

Обеспечение доверия к ОС СН Astra Linux Special Edition

Девянин П.Н., научный руководитель ГК Astra Linux

КОМПОНЕНТЫ СЗИ СЕРТИФИЦИРОВАННОЙ ПО 1 КЛАССУ ЗАЩИТЫ (ДОВЕРИЯ) ОССН ASTRA LINUX SPECIAL EDITION

02



ОСНОВНЫЕ НАПРАВЛЕНИЯ ФОРМИРОВАНИЯ МЕТОДОЛОГИИ РАЗРАБОТКИ БЕЗОПАСНОГО СИСТЕМНОГО ПО

03



ИСП РАН

ТРУДЫ ИНСТИТУТА СИСТЕМНОГО ПРОГРАММИРОВАНИЯ РАН

DOI: 10.15514/ISPRAS-2021-33(5)-2

Формирование методологии разработки безопасного системного программного обеспечения на примере операционных систем

¹ П.Н. Девянин, ORCID: 0000-0003-2561-794X <pdevyanin@astralinux.ru>
¹ В.Ю. Тележников, ORCID: 0000-0002-6192-2856 <vtelezhnikov@astralinux.ru>
^{2,3,4,5} А.В. Хорошилов, ORCID: 0000-0002-6512-4632 <khoroshilov@ispras.ru>

¹ ООО «РусБИТех-Астра»,
117105, г. Москва, Варшавское ш., д. 26, стр.11
² Институт системного программирования имени В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
³ Московский государственный университет имени М.В. Ломоносова,
119991, Россия, Москва, Ленинские горы, д. 1
⁴ НИУ Высшая школа экономики,
101978, Россия, г. Москва, ул. Мясницкая, д. 20
⁵ Московский физико-технический институт,
141701, Россия, Московская область, г. Долгопрудный, Институтский пер., 9

Аннотация. Разработка безопасного системного программного обеспечения (ПО), на основе которого строятся сертифицированные средства защиты информации, достижение доверия к нему согласно требованиям нормативных документов отечественных регуляторов – крупная научно-техническая проблема. Возможным путем ее решения является формирование соответствующей методологии, которая должна включить передовые научные результаты в области информационной безопасности и системного программирования и отразить лучшие практики разработки такого ПО. В статье рассматриваются текущие результаты формирования этой методологии, которое осуществляется по следующим направлениям. Во-первых, это деятельность по развитию нормативной базы в области разработки и обеспечения доверия к безопасному системному ПО, включая создание профильных национальных стандартов. Во-вторых, разработка и верификация формальных моделей управления доступом, как основы механизмов защиты, входящих в состав системного ПО и составляющих его поверхность атаки. В-третьих, методы и технологии статического и динамического анализа программного кода системного ПО с учётом его специфики. В-четвертых, способы сбора и аналитической обработки данных, получаемых в ходе анализа программного кода системного ПО. Все направления формирования методологии иллюстрируются примерами ее практического применения и апробации при разработке ОС семейства Linux, в особенности сертифицированной по высшим классам защиты и уровням доверия операционной системы специального назначения Astra Linux Special Edition.

Ключевые слова: системное программное обеспечение; формальная модель управления доступом; верификация; статический и динамический анализ кода; операционная система; Astra Linux

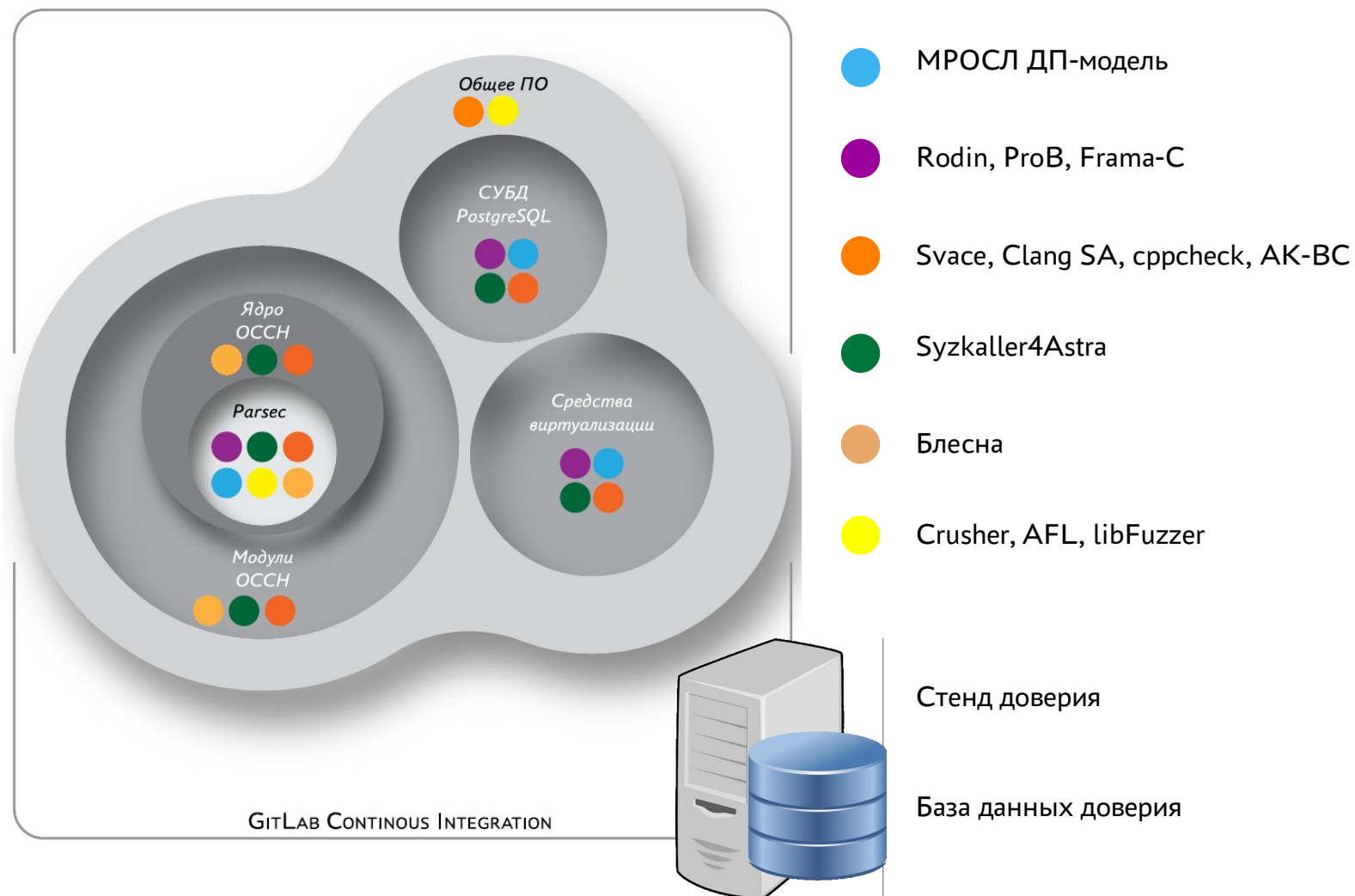
Для цитирования: Девянин П.Н., Тележников В.Ю., Хорошилов А.В. Формирование методологии разработки безопасного системного программного обеспечения на примере операционных систем. Труды ИСП РАН, том 33, вып. 5, 2021 г., стр. 25–40. DOI: 10.15514/ISPRAS-2021-33(5)-2.

