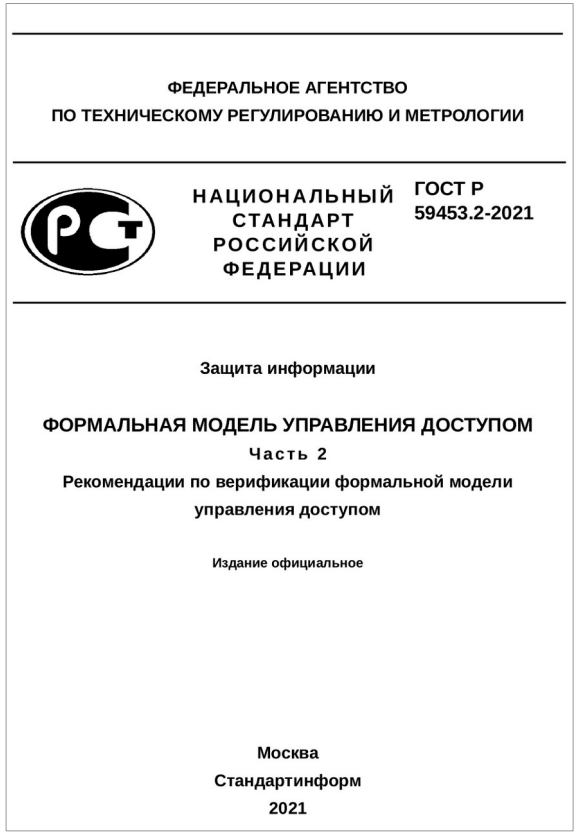
Общие положения

Издание официальное

Москва
Стандартинформ
2021

**СОДЕРЖАНИЕ**





ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 59453.2-2021

Защита информации

ФОРМАЛЬНАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ

Часть 2

Рекомендации по верификации формальной модели управления доступом

Издание официальное

Москва
Стандартинформ
2021

**СОДЕРЖАНИЕ**

# ГОСТ Р 59453.1-2021. ПРИМЕРЫ ОПРЕДЕЛЕНИЙ



> **формальная модель управления доступом**: Математическое или формализованное (машиночитаемое, пригодное для автоматизированной обработки) описание средства защиты информации и компонентов среды его функционирования, предоставление доступов между которыми регламентируется политиками управления доступом, реализуемыми этим средством защиты информации.

> **объект доступа, ассоциированный функционально с субъектом доступа:** Объект доступа, содержание информации в котором влияет на функциональность субъекта доступа, и информационный поток по памяти к этому объекту от другого субъекта доступа позволяет второму субъекту доступа получить управление первым субъектом доступа.

> **политика мандатного контроля целостности:** Политика управления доступом, при реализации которой задаются классификационные метки (уровни целостности): каждому объекту и субъекту доступа присваивается уровень целостности; субъект доступа может получить доступ к объекту доступа или другому субъекту доступа только в случае, когда выполняются следующие правила:

- при получении доступа на запись к объекту доступа уровень целостности субъекта доступа должен быть не ниже уровня целостности объекта доступа;
- доступ субъекта доступа к объекту или другому субъекту доступа не приводит к получению субъектом доступа управления некоторым субъектом доступа, уровень целостности которого не сравним или выше уровня целостности первого субъекта доступа.

# ФОРМАЛЬНАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ ОССН (МРОСЛ ДП-МОДЕЛЬ) В МАТЕМАТИЧЕСКОЙ НОТАЦИИ



# ПРИМЕРЫ ОПРЕДЕЛЕНИЯ ЭЛЕМЕНТОВ МОДЕЛИ ДЛЯ УРОВНЯ 1.2 (РОЛЕВОЕ УПРАВЛЕНИЕ ДОСТУПОМ В СУБД POSTGRESQL)



$DB\_R$ — множество ролей СУБД;
$DB\_ADMIN\_PRIVILEGES = \{SUPERUSER, CREATEROLE, CREATEDB, LOGIN, REPLICATION, INHERIT\}$ — множество административных привилегий СУБД;
$DB\_AP: DB\_R \to 2^{DB\_ADMIN\_PRIVILEGES}$ — функция административных привилегий ролей СУБД;
$db\_login: S \to \{r \in DB\_R : LOGIN \in DB\_AP(r)\}$ — функция роли входа субъект-сессии в СУБД;
$db\_inherit: DB\_R \to 2^{DB\_R}$ — функция наследования привилегий ролей к элементам СУБД
$db\_with\_admin\_option: DB\_R \to 2^{DB\_R}$ — функция управления подчинённостью ролей в иерархии;
$DB\_E = DB\_O \cup DB\_C$ — множество элементов СУБД, не являющихся ролями, где $DB\_O$ — множество элементов-объектов СУБД, $DB\_C$ — множество элементов-контейнеров СУБД;
$DB\_PRIVILEGES = \{SELECT, INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES, TRIGGER, USAGE, CREATE, CONNECT, TEMPORARY, TEMP, EXECUTE, OWN\}$ – множество видов привилегий СУБД;
$DB\_E\_E \subseteq DB\_E$ — множество сущностей СУБД;
$db\_path: DB\_E\_E \to 2^E$ — функция, ставящая в соответствие сущностям СУБД сущности ОССН, в которых они содержатся;
$db\_privileges: DB\_R \to 2^{(DB\_E \times DB\_PRIVILEGES) \times DB\_R}$ — функция привилегий к элементам СУБД ролей СУБД;
$A \subseteq S \times (E \cup DB\_E\_E) \times R_a$ — множество доступов субъект-сессий к сущностям или сущностям СУБД;
$AA \subseteq S \times (R \cup NR \cup AR \cup DB\_R) \times R_a$ — множество доступов субъект-сессий к ролям, запрещающим ролям, административным ролям или ролям СУБД.

# ПРИМЕР УСЛОВИЙ БЕЗОПАСНОСТИ ДЛЯ УРОВНЯ 2.2 (МАНДАТНЫЙ КОНТРОЛЬ ЦЕЛОСТНОСТИ В СУБД POSTGRESQL)



**Теорема т.Ц.01.БДЦ.** Пусть $G_0$ – безопасное начальное состояние системы $\Sigma(G^*, OP, G_0)$. Пусть на всех траекториях системы без кооперации доверенных или недоверенных субъект-сессий $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$, где $N \geq 0$, и в каждом состоянии $G_N$ для каждой субъект-сессии $s \in S_N$, сущности или сущности СУБД $e \in E_N \cup DB\_E\_E_N$ выполняются следующие условия условия.

***Условие* Ц.1.** (*корректность уровней целостности сущностей, функционально ассоциированных с субъект-сессиями*) Если $e \in [s]$, то выполняется условие $i_{sN}(s) \leq i_{eN}(e)$.

***Условие* Ц.2.** (*корректность уровней целостности, а также прав доступа на чтение к сущностям, параметрически ассоциированным с субъект-сессиями*) Если $e \in ]s[$, то $i_{sN}(s) \leq i_{eN}(e)$ и для каждой роли или административной роли $r \in R_N \cup AR_N$ такой, что $(e, read_r) \in PA_N(r)$, выполняется условие $i_{eN}(e) \leq i_{rN}(r)$.

