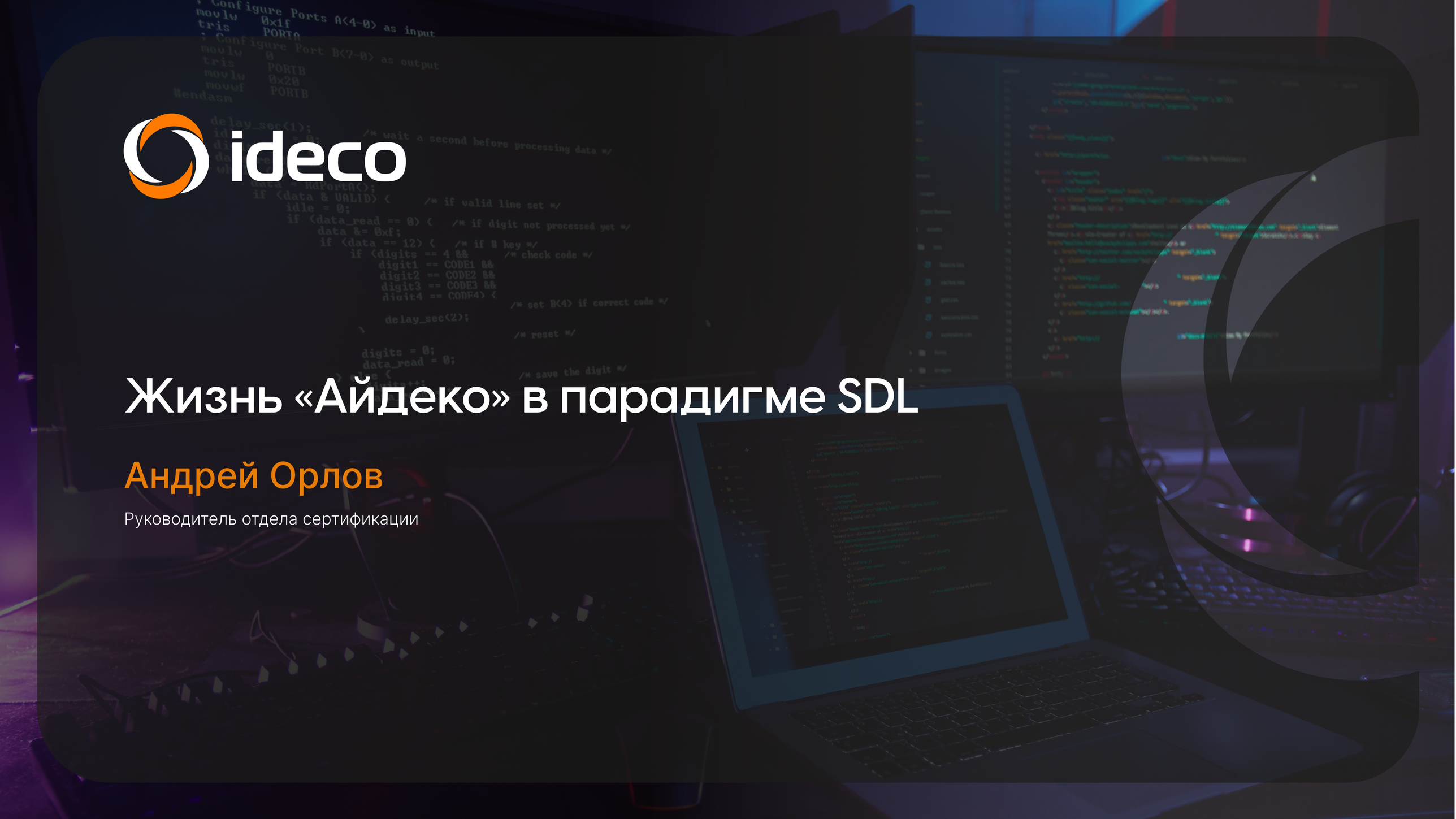


ideco

Жизнь «Айдеко» в парадигме SDL

Андрей Орлов

Руководитель отдела сертификации

О компании

Помогаем компаниям защитить сеть от современных угроз безопасности удобным и «умным» межсетевым экраном нового поколения Ideco UTM. Экономим ваше время на настройке интернет-шлюза.