ISSN PRINT
2879-8156
ТОМ 33
ВЫПУСК 5

ISSN ONLINE
2228-6426
VOLUME 33
ISSUE 5

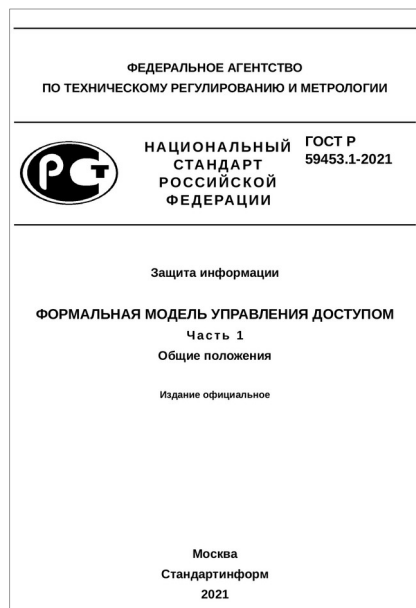
Для разработки и верификации программного кода компонент ОССН в зависимости от степени их важности для обеспечения безопасности (от основы **поверхности атаки** — модуля **PARSEC**) применяются:

- **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-BC);
- Инструментальные средства **динамического анализа** (фаззинга) программного кода (Crusher, AFL, libFuzzer, Syzkaller4Astra);
- Инструментальное средство **сбора трасс и анализа помеченных данных** (Блесна);
- Средства **сбора и аналитической обработки** результатов анализа программного кода в «**Базе данных доверия**» на «**Стенде доверия**»;
- Система **непрерывной разработки и интеграции** GitLab.



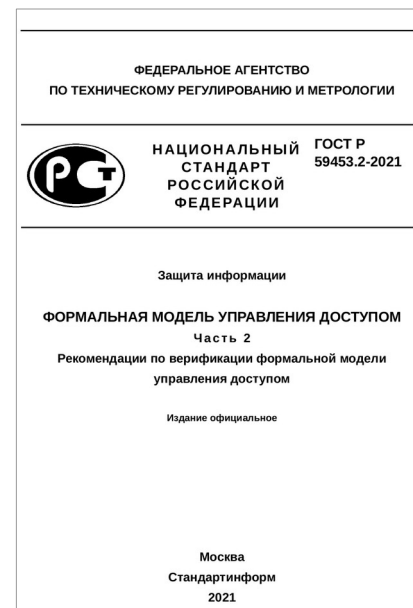
РАЗВИТИЕ НОРМАТИВНОЙ БАЗЫ НА ПРИМЕРЕ ГОСТ Р 59453.1,2-2021 «ЗАЩИТА ИНФОРМАЦИИ. ФОРМАЛЬНАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ»

05



СОДЕРЖАНИЕ

1. Область применения
2. Нормативные ссылки
3. Термины и определения
4. Общие положения
5. Описание состояний в рамках формальной модели управления доступом
6. Описание правил перехода из состояний в состоянии в рамках формальной модели управления доступом
7. Доказательство выполнения условий безопасности



СОДЕРЖАНИЕ

1. Область применения
2. Нормативные ссылки
3. Термины и определения
4. Общие положения
5. Выбор инструментальных средств верификации формальной модели управления доступом
6. Формализованное (машиночитаемое) описание формальной модели управления доступом
7. Верификация формализованного (машиночитаемого) описания формальной модели управления доступом

Приложение А (справочное). Примеры перевода элементов математического описания формальной модели управления доступом в формализованное (машиночитаемое) описание

ПРИМЕРЫ ОПРЕДЕЛЕНИЙ:

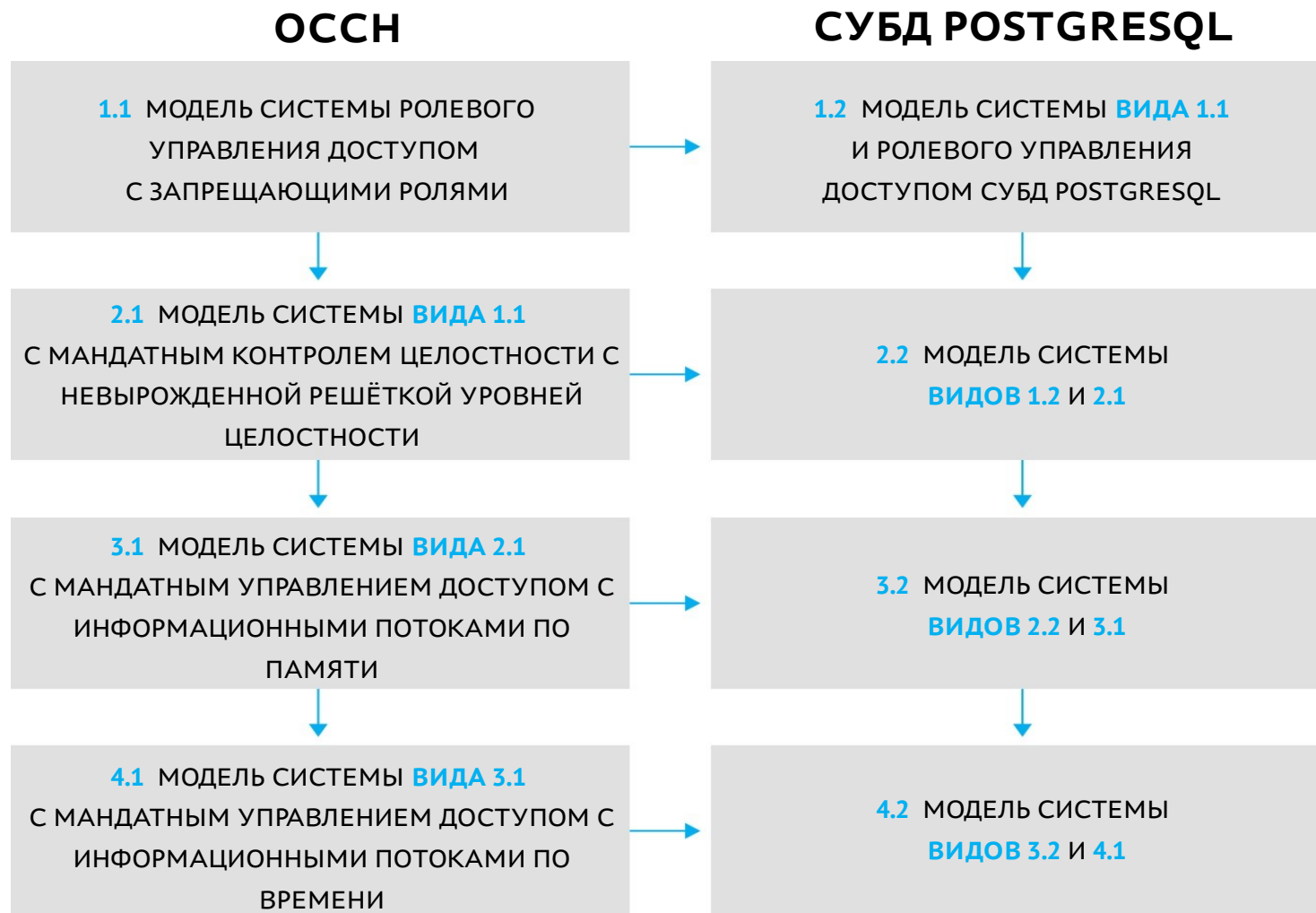
> **формальная модель управления доступом:** Математическое или формализованное (машиночитаемое, пригодное для автоматизированной обработки) описание средства защиты информации и компонентов среды его функционирования, предоставление доступов между которыми регламентируется политиками управления доступом,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

> **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

06



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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \dots \vdash_{opN} G_N$, 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.

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:

$DB_L_{S_N} = DB_L_{S_0}$. Для каждой субъект-сессии $x, y \in S_N$ выполняется, если $y \in DB_L_{S_0} \cap de_facto_own_N(x)$, то $x \in DB_L_{S_0}$.

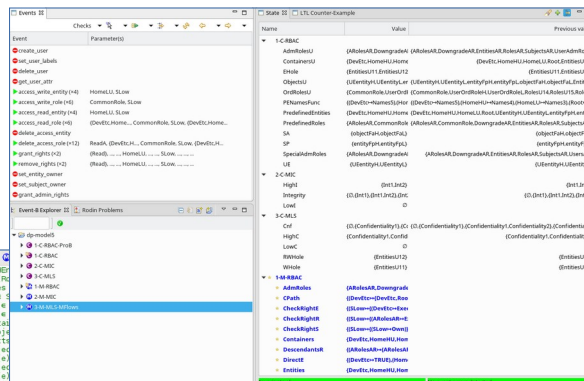
Тогда на этих траекториях система $\Sigma(G^*, OP, G_0)$ безопасна в смысле мандатного контроля целостности.