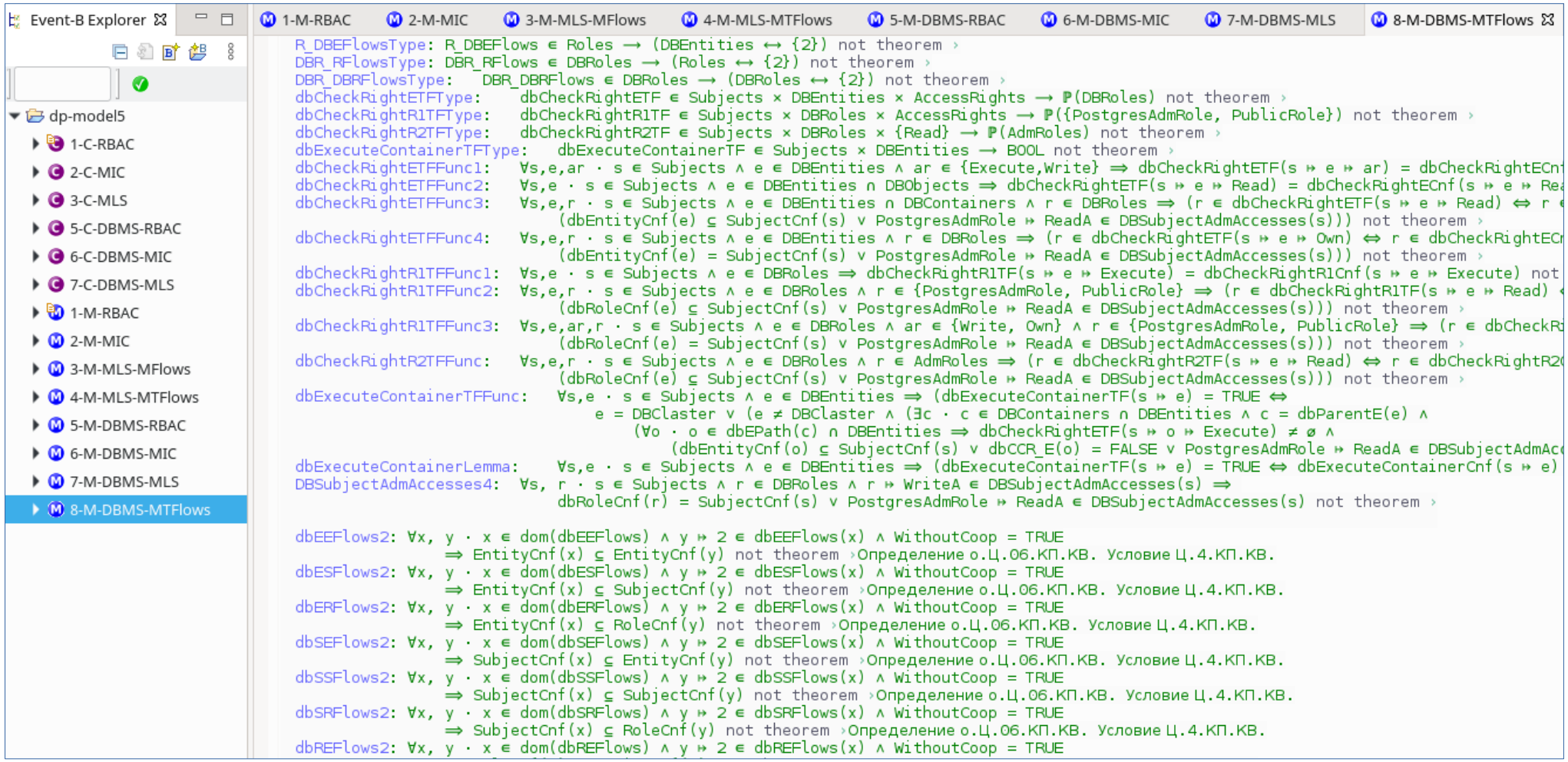
***Условие* Ц.3.БДЦ.** (*функциональная и параметрическая корректность всех доверенных субъект-сессий относительно всех доверенных субъект-сессий и сущностей ОССН и СУБД*) Для всех субъект-сессий $s \in S_N$ таких, что $i\_low < i_{sN}(s)$, выполняются условия $\{s' \in S_N \mid i\_low < i_{sN}(s') \leq i_{sN}(s)\} \times (E_N \cup DB\_E\_E_N \cup S_N) \subset f\_correct_N(s)$, $\{s' \in S_N \mid i\_low < i_{sN}(s') \leq i_{sN}(s)\} \times (E_N \cup S_N) \subset p\_correct_N(s)$.

***Условие* БДЦ.4.** (*неизменность множества доверенных субъект-сессий СУБД*) Множество доверенных субъект-сессий СУБД не меняется на траекториях функционирования системы: $DB\_L_{SN} = DB\_L_{S0}$. Для каждых субъект-сессий $x, y \in S_N$ выполняется, если $y \in DB\_L_{S0} \cap de\_facto\_own_N(x)$, то $x \in DB\_L_{S0}$.

Тогда на этих траекториях система $\Sigma(G^*, OP, G_0)$ безопасна в смысле мандатного контроля целостности.

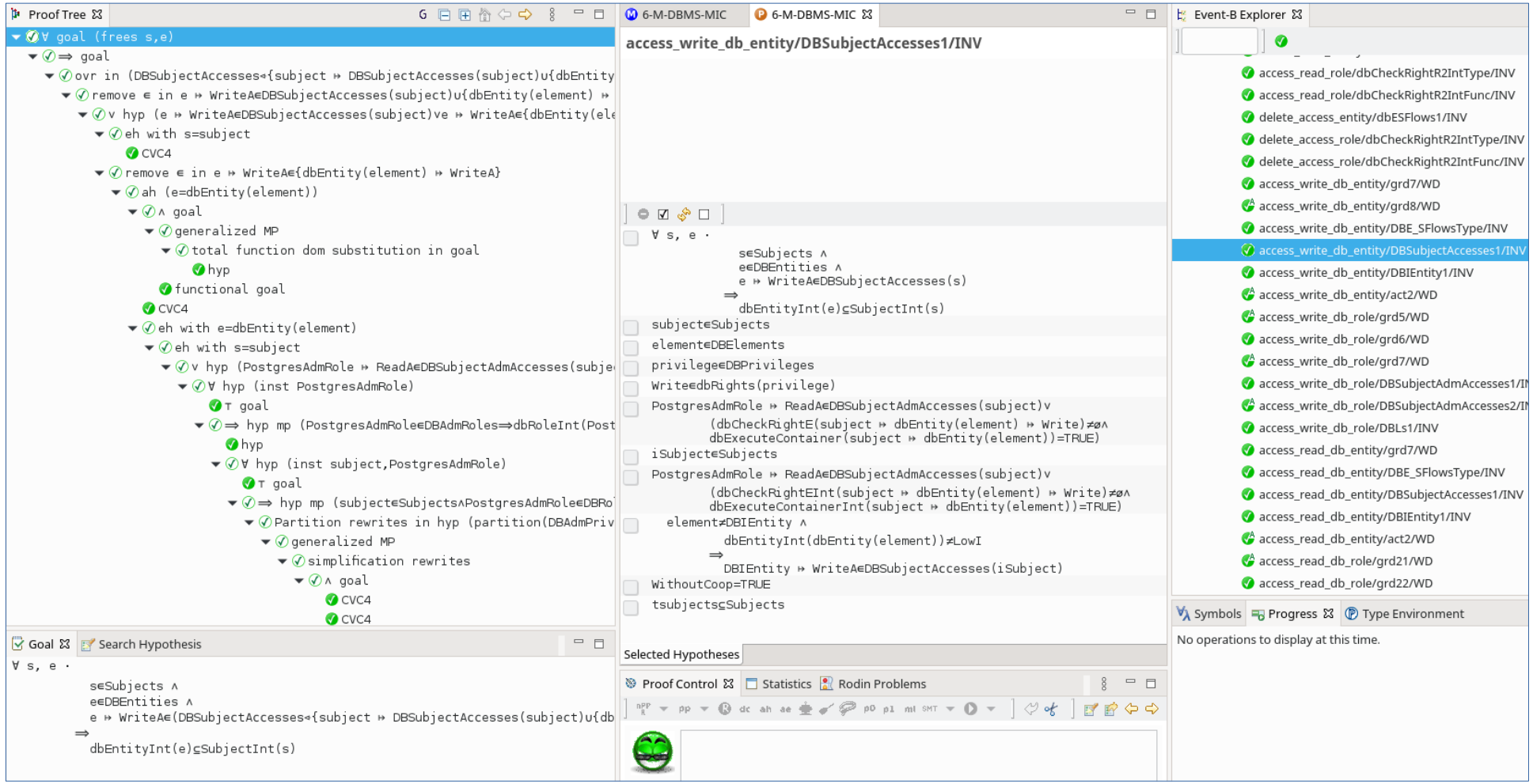
# ФРАГМЕНТ ЗАДАНИЯ В ФОРМАЛИЗОВАННОЙ НОТАЦИИ УРОВНЯ 4.2 (МАНДАТНОГО УПРАВЛЕНИЯ ДОСТУПОМ С ПОТОКАМИ ПО ВРЕМЕНИ В СУБД POSTGRESQL)