Используем собственные ноу-хау и базы данных, которые создаются с помощью технологий машинного обучения и искусственного интеллекта.

Более 4 000 компаний защищает Ideco UTM

Наше решение блокирует 25 000 атак ежедневно

17 лет и 400 000 часов разработки

123 человека в команде Sales / R&D

2023 Ideco UTM 16 фильтрация трафика до 40 Гбит/сек.

2022 Ideco UTM 14 — новая система отчётов, BGP, OSPF, LACP

2021 Получение сертификата ФСТЭК № 4503 на МЭ, СОВ и УД

2018 Получение лицензии ФСТЭК на разработку СЗКИ

2016 Ideco UTM вошёл в «ЕРРП для ЭВМ и БД»

2015 Ideco ICS 6 получил сертификат ФСТЭК МЭ3/НДВ4

2013 Конкурс ITStarz собрал более 100 000 участников

2010 Ideco ICS 4.0 — продукт года по версии Softool

2008 Ideco ICS получил премию журнала PC Magazine/RE

2005 Основание компании Айдеко

Вклад в безопасную разработку

Вошли в экспертную группу по проработке вопросов функционирования Технологического центра.

03 февраля 2023

	Назначено	В работе	Подтверждено					Won't Fixed				False Positive			
			В работе	Сообщено	Исправлено	в 5.10	Всего	Без вериф.	Обсуждается	Подтверждено	Всего	Без вериф.	Обсуждается	Подтверждено	Всего
01-bellsoft	240	40	46	1	12	3	62	16	-	2	18	36	9	75	120
02-basealt	240	31	72	2	-	1	75	60	6	43	109	11	3	11	25
03-astralinux	240	9	33	5	-	5	43	25	-	76	101	28	3	56	87
04-rosa	240	3	28	6	1	-	35	61	2	40	103	43	6	50	99
05-ivk	240	1	19	1	2	-	22	77	1	33	111	50	-	56	106
06-redsoft	240	14	62	1	3	2	68	28	-	36	64	22	3	69	94
07-yandex	240	10	17	4	6	-	27	53	-	47	100	32	1	70	103
08-aladdin	240	7	34	-	6	1	41	48	-	34	82	38	2	70	110
09-mcst	240	-	66	-	1	-	67	55	1	72	128	13	1	31	45
10-omp	240	23	20	-	23	3	46	67	-	15	82	45	8	36	89
12-securitycode	240	-	93	-	2	-	95	34	1	20	55	30	1	59	90
13-infotecs	240	2	23	4	1	1	29	42	-	41	83	60	-	66	126
14-swemel	200	1	7	9	3	-	19	78	3	53	134	5	2	39	46
15-fintech	200	16	41	1	1	4	47	46	-	19	65	29	6	37	72
16-factor-ts	195	2	9	10	3	1	23	62	1	48	111	27	-	32	59
17-confident	160	131	3	-	-	-	3	1	-	6	7	2	4	13	19
18-rasu	150	17	18	-	1	2	21	25	2	68	95	4	2	11	17
19-itb	140	84	19	-	-	-	19	7	2	4	13	18	-	6	24
20-ideco	155	92	14	-	-	-	14	10	4	7	21	13	3	12	28
21-nppct	20	-	8	-	-	-	8	5	1	-	6	2	-	4	6
Всего:	**4100**	483	632	44	65	23	764	800	24	664	1488	508	54	803	1365