ВЕРИФИКАЦИЯ МРОСЛ ДП-МОДЕЛИ В ФОРМАЛИЗОВАННОЙ НОТАЦИИ И ВЕРИФИКАЦИЯ ЕЕ РЕАЛИЗАЦИИ В ПРОГРАММНОМ КОДЕ ОССН

07

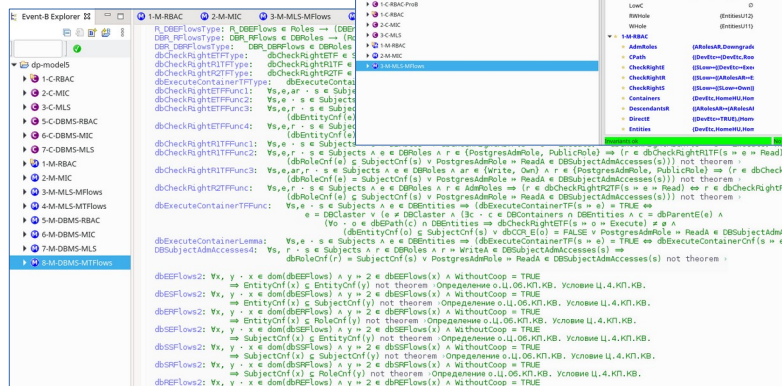
access_read(x, x', y, α _y)		
1.1	x, y	$x \in S$, если $y \in E \cup R \cup NR \cup AR$, то существует $r \in R \cup AR$: $(x, r, read_y) \in AA$, [если $y \in E$, то $(y, read_y) \in PA(r)$ и существует контейнер $c \in S$ такой, что $execute_container(x, c, y) = true$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_y) \in AA$ и $(y, read_y) \in PA(nr)$], [если $y \in R \cup NR \cup AR$, то $(y, read_y) \in APA(r)$], [если $y \in R \cup AR$, то для всех $nr \in constraint_w(y)$ верно $(x, nr, read_y) \in AA$]
1.2	x'	$x' \in S$, [если $y \in R \cup NR \cup AR$, то $i(y) \leq i(x)$, для $e \in y$] либо $(x, e, read_y) \in A$, либо $(x, e, write_y) \in A$, [если $y \in R \cup NR \cup AR$ и $i(y) > i_{low}$, то $(x', i_entity, write_y) \in A$]
2.1	α _y	[если $y \in E \setminus DB_E$, то $α_y = \emptyset$], [если $y \in DB_E$, то или $(x, postgres_admin_role, read_y) \in AA$, или существует $r \in DB_R$: $(x, r, read_y) \in AA$, α _y ∈ DB_PRIVILEGES, read_y ∈ db_rights(α _y), (db_entity(y), read_y) ∈ DB_PA(r), и существует контейнер $c \in C \cup DB_C$ такой, что $execute_container(x, c, y) = true$], [если $y \in DB_R$, то $y \neq public_role$, α _y = ∅ и или [db_login(x) = ∅, существует $r \in AR$: $(x, r, read_y) \in AA$, (y, read_y) ∈ DB_APA(r)], или [(x, postgres_admin_role, read_y) ∈ AA или $y \leq db_login(x)$]]]
2.2	-	если $y \in DB_R$, то [для $e \in y$] либо $(x, e, read_y) \in A$, либо $(x, e, write_y) \in A$, [i(y) ≤ i(x)], и если $y \neq public_role$, то $i(y) = i(db_login(x))$, [если $i(y) > i_{low}$, то $(x', db_i_entity, write_y) \in A$]

МРОСЛ ДП-модель в математической нотации (более 500 страниц описания)



Верификация МРОСЛ ДП-модели в формализованной нотации:

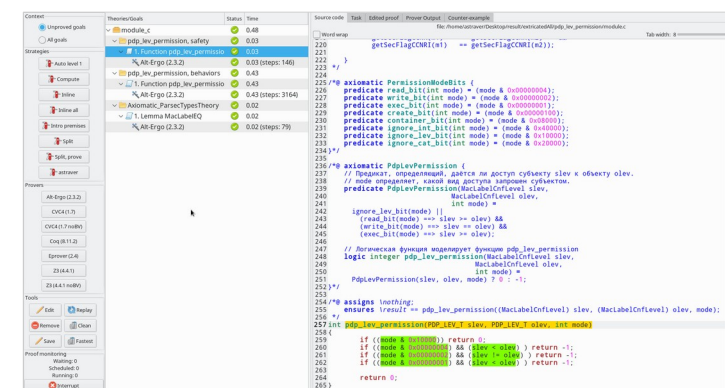
- > дедуктивно инструментом Rodin;
- > по методу проверки моделей (model checking) инструментом ProB.



Дедуктивная верификация спецификаций на языке ACSL функций подсистемы безопасности PARSEC инструментом Frama-C

МРОСЛ ДП-модель в формализованной нотации на языке метода Event-B:

- > базовая (более 30 тыс. строк кода);
- > экспериментальная (редуцированная);
- > адаптированная для системных вызовов управления доступом в ОССН.



СТАТИЧЕСКИЙ АНАЛИЗ КОДА ОССН С ПРИМЕНЕНИЕМ SVACE, CLANG SA, CPPCHECK, АК-ВС

МЕНЕДЖЕР ФАЙЛОВ:

<<< utils-common

CMakeLists.txt

_Makefile

getparsec.c

ls.c

setparsec.c

ASTRA LINUX
ПРОСМОТР ИСХОДНИКОВ

/parsec/parsec-3.0+ci96/utils-common/setparsec.c

318

#ifdef SETFMAC

319

mac_t mac = NULL;

320

cmd_t cmd = cmd_init_setmac();

321

if (!cmd)

322

goto fexit;

323

mac = mac_init(MAC_TYPE_OBJECT);

324

if ((p = strchr(*text_p, '\n'))

325

*p = 0;

Svace static

Cppcheck static

DEREF_AFTER_NULL.EX

Описание: After having been compared to NULL value at setparsec.c:316, pointer 'text_p' is dereferenced at setparsec.c:324.

Средство анализа: Svace 3.2.0

CWE: -----

Bugtracker: BT-1347

Статус: Подтверждено

Параметры анализа: svace warning all true

DEREF_AFTER_NULL.EX

Описание: After having been compared to NULL value at setparsec.c:316, pointer 'text_p' is dereferenced at setparsec.c:324 by calling function 'strchr'.

Средство анализа: Svace 3.2.0

CWE: -----

Bugtracker: BT-1347

Статус: Подтверждено

Параметры анализа: svace warning all true

nullPointerRedundantCheck

Описание: Either the condition 'if(text_p&&*text_p)' is redundant or there is possible null pointer dereference: text_p.