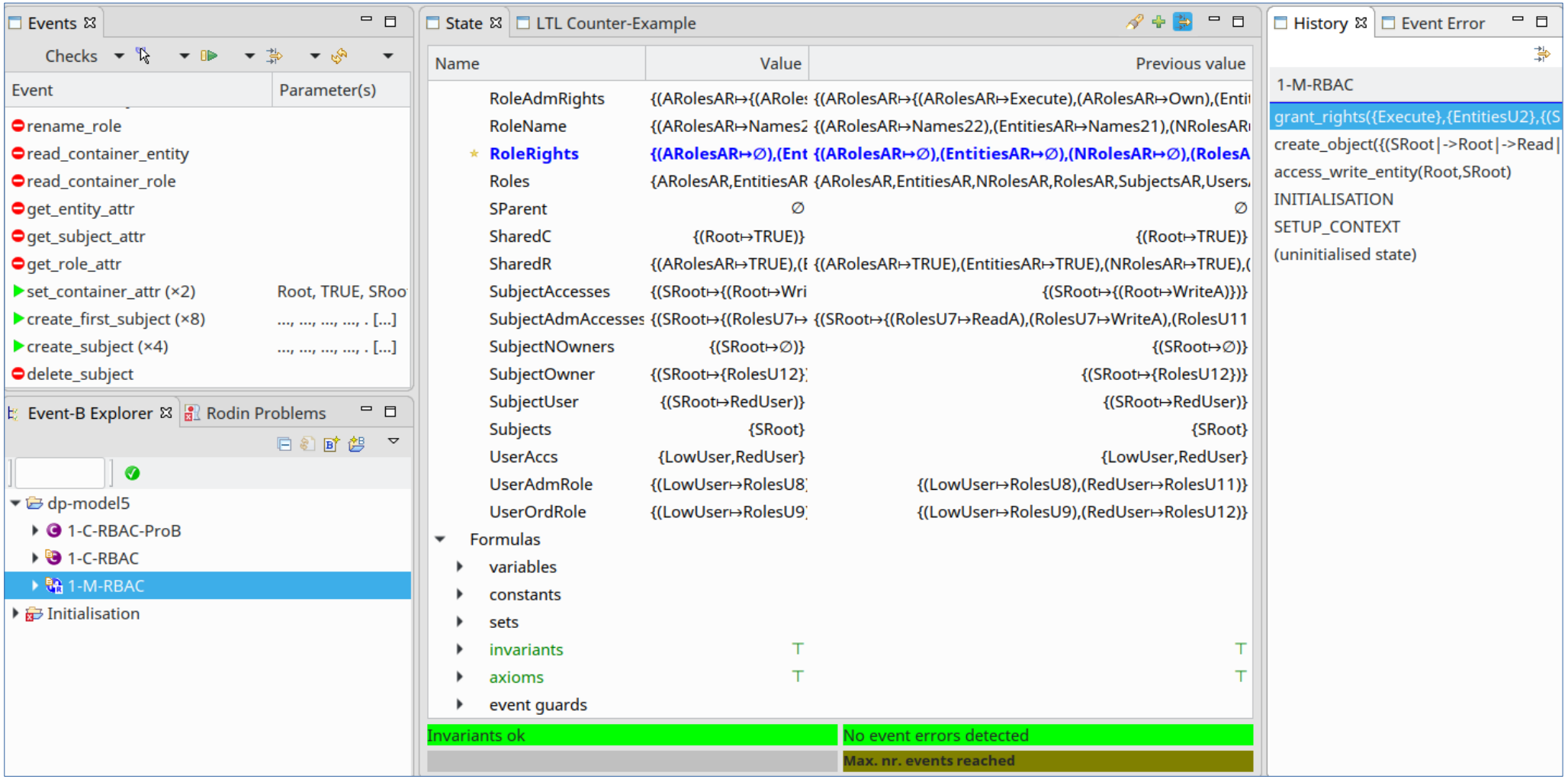
# ВЕРИФИКАЦИЯ МРОСЛ ДП-МОДЕЛИ (УРОВНЯ 2.2) В ФОРМАЛИЗОВАННОЙ НОТАЦИИ НА ЯЗЫКЕ МЕТОДА EVENT-B ИНСТРУМЕНТОМ RODIN





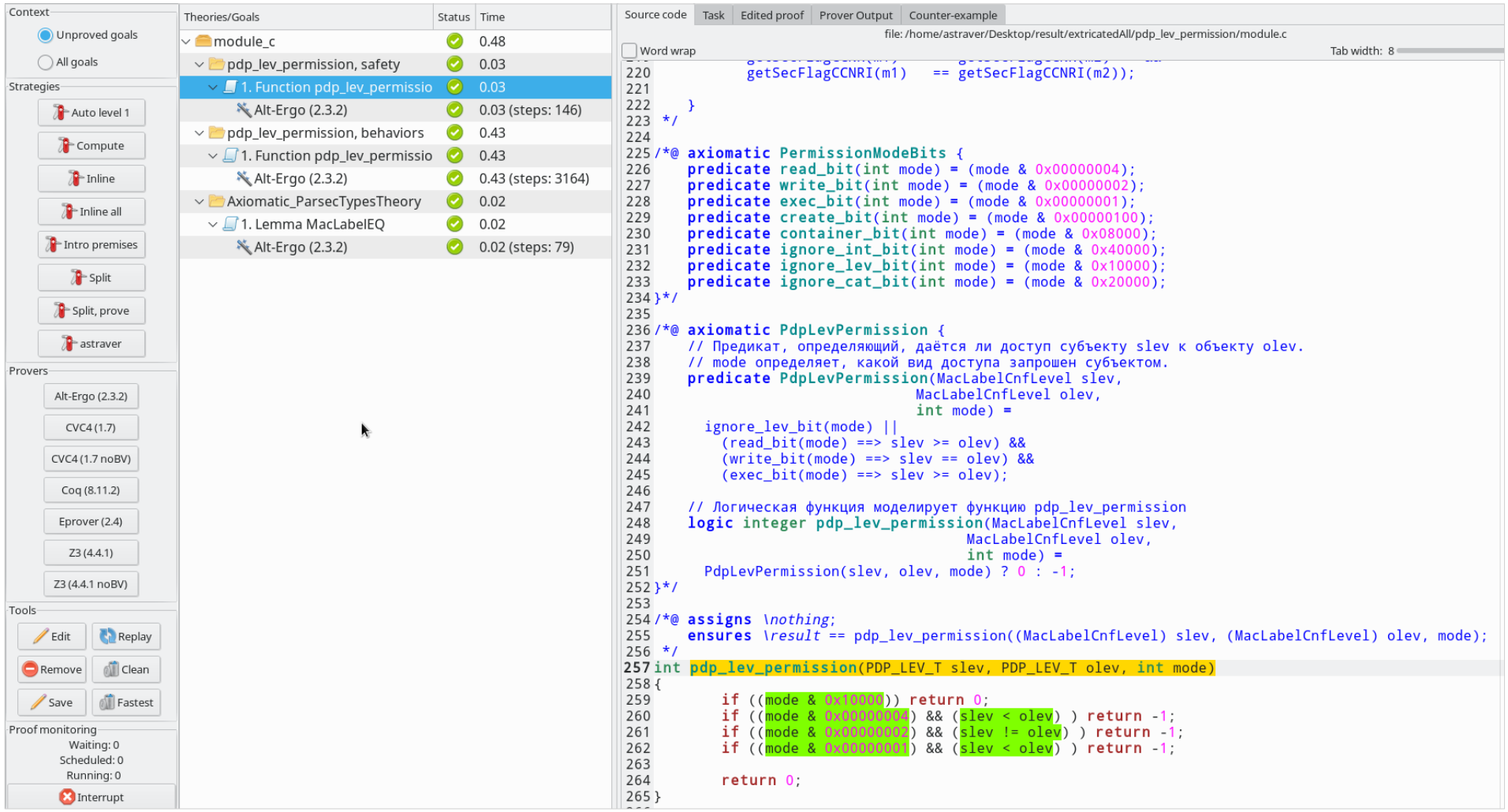
# ВЕРИФИКАЦИЯ МРОСЛ ДП-МОДЕЛИ (УРОВНЯ 1.1) В ФОРМАЛИЗОВАННОЙ НОТАЦИИ ПО МЕТОДУ ПРОВЕРКИ МОДЕЛЕЙ ИНСТРУМЕНТОМ PROB