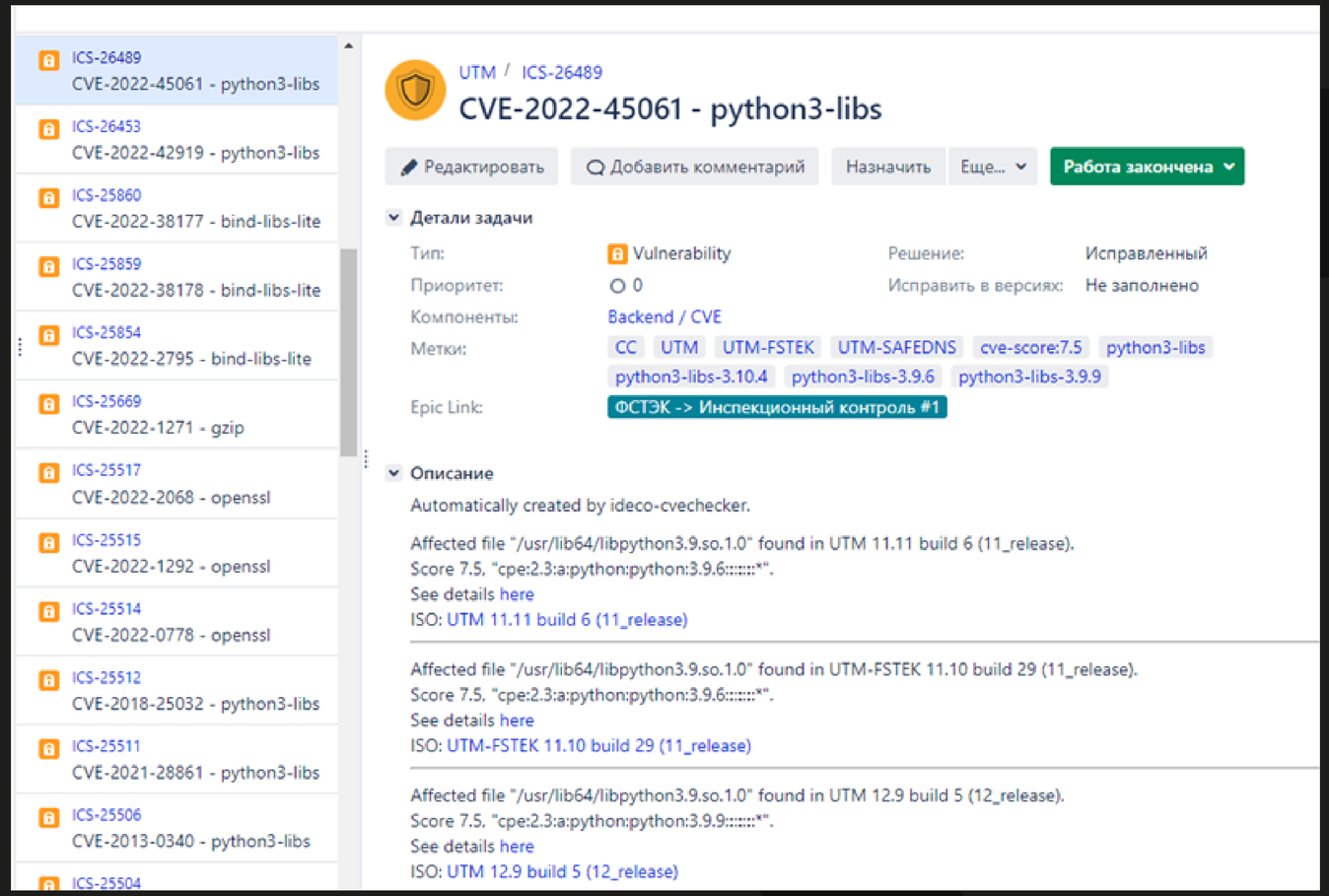
Security Development Lifecycle

Автоматизировано:

1. Пересборка всего.
2. Поиск CVE и создание задач в JIRA.
3. Unit, интеграционные и нагрузочные тесты.
4. Запуск линтеров
5. Статический анализ (SVACE).
6. Фаззинг (AFL++, SYZKALLER, GO-FUZZ)

Вручную:

Анализ прав доступа, поиск подозрительных лексем, binary-security-check, gixy, metasploit, nikto, nmap, OWASP ZAP, Burpsuite, ssh_scan, сбор трафика Wireshark'ом, ossaudit, cppchecker и др.