Средство анализа: Cppcheck 1.86

CWE: 476

Bugtracker: BT-1347

Статус: Подтверждено

Параметры анализа: cppcheck --enable=warning,unusedFunction,performance -f --suppress=unknownMacro

СПИСОК ОШИБОК:

COMMENTED_SOURCE_CODE:270

Svace static

COMMENTED_SOURCE_CODE:279

Svace static

COMMENTED_SOURCE_CODE:284

Svace static

DEREF_AFTER_NULL.EX:324

Svace static

DEREF_AFTER_NULL.EX:324

Svace static

nullPointerRedundantCheck

Svace static

ПАРАМЕТРЫ:

Критичность:

Средняя

Статус:

Подтверждено

Bug tracker:

BT - 1347

Применить

Средство

Выбрать [tname+type]

Svace - статический анализатор кода

Cppcheck - статический анализатор кода

Clang - статический анализатор кода

Анализ CVE

АК-ВС

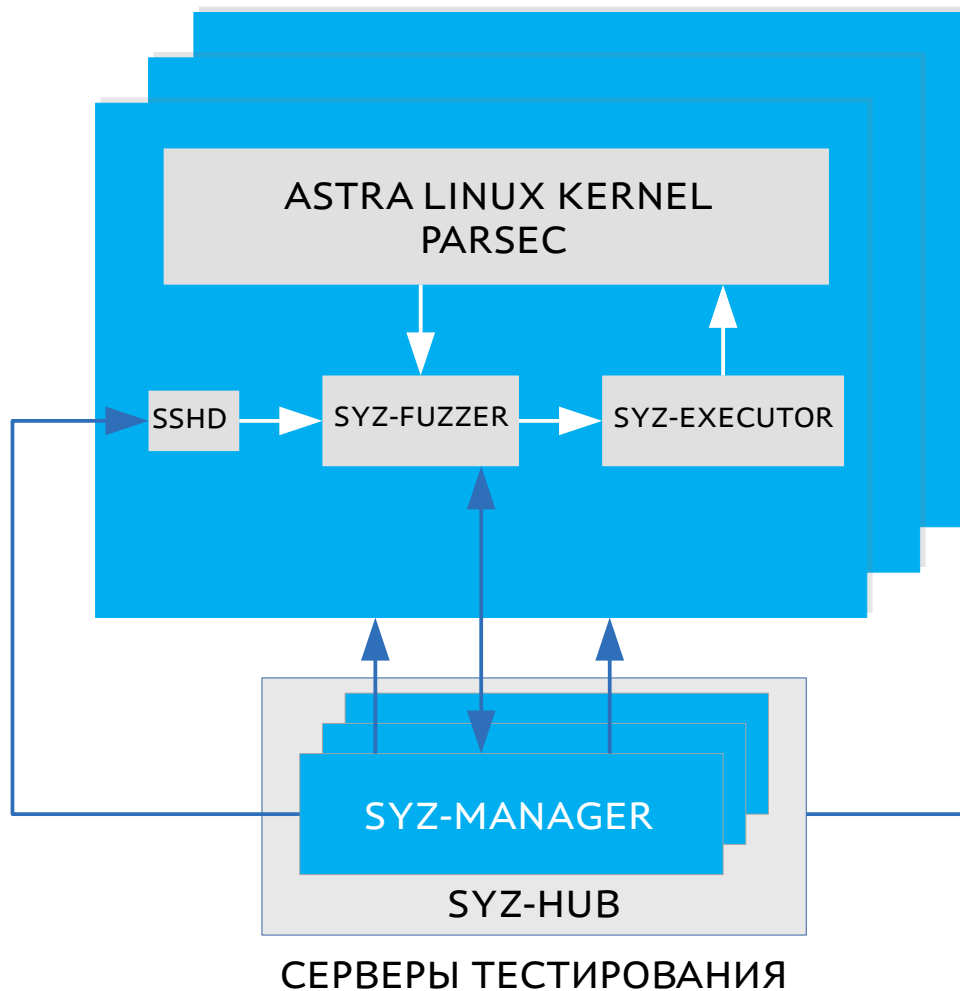
Syzkaller - фаззер уровня ядра ОС

Syzcrawler - среда регрессионного фаззинг-тестирования

ДИНАМИЧЕСКИЙ АНАЛИЗ КОДА ОССН С ИСПОЛЬЗОВАНИЕМ СТЕНДА SYZKALLER4ASTRA

09

ВИРТУАЛЬНЫЕ МАШИНЫ С ОССН



БАЗА ДАННЫХ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ

РАСЧЕТ ПОКРЫТИЯ КОДА

▼ parsec	60%	of 2096	int pdp_lev_permission(PDP_LEV_T slev, PDP_LEV_T olev, int mode)
▼ linux-astar/parsec	61%	of 2071	{
▼ dp	82%	of 169	if ((mode & LEGACY_IGNORE_LEV)) return 0;
pdp_common.c	82%	of 168	if ((mode & R_OK) && (slev < olev)) return -EACCES;
pdp_common.h	100%	of 1	if ((mode & W_OK) && (slev != olev)) return -EACCES;
audit-file.c	37%	of 135	if ((mode & X_OK) && (slev < olev)) return -EACCES;
audit-kernel.c	60%	of 22	return 0;
cap.c	80%	of 74	}
cmdline.c	---	of 46	int pdp_cat_permission(PDP_CAT_T scat, PDP_CAT_T ocat, int mode)
crc16.c	---	of 5	{
logfs.c	---	of 27	if(mode & LEGACY_IGNORE_CAT) return 0;
net.c	48%	of 151	if((mode&R_OK) && ((scat & ocat) != ocat)) return -EACCES;
parsec-fs.c	85%	of 91	if((mode&W_OK) && (ocat != scat)) return -EACCES;
path.c	50%	of 6	if((mode&X_OK) && ((scat & ocat) != ocat)) return -EACCES;
sec-audit.c	60%	of 220	return 0;
sec-audit.h	100%	of 1	}
sec-hooks.c	51%	of 515	int pdpml_conf_permission(const PDPML_I *s, const PDPML_I *o, int mode)
security.c	72%	of 150	{
security.h	100%	of 1	if(pdp_lev_permission(s->lev,o->lev,mode) pdp_cat_permission(s->cat,o->cat,mode)) {
syscalls.c	88%	of 217	return -EACCES;
userconfine.c	43%	of 78	return 0;
xattr.c	73%	of 163	}
parsec_elfrand.c	---	of 5	
parsec_gost89.c	---	of 20	
safesetid	---	of 78	