# ВЕРИФИКАЦИЯ СПЕЦИФИКАЦИЙ НА ЯЗЫКЕ ACSL ПРОГРАММНОЙ РЕАЛИЗАЦИИ МРОСЛ ДП-МОДЕЛИ ИНСТРУМЕНТОМ FRAMA-C





# СТАТИЧЕСКИЙ АНАЛИЗ КОДА ОССН С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТА CLANG STATIC ANALYZER



```
2788  static void
2789  _ignore_completion_names (names, name_func)
2790       char **names;
2791       sh_ignore_func_t *name_func;
2792  {
2793    char **newnames;
2794    int idx, nidx;
2795    char **oldnames;
```

9 ← 'oldnames' declared without an initial value →

```
2796    int oidx;
2797
2798    /* If there is only one completion, see if it is acceptable.  If it is
2799       not, free it up.  In any case, short-circuit and return.  This is a
2800       special case because names[0] is not the prefix of the list of names
2801       if there is only one completion; it is the completion itself. */
2802    if (names[1] == (char *)0)
```

10 ← Assuming the condition is false →

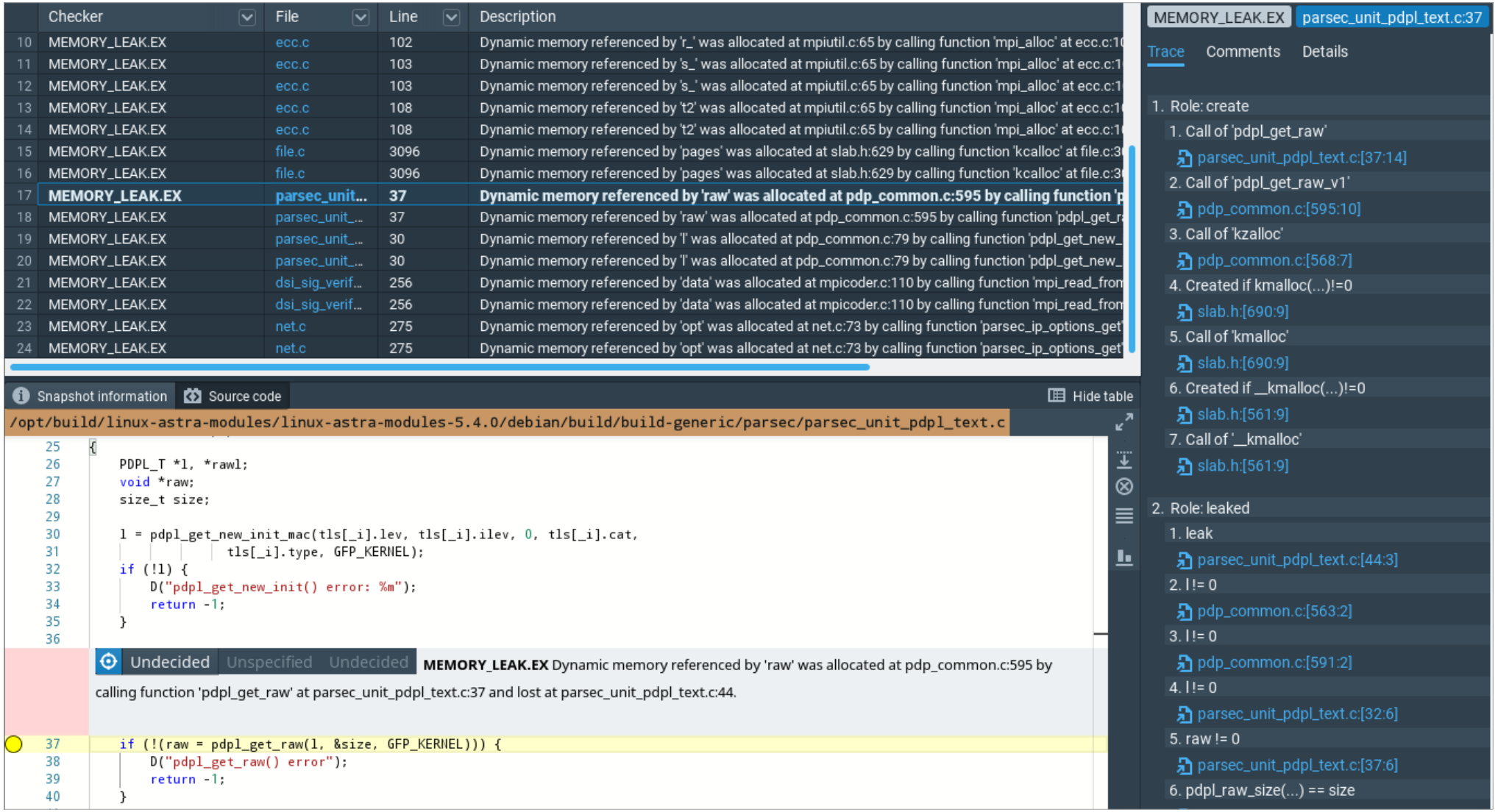
11 ← Taking false branch →

```
2803      {
2804        if (force_fignore)
2805          if ((*name_func) (names[0]) == 0)
2806            {
2807              free (names[0]);
2808              names[0] = (char *)NULL;
2809            }
2810
2811        return;
2812      }
```

| Bug Type | Quantity | Display? |
|---|---|---|
| **All Bugs** | **239** | ■ |
| API | | |
| Argument with 'nonnull' attribute passed null | 12 | ■ |
| Dead store | | |
| Dead assignment | 152 | ■ |
| Dead increment | 3 | ■ |
| Dead initialization | 1 | ■ |
| Logic error | | |
| Assigned value is garbage or undefined | 3 | ■ |
| Branch condition evaluates to a garbage value | 4 | ■ |
| Dereference of null pointer | 47 | ■ |