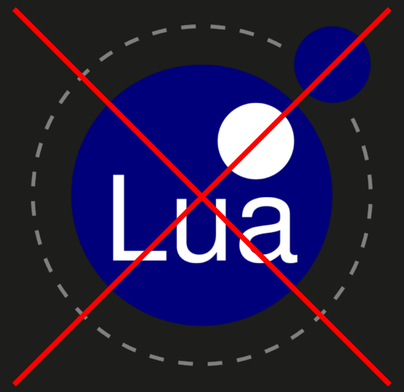


Security Development Lifecycle

ideco

Что еще:

1. Уменьшение компонентной базы.
2. Поиск НДВ.
3. Обновление ПО.
4. Борьба с supply chain attack.
5. Code review.
6. Architecture review.
7. Runtime ограничение для всех процессов.
8. Анализ с помощью Natch.



Яндекс Облако

Немного цифр

ideco

Статический анализ

Найдено - **237**

В апстрим отправлено - **7**

CVE-уязвимости

Найдено и исправлено - **184**

Фаззинг

Найдено - **34**

В апстрим отправлено - **11**

Объём

Исходников - **11,6 ГБ**

Файлов - **533 тыс.**

Кейс №1

ideco

DOUBLE_FREE в syslog-ng исправлено в апстрим

```
432  static void
433  _relocate_qfile(PersistState *state, const gchar *name)
434  {
435    if (_is_persist_entry_holds_diskq_file(state, name))
436      {
437        gchar *qfile = persist_state_lookup_string(state, name, NULL, NULL);
438        printf("found qfile, key: %s, path: %s\n", name, qfile);
439        gchar *base = g_path_get_basename(qfile);
440        gchar *relocated_qfile = g_build_filename(new_diskq_path, base, NULL);
441        if (!relocated_qfile)
442          {
443            fprintf(stderr, "Invalid path. new_diskq_dir: %s, qfile: %s\n", new_diskq_path, qfile);
                TRACE 2.1  Variable 'qfile' is passed to function 'g_free' as 1st parameter
444            g_free(qfile);
445          }
446
447        if (_move_file(qfile, relocated_qfile))
448          {
449            printf("new qfile_path: %s\n", relocated_qfile);
450            persist_state_alloc_string(state, name, relocated_qfile, -1);
451          }
452        else
453          {
454            fprintf(stderr, "Failed to move file to new qfile_path: %s\n", relocated_qfile);
455          }
456        g_free(base);
           Undecided  Unspecified  Undecided  DOUBLE_FREE.EX Pointer 'qfile' is passed to a function free at dqtool.c:457 by calling function 'g_free' after the referenced memory was deallocated at dqtool.c:444 by calling function 'g_free'.
           TRACE 1.1  Variable 'qfile' is passed to function 'g_free' as 1st parameter
457        g_free(qfile);
458        g_free(relocated_qfile);
459      }
460  }
```