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

10









LCOV - code coverage report

Current view: **top level**

Test: **cov.info**

Date: **2021-11-25 13:15:09**

	Hit	Total	Coverage
Lines:	2700	3101	87.1 %
Functions:	226	234	96.6 %

Directory	Line Coverage ↕			Functions ↕	
/usr/include/x86_64-linux-gnu/bits		76.5 %	26 / 34	-	0 / 0
/usr/include		100.0 %	2 / 2	100.0 %	1 / 1
/usr/include/x86_64-linux-gnu/sys		100.0 %	1 / 1	-	0 / 0
lib-aud		88.6 %	1170 / 1320	97.9 %	94 / 96
lib-aud-db-files		90.9 %	189 / 208	100.0 %	11 / 11
lib-aux		92.5 %	544 / 588	100.0 %	42 / 42
lib-base		81.0 %	235 / 290	100.0 %	24 / 24
lib-cap		80.9 %	511 / 632	93.1 %	54 / 58

70% – покрытие модулей безопасности PARSEC (поверхности атаки) 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)

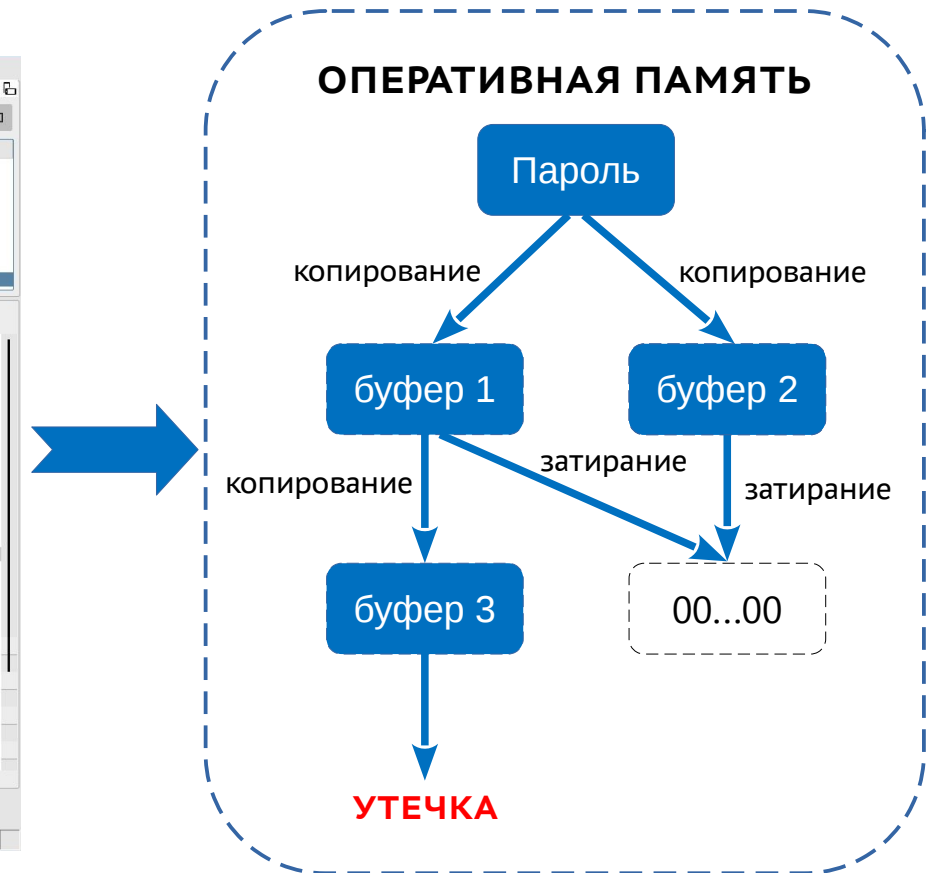
11

Инструмент БЛЕСНА

The screenshot displays the BLESNA (БЛЕСНА) tool interface. The top section shows a call stack (Стек вызовов) with columns for Call (Вызов), Return (Возврат), Function (Функция), Address (Адрес), Module (Модуль), and Offset (Смещение). The bottom section shows a memory dump (Поиск утечек) with columns for Address (Адрес), Disassembly (Дисассемблирование), and Comment (Комментарий). The memory dump shows a sequence of instructions and data, including a jump instruction at address 19FC71C1.