# СТАТИЧЕСКИЙ АНАЛИЗ КОДА ОССН С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТА SVACE (ИСП РАН)



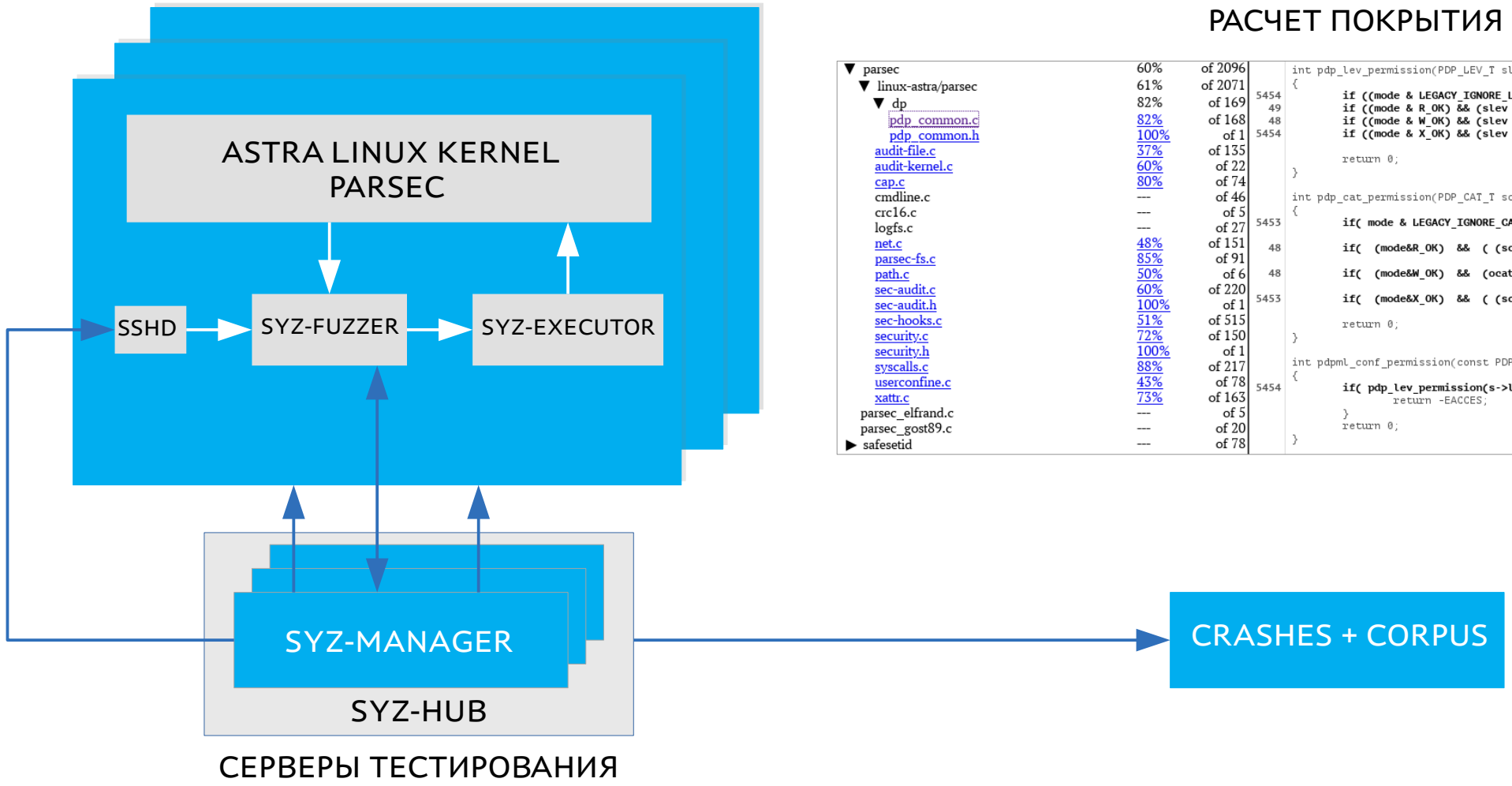


| | Checker | File | Line | Description |
|---|---|---|---|---|
| 10 | MEMORY_LEAK.EX | ecc.c | 102 | Dynamic memory referenced by 'r_' was allocated at mpiutil.c:65 by calling function 'mpi_alloc' at ecc.c:1 |
| 11 | MEMORY_LEAK.EX | ecc.c | 103 | Dynamic memory referenced by 's_' was allocated at mpiutil.c:65 by calling function 'mpi_alloc' at ecc.c:1 |
| 12 | MEMORY_LEAK.EX | ecc.c | 103 | Dynamic memory referenced by 's_' was allocated at mpiutil.c:65 by calling function 'mpi_alloc' at ecc.c:1 |
| 13 | MEMORY_LEAK.EX | ecc.c | 108 | Dynamic memory referenced by 't2' was allocated at mpiutil.c:65 by calling function 'mpi_alloc' at ecc.c:1 |
| 14 | MEMORY_LEAK.EX | ecc.c | 108 | Dynamic memory referenced by 't2' was allocated at mpiutil.c:65 by calling function 'mpi_alloc' at ecc.c:1 |
| 15 | MEMORY_LEAK.EX | file.c | 3096 | Dynamic memory referenced by 'pages' was allocated at slab.h:629 by calling function 'kcalloc' at file.c:3 |
| 16 | MEMORY_LEAK.EX | file.c | 3096 | Dynamic memory referenced by 'pages' was allocated at slab.h:629 by calling function 'kcalloc' at file.c:3 |
| 17 | **MEMORY_LEAK.EX** | **parsec_unit...** | **37** | **Dynamic memory referenced by 'raw' was allocated at pdp_common.c:595 by calling function 'p** |
| 18 | MEMORY_LEAK.EX | parsec_unit_... | 37 | Dynamic memory referenced by 'raw' was allocated at pdp_common.c:595 by calling function 'pdpl_get_r |
| 19 | MEMORY_LEAK.EX | parsec_unit_... | 30 | Dynamic memory referenced by 'l' was allocated at pdp_common.c:79 by calling function 'pdpl_get_new_ |
| 20 | MEMORY_LEAK.EX | parsec_unit_... | 30 | Dynamic memory referenced by 'l' was allocated at pdp_common.c:79 by calling function 'pdpl_get_new_ |
| 21 | MEMORY_LEAK.EX | dsi_sig_verif... | 256 | Dynamic memory referenced by 'data' was allocated at mpicoder.c:110 by calling function 'mpi_read_fron |
| 22 | MEMORY_LEAK.EX | dsi_sig_verif... | 256 | Dynamic memory referenced by 'data' was allocated at mpicoder.c:110 by calling function 'mpi_read_fron |
| 23 | MEMORY_LEAK.EX | net.c | 275 | Dynamic memory referenced by 'opt' was allocated at net.c:73 by calling function 'parsec_ip_options_get' |
| 24 | MEMORY_LEAK.EX | net.c | 275 | Dynamic memory referenced by 'opt' was allocated at net.c:73 by calling function 'parsec_ip_options_get' |

# ДИНАМИЧЕСКИЙ АНАЛИЗ КОДА ОССН С ИСПОЛЬЗОВАНИЕМ СТЕНДА SYZKALLER4ASTRA



ВИРТУАЛЬНЫЕ МАШИНЫ С ОССН




СЕРВЕРЫ ТЕСТИРОВАНИЯ

БАЗА ДАННЫХ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ

РАСЧЕТ ПОКРЫТИЯ КОДА

| | | |
|---|---|---|
| ▼ parsec | 60% | of 2096 |
| ▼ linux-astra/parsec | 61% | of 2071 |
| ▼ dp | 82% | of 169 |
| pdp_common.c | 82% | of 168 |
| pdp_common.h | 100% | of 1 |
| audit-file.c | 37% | of 135 |
| audit-kernel.c | 60% | of 22 |
| cap.c | 80% | of 74 |
| cmdline.c | --- | of 46 |
| crc16.c | --- | of 5 |
| logfs.c | --- | of 27 |
| net.c | 48% | of 151 |
| parsec-fs.c | 85% | of 91 |
| path.c | 50% | of 6 |
| sec-audit.c | 60% | of 220 |
| sec-audit.h | 100% | of 1 |
| sec-hooks.c | 51% | of 515 |
| security.c | 72% | of 150 |
| security.h | 100% | of 1 |
| syscalls.c | 88% | of 217 |
| userconfine.c | 43% | of 78 |
| xattr.c | 73% | of 163 |
| parsec_elfrand.c | --- | of 5 |
| parsec_gost89.c | --- | of 20 |
| ▶ safesetid | --- | of 78 |

```
int pdp_lev_permission(PDP_LEV_T slev, PDP_LEV_T olev, int mode)
{
5454        if ((mode & LEGACY_IGNORE_LEV)) return 0;
  49        if ((mode & R_OK) && (slev <  olev) ) return -EACCES;
  48        if ((mode & W_OK) && (slev != olev) ) return -EACCES;
5454        if ((mode & X_OK) && (slev <  olev) ) return -EACCES;

            return 0;
}

int pdp_cat_permission(PDP_CAT_T scat, PDP_CAT_T ocat, int mode)
{
5453        if( mode & LEGACY_IGNORE_CAT ) return 0;

  48        if(  (mode&R_OK)  &&  ( (scat & ocat) != ocat )  ) return -EACCES;

  48        if(  (mode&W_OK)  &&  (ocat != scat)  ) return -EACCES;

5453        if(  (mode&X_OK)  &&  ( (scat & ocat) != ocat )  ) return -EACCES;

            return 0;
}

int pdpml_conf_permission(const PDPML_T *s, const PDPML_T *o, int mode)
{
5454        if( pdp_lev_permission(s->lev,o->lev,mode) || pdp_cat_permission(s->cat,o->cat,mode) ) {
                return -EACCES;
            }
            return 0;
}
```

# РЕЗУЛЬТАТЫ ФАЗЗИНГ-ТЕСТИРОВАНИЯ ИНСТРУМЕНТАМИ CRUSHER (ИСП РАН), AFL И SYZKALLER4ASTRA МЕХАНИЗМОВ ЗАЩИТЫ ОССН



**LCOV - code coverage report**

| | | | Hit | Total | Coverage |
|---|---|---|---|---|---|
| Current view: | top level - utils-pdp | Lines: | 2409 | 3009 | 80.1 % |
| Test: | cov.info | Functions: | 127 | 163 | 77.9 % |
| Date: | 2021-08-12 05:10:30 | | | | |

| Filename | Line Coverage | | Functions | |
|---|---|---|---|---|
| ls_pdp.c | 100.0 % | 33 / 33 | 100.0 % | 2 / 2 |
| ls_proc.c | 67.4 % | 698 / 1036 | 76.2 % | 48 / 63 |
| pdp-id.c | 90.1 % | 64 / 71 | 100.0 % | 2 / 2 |
| pdp-ls.c | 86.3 % | 1212 / 1404 | 74.7 % | 56 / 75 |
| pdpl-file.c | 82.9 % | 189 / 228 | 100.0 % | 7 / 7 |
| pdpl-ps.c | 89.2 % | 58 / 65 | 100.0 % | 4 / 4 |
| pdpl-user.c | 90.1 % | 155 / 172 | 80.0 % | 8 / 10 |

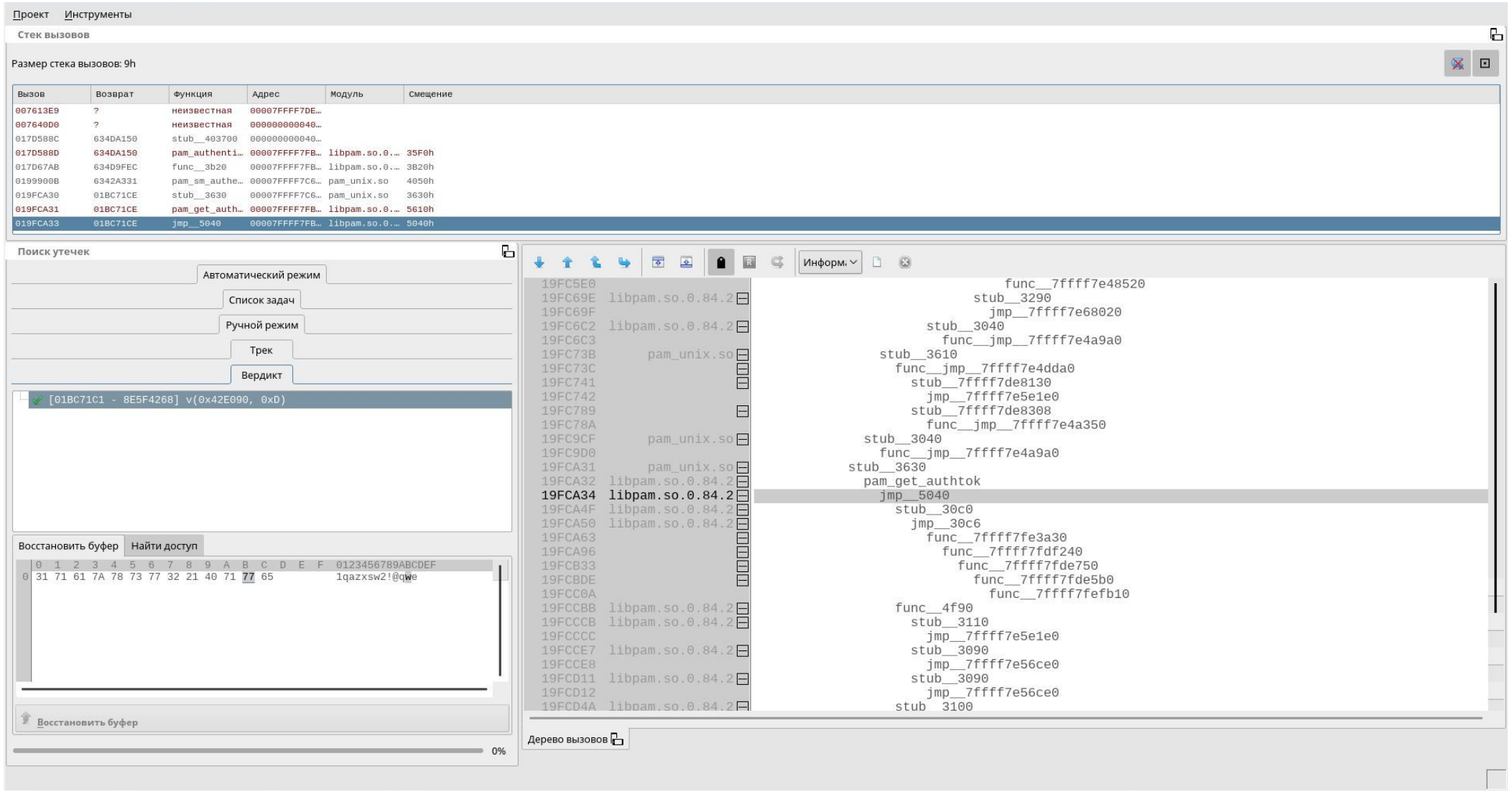
**70%** – покрытие модулей безопасности PARSEC по базовым блокам;