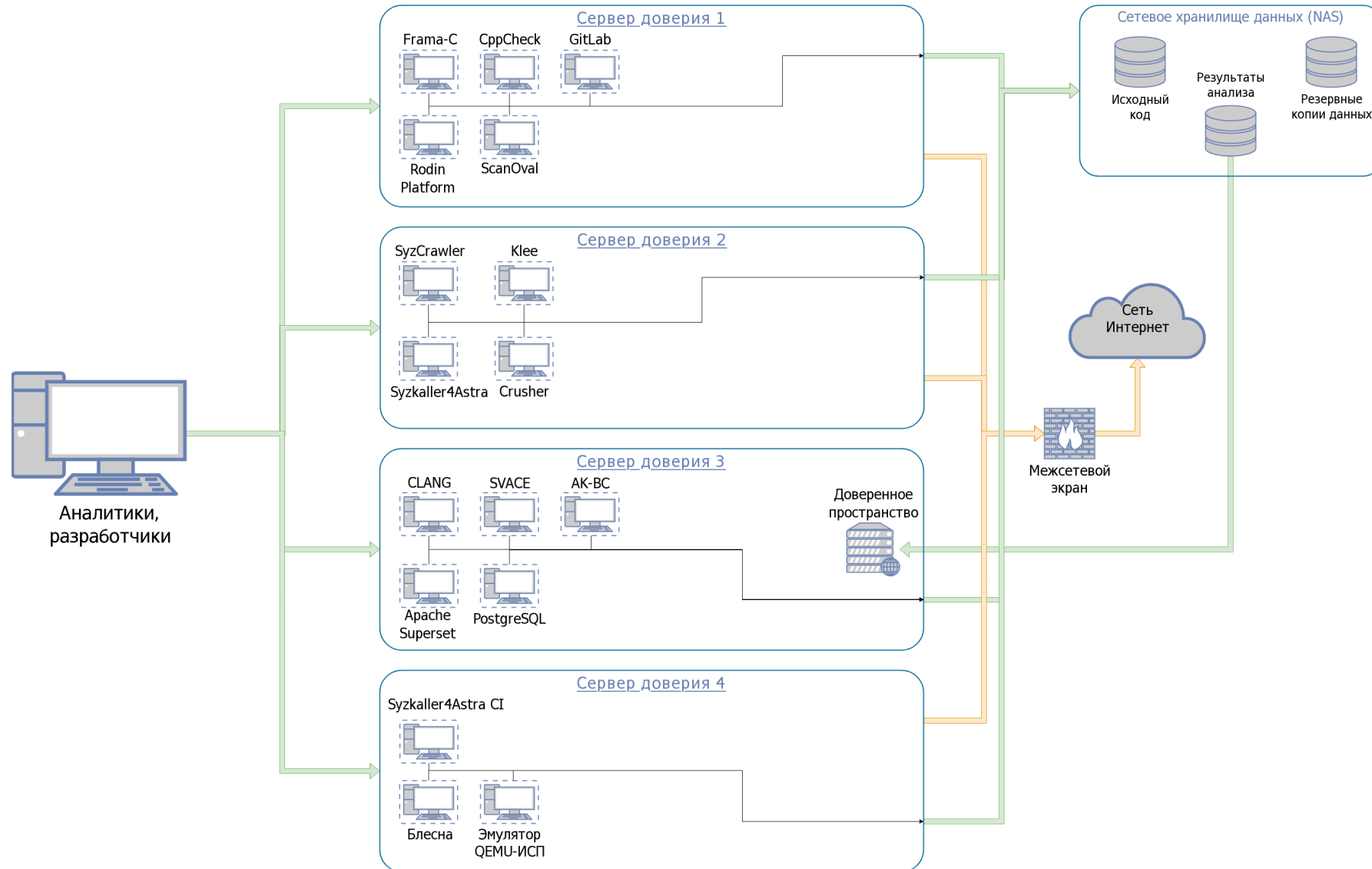
Вызов	Возврат	Функция	Адрес	Модуль	Смещение
007613E9	?	неизвестная	00007FFF7DE...		
00764009	?	неизвестная	000000000040...		
0170598C	6340A150	stub_403700	000000000040...		
0170598D	6340A150	pam_authenti...	00007FFF7FB...	libpam.so.0...	35F0h
017067A8	63409FEC	func_3b20	00007FFF7FB...	libpam.so.0...	3820h
01999008	6342A331	pam_sm_auth...	00007FFF7C6...	pam_unix.so	4050h
019FCA39	01BC71CE	stub_3630	00007FFF7C6...	pam_unix.so	3630h
019FCA31	01BC71CE	pam_get_auth...	00007FFF7FB...	libpam.so.0...	5610h
019FCA33	01BC71CE	jmp_5040	00007FFF7FB...	libpam.so.0...	5040h

Адрес	Дисассемблирование	Комментарий
19FC5E0	func_7ffff7e48520	
19FC69E	stub_3290	
19FC69F	jmp_7ffff7e68020	
19FC6C2	stub_3040	
19FC6C3	func_jmp_7ffff7e4a9a0	
19FC73B	stub_3610	
19FC73C	func_jmp_7ffff7e4dda0	
19FC741	stub_7ffff7de8130	
19FC742	jmp_7ffff7e5e1e0	
19FC789	stub_7ffff7de8308	
19FC78A	func_jmp_7ffff7e4a350	
19FC9CF	stub_3040	
19FC9D0	func_jmp_7ffff7e4a9a0	
19FCA31	stub_3630	
19FCA32	pam_get_auth...	
19FCA34	jmp_5040	
19FCA4F	stub_30c0	
19FCA50	jmp_30c6	
19FCA63	func_7ffff7fe3a30	
19FCA96	func_7ffff7df240	
19FCB33	func_7ffff7de750	
19FCBDE	func_7ffff7de5b0	
19FCC0A	func_7ffff7febf10	
19FCCB8	func_4f90	
19FCCB8	stub_3110	
19FCCCC	jmp_7ffff7e5e1e0	
19FCCCE	stub_3090	
19FCCCE	jmp_7ffff7e56ce0	
19FCD11	stub_3090	
19FCD12	jmp_7ffff7e56ce0	
19FCD4A	stub_3100	



МАСШТАБИРУЕМАЯ СТРУКТУРА СТЕНДА ДОВЕРИЯ ДЛЯ ВЕРИФИКАЦИИ И АНАЛИЗА КОДА ОССН

12



ИНТЕРФЕЙС БАЗЫ ДАННЫХ ДОВЕРИЯ. ВЫВОД ИНФОРМАЦИИ О ОШИБКАХ

МЕНЕДЖЕР ФАЙЛОВ:

<<< utils-pdp

CMakeLists.txt

ls_pdp.c

ls_proc.c

ls_xstrtol.c

pdp-id.c

pdp-init-fs

pdp-init-fs.c

pdp-ls.c

pdp-file.c

pdp-ps.c

pdp-user.c

set-fs-ilev.c

setfattr.c

unset-fs-ilev

ASTRA LINUX

ПРОСМОТР ИСХОДНИКОВ

/parsec/parsec-3.0+ci65/utlis-pdp/pdp-ls.c

2446

size_t i;