**80%** – покрытие модулей пользовательского пространства по строкам;

**72 часа** – среднее время устранения выявляемых ошибок в модулях безопасности.

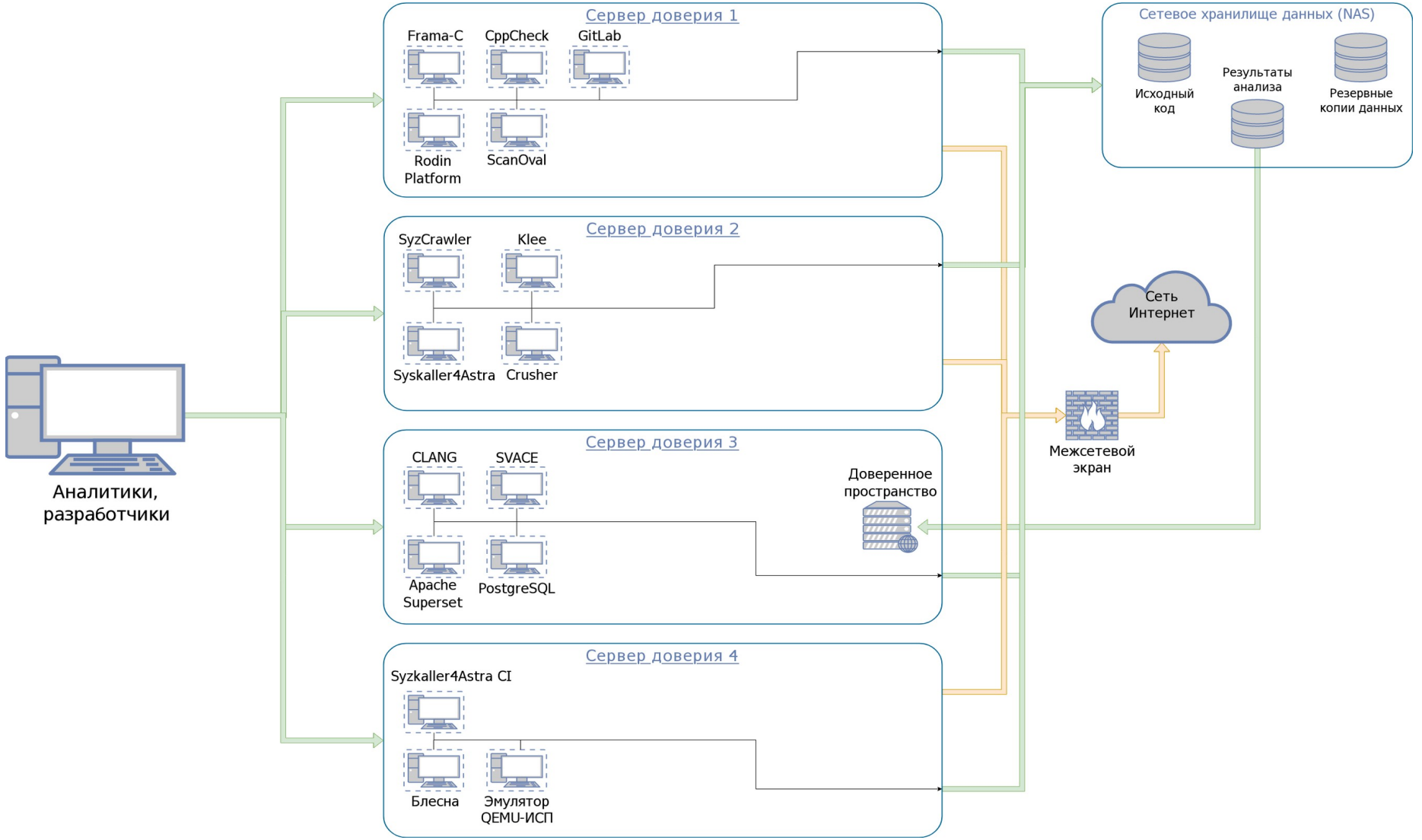
# АНАЛИЗ ПОМЕЧЕННЫХ ДАННЫХ ПРИ СБОРЕ ТРАСС ПРОГРАММ ИНСТРУМЕНТОМ БЛЕСНА (ИСП РАН)





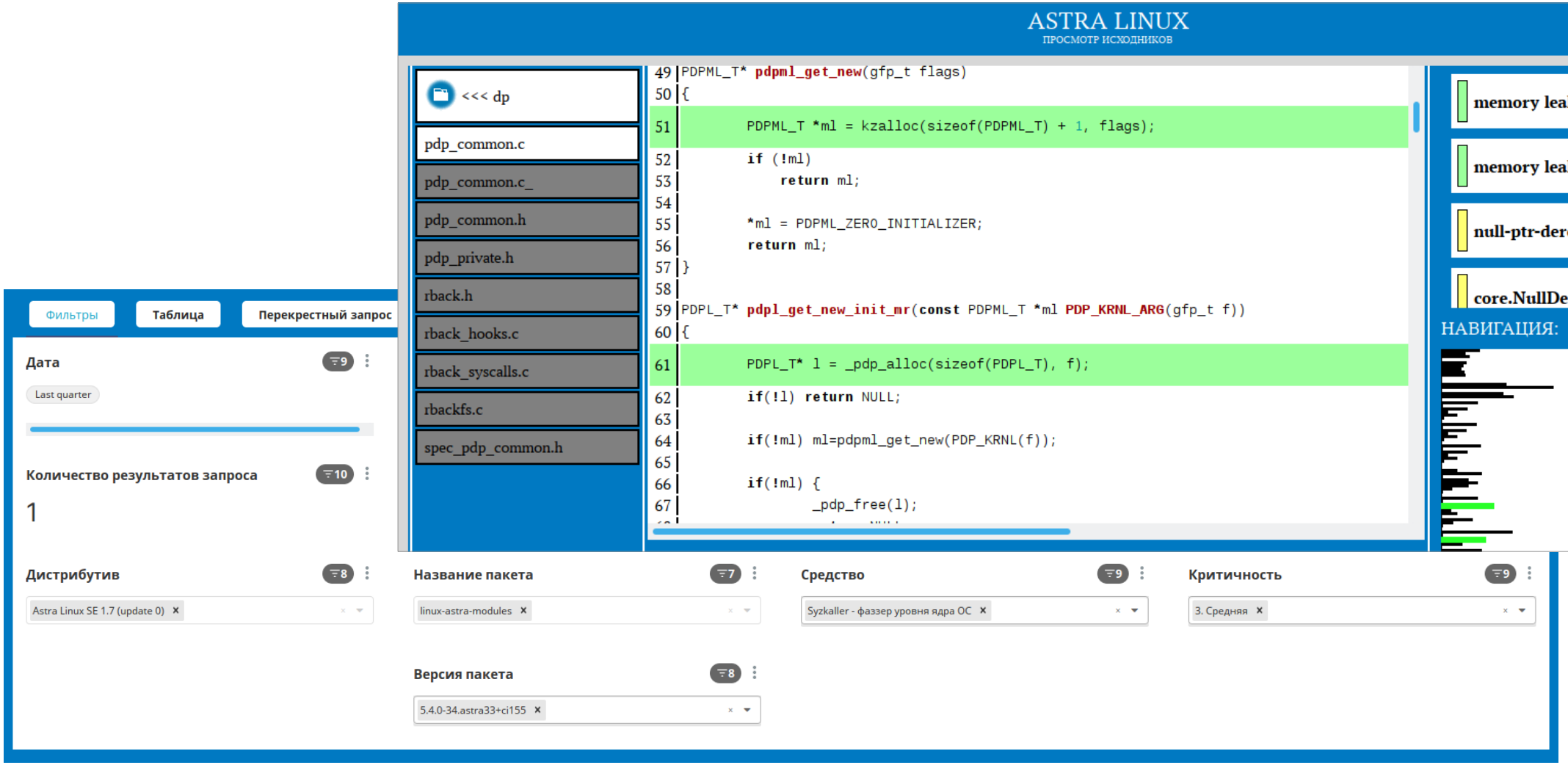
# МАСШТАБИРУЕМАЯ СТРУКТУРА СТЕНДА ДОВЕРИЯ ДЛЯ ВЕРИФИКАЦИИ И АНАЛИЗА КОДА ОССН





# ОСНОВНОЙ ИНТЕРФЕЙС БАЗЫ ДАННЫХ ДОВЕРИЯ





# Ошибка 93

(критичность: **4**, аналоги к-во abs: **0**)

Найдена в linux-astra-modules-5.4.0/linux-astra/parsec/parsec_unit_pdpl_text.c:37 (уник. номер **104826714**) Вверх к описанию 93. Нет HTML для Svace

**Dynamic memory referenced by 'raw' was allocated at pdp_common.c:595 by calling function 'pdpl_get_raw' at parsec_unit_pdpl_text.c:37 and lost at parsec_unit_pdpl_text.c:44.**

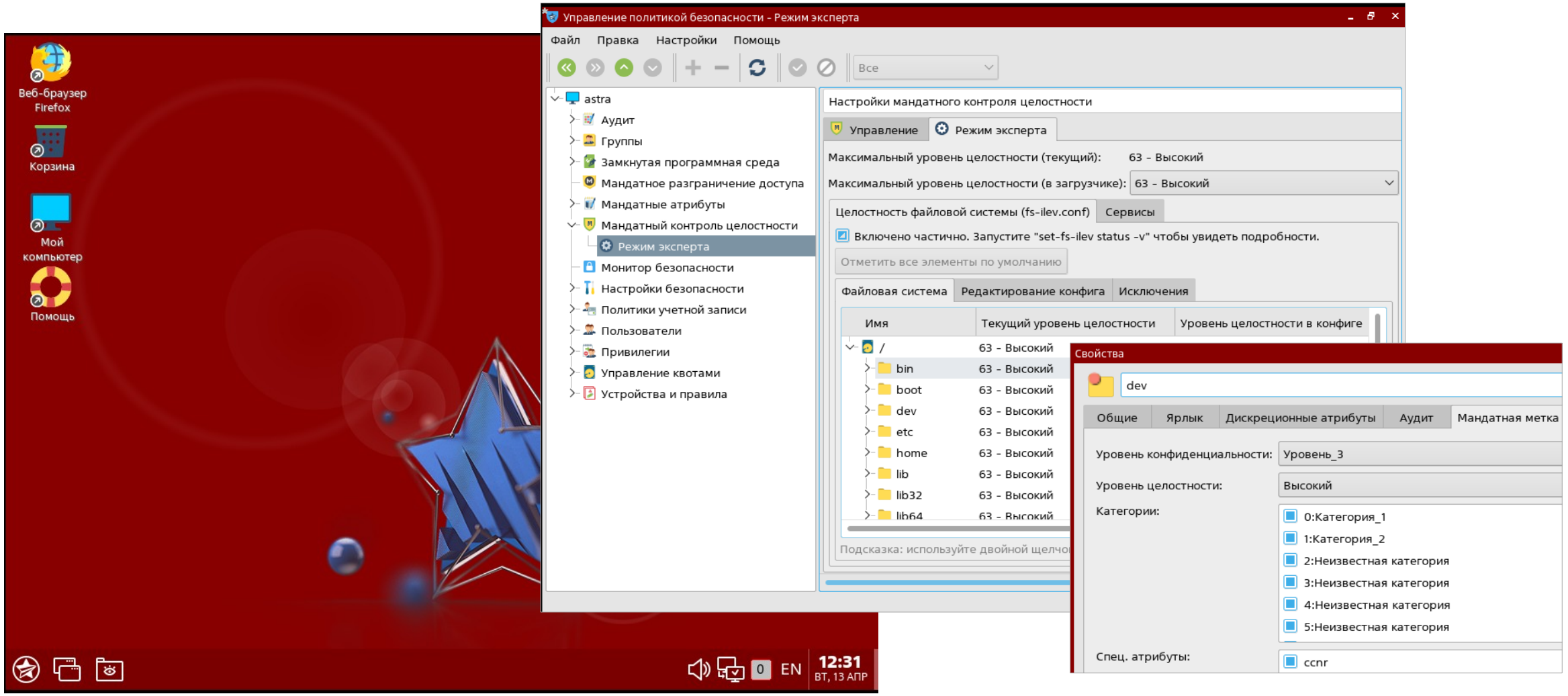
ЗАКЛЮЧЕНИЕ АНАЛИТИКА: В файле linux-astra/parsec/parsec_unit_pdpl.c:30 выполняется выделение памяти в pdpl_get_new_init_mac и выход из функции (return -1) в строке 39 в случае перехода в конструкцию if, при условии, что функция pdpl_get_raw возвращает NULL. В файле linux-astra/parsec/pdp_common.c:599 вызывается функция _pdp_alloc, если память в ней не выделится, то функция pdpl_get_raw возвращает NULL.

Участок исследуемого кода:

```
17  struct PDP_TEST_LABEL_T tls[] = {
18      {0,    0, 0x00, 0x0, "0:0:0:0",    0, 14},     /*0 */
19      {0,    0xFFFF, 0x00, 0x0, "0:0:0:0",    0, 18},    /*1 */
20  };
22  const int tls_size = sizeof(tls) / sizeof(struct PDP_TEST_LABEL_T);
24  int __init test_pdpl_2from_raw(int _i)
25  {
26      PDPL_T *l, *rawl;
27      void *raw;
28      size_t size;
30      l = pdpl_get_new_init_mac(tls[_i].lev, tls[_i].ilev, 0, tls[_i].cat,
31                    tls[_i].type, GFP_KERNEL);
32      if (!l) {
33          D("pdpl_get_new_init() error: %m");
34          return -1;
35      }
37      if (!(raw = pdpl_get_raw(l, &size, GFP_KERNEL))) { // ------------- Ошибка сработала здесь
38          D("pdpl_get_raw() error");
39          return -1;
40      }
42      if ((pdpl_raw_size(l) != size) || (size != tls[_i].raw_size)) {
```

# РЕАЛИЗАЦИЯ НА ОСНОВЕ МРОСЛ ДП-МОДЕЛИ ВЕРИФИЦИРОВАННОГО МЕХАНИЗМА ЗАЩИТЫ ОССН ASTRA LINUX SPECIAL EDITION





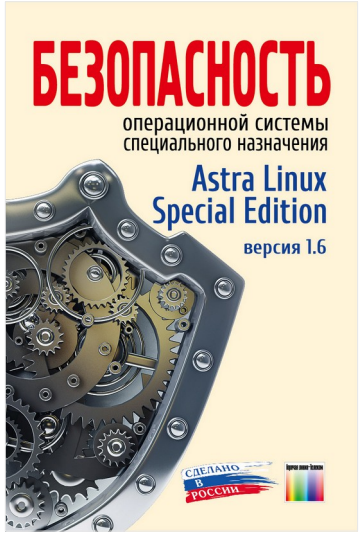
# МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ (https://astralinux.ru/information/library/publications/)

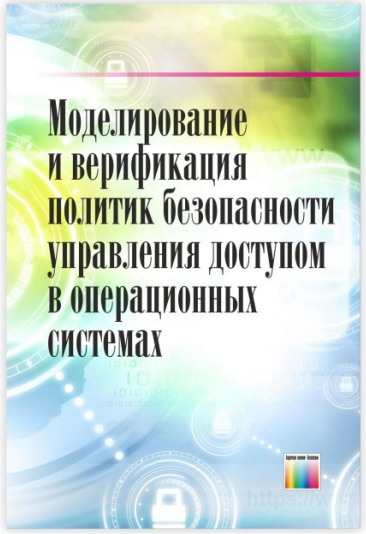


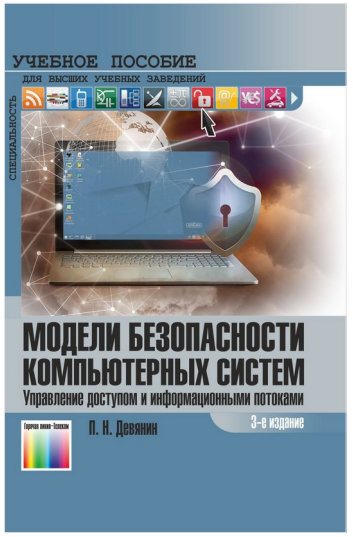
> 1 | **БЕЗОПАСНОСТЬ ОПЕРАЦИОННОЙ СИСТЕМЫ СПЕЦИАЛЬНОГО НАЗНАЧЕНИЯ ASTRA LINUX SPECIAL EDITION.** Учебное пособие для вузов / П.В. Буренин, П.Н. Девянин, Е.В. Лебеденко и др.; Под редакцией доктора техн. наук, профессора П.Н. Девянина. 3-е издание, перераб. и доп. М.: Горячая линия – Телеком, 2019, 404 с.: ил.

> 2 | Девянин П.Н., Ефремов Д.В., Кулямин В.В., Петренко А.К., Хорошилов А.В., Щепетков И.В. **МОДЕЛИРОВАНИЕ И ВЕРИФИКАЦИЯ ПОЛИТИК БЕЗОПАСНОСТИ УПРАВЛЕНИЯ ДОСТУПОМ В ОПЕРАЦИОННЫХ СИСТЕМАХ.** М.: Горячая линия — Телеком, 2019. 214 с.

> 3 | Девянин П.Н. **МОДЕЛИ БЕЗОПАСНОСТИ КОМПЬЮТЕРНЫХ СИСТЕМ. УПРАВЛЕНИЕ ДОСТУПОМ И ИНФОРМАЦИОННЫМИ ПОТОКАМИ. УЧЕБНОЕ ПОСОБИЕ ДЛЯ ВУЗОВ.** 3-е изд., перераб. и доп. М.: Горячая линия – Телеком, 2020. 352 с.




ИСП РАН

ASTRA LINUX®

ASTRA LINUX®

# Спасибо за внимание!