2447

2448

for (i = 0; i < files_index; i++) {

2449

free(files[i].name);

2450

free(files[i].linkname);

DANGLING_POINTER.STRICT

Описание: Several pointers freed in function 'clear_files' are not overwritten. For example, pointer 'files[i]->linkname

Средство анализа: Svace 3.2.0

CWE: -----

Bugtracker: BT-22564

Статус: Исправлено

Параметры анализа: svace warning all true

SEGV on unknown address

Описание: SEGV on unknown address

Средство анализа: AFL 2.68c

CWE: -----

Bugtracker: BT-24261

Статус: Исправлено

Параметры анализа: -----

СПИСОК ОШИБОК:

core.UndefinedBinaryOperatorResult:2437

Clang

static

DANGLING_POINTER.STRICT:2450

Svace

static

SEGV on unknown address:2450

AFL

dynamic

allocaCalled:2524

Cppcheck

static

PROC_USE.VULNERABLE:2760

Svace

static

security.insecureAPI.strcpy:2760

Clang

static

ПАРАМЕТРЫ:

Критичность:

Средняя

Фильтры

Таблица

Перекрестный запрос

Дата

5

ПЕРИОД ВРЕМЕНИ

No filter

Количество результатов запроса

6

17

Дистрибутив

5

ДИСТРИБУТИВ

Astra Linux SE 1.7 (update0)

Название пакета

5

linux-astra-modules

Версия пакета

5

5.4.0-34.astra33+ci155

Статус

5

СТАТУС

Подтверждено

Пользователь

6

ПОЛЬЗОВАТЕЛЬ

Выбрать [Пользователь]

Баг-трекер

6

BUG_TRACK

Выбрать [bug_track]

Только зарегистрированные

Только незарегистрированные

Средство

5

TNAME+TYPE

Syzkaller - фаззер уровня ядра ОС

Svace - статический анализатор кода

Критичность

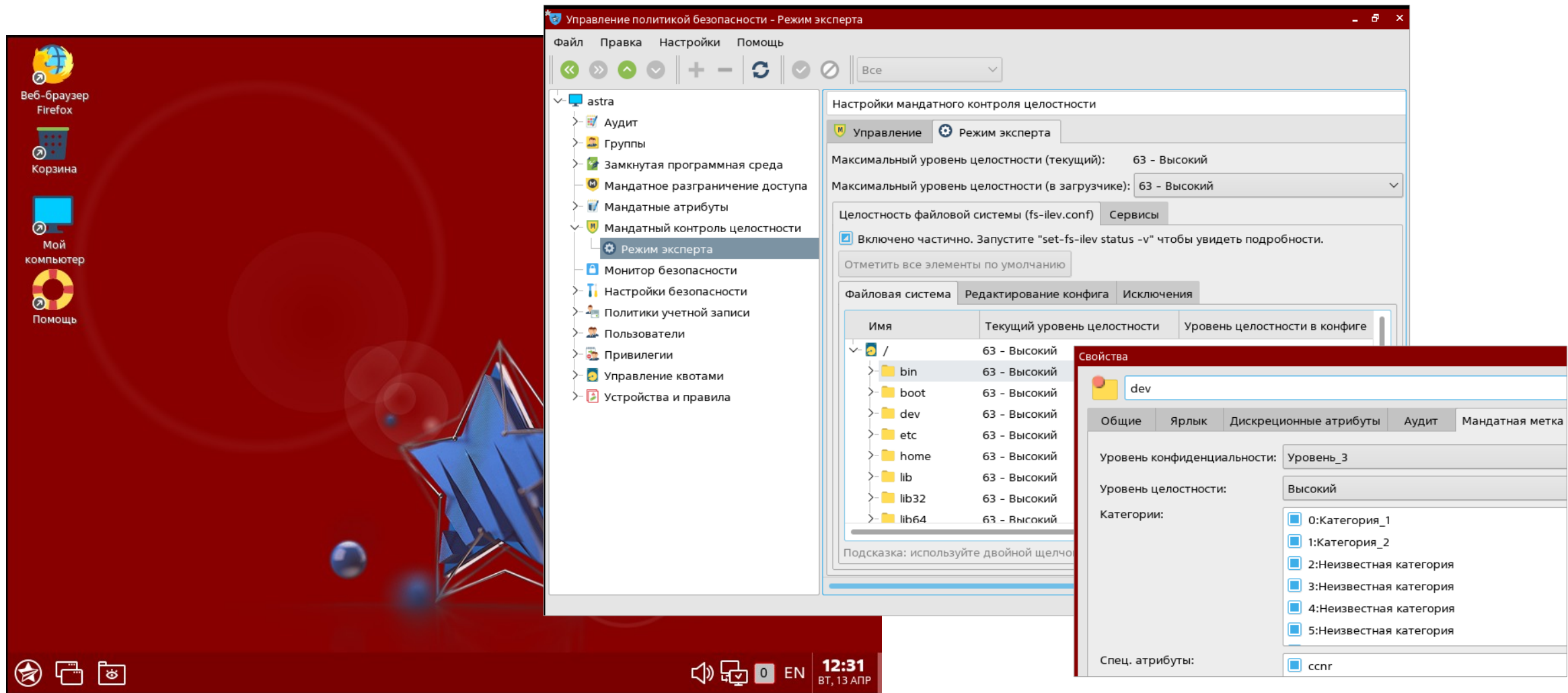
6

AUTOSEVERITY

Выбрать [autoseverity]

РЕАЛИЗАЦИЯ С ПРИМЕНЕНИЕМ МЕТОДОЛОГИИ ВЕРИФИЦИРОВАННОГО МЕХАНИЗМА ЗАЩИТЫ ОСН АСТРА LINUX SPECIAL EDITION

14



Спасибо за внимание!