Кейс №2

ideco

AddressSanitizer: SEGV в netmap-ipfw не исправлено в апстрим

```
2288
2289    CTL3_LOCK();
2290    count = ctl3_hsize;
2291    size = count * sizeof(ipfw_sopt_info) + sizeof(ipfw_obj_lheader);
2292
2293    /* Fill in header regadless of buffer size */
2294    olh->count = count;
2295    olh->objsize = sizeof(ipfw_sopt_info);
2296
2297    if (size > olh->size) {
2298        olh->size = size;
2299        CTL3_UNLOCK();
2300        return (ENOMEM);
2301    }
2302    olh->size = size;
2303
2304    for (n = 1; n <= count; n++) {
2305        i = (ipfw_sopt_info *)ipfw_get_sopt_space(sd, sizeof(*i));
2306        KASSERT(i != 0, ("previously checked buffer is not enough"));
2307        sh = &ctl3_handlers[n];
2308        i->opcode = sh->opcode;
2309        i->version = sh->version;
2310        i->refcnt = sh->refcnt;
2311    }
2312    CTL3_UNLOCK();
```

```
(lldb) frame select 5
frame #5: 0x00000000003960f9 fuzz_ipfw_ctl`dump_soptcodes(chain=0x00000000011b5900, op3=0x0000617000000400, sd=0x00007fffffffdf
(lldb) bt
* thread #1, name = 'fuzz_ipfw_ctl', stop reason = Heap buffer overflow
    frame #0: 0x0000000000315360 fuzz_ipfw_ctl`__asan::AsanDie()
    frame #1: 0x000000000032cc7a fuzz_ipfw_ctl`__sanitizer::Die() + 42
    frame #2: 0x00000000000314ef4 fuzz_ipfw_ctl`.annobin__ZN6__asan19ScopedInErrorReportD2Ev.start + 420
    frame #3: 0x00000000003149a7 fuzz_ipfw_ctl`__asan::ReportGenericError(unsigned long, unsigned long, unsigned long, unsigned
    frame #4: 0x00000000003154f8 fuzz_ipfw_ctl`__asan_report_load2 + 40
  * frame #5: 0x00000000003960f9 fuzz_ipfw_ctl`dump_soptcodes(chain=0x00000000011b5900, op3=0x0000617000000400, sd=0x00007ffff
    frame #6: 0x00000000003b1931 fuzz_ipfw_ctl`ipfw_ctl3(sopt=0x00007fffffffe420) at ip_fw_sockopt.c:2704:10
    frame #7: 0x00000000003479fd fuzz_ipfw_ctl`main at fuzz_ipfw_ctl.c:85:5
    frame #8: 0x00007ffff7bcb1e2 libc.so.6`__libc_start_main + 242
    frame #9: 0x000000000026a7de fuzz_ipfw_ctl`_start + 46
(lldb) frame variable
(ip_fw_chain *) chain = 0x00000000011b5900
(ip_fw3_opheader *) op3 = 0x0000617000000400
(sockopt_data *) sd = 0x00007fffffffdfa0
(_ipfw_obj_lheader *) olh = 0x0000617000000400
(ipfw_sopt_info *) i = 0x00006170000005c8
(ipfw_sopt_handler *) sh = 0x0000617000000320
(uint32_t) count = 28
(uint32_t) n = 28
(uint32_t) size = 472
(lldb) print ctl3_handlers[27]
(ipfw_sopt_handler) $1 = {
  opcode = 116
  version = '\0'
  dir = '\x02'
  handler = 0x0000000000395600 (fuzz_ipfw_ctl`dump_soptcodes at ip_fw_sockopt.c:2277)
  refcnt = 1
}
(lldb) print ctl3_handlers[28]
(ipfw_sopt_handler) $2 = {
  opcode = 0
  version = '\0'
  dir = '\0'
  handler = 0x0000000000000000
  refcnt = 0
}
(lldb)
```

Кейс №3

ideco

AddressSanitizer: heap-buffer-overflow в Squid не исправлено в апстрим

```
void
Http::One::RequestParser::skipGarbageLines()
{
    if (Config.onoff.relaxed_header_parser) {
        if (Config.onoff.relaxed_header_parser < 0 && (buf_[0] == '\r' || buf_[0] == '\n'))
            debugs(74, DBG_IMPORTANT, "WARNING: Invalid HTTP Request: " <<
                   "CRLF bytes received ahead of request-line. " <<
                   "Ignored due to relaxed_header_parser.");
        // Be tolerant of prefix empty lines
        // ie any series of either \n or \r\n with no other characters and no repeated \r
        while (!buf_.isEmpty() && (buf_[0] == '\n' || (buf_[0] == '\r' && buf_[1] == '\n'))) {
            buf_.consume(1);
        }
    }
}
```

```
@@ -45,8 +45,18 @@ Http::One::RequestParser::skipGarbageLin
                   "Ignored due to relaxed_header_parser.");
         // Be tolerant of prefix empty lines
         // ie any series of either \n or \r\n with no other characters and no repeated \r
-        while (!buf_.isEmpty() && (buf_[0] == '\n' || (buf_[0] == '\r' && buf_[1] == '\n'))) {
-            buf_.consume(1);
+        while (!buf_.isEmpty()) {
+            if (buf_[0] == '\n') {
+                buf_.consume(1);
+                continue;
+            }
+
+            if (buf_.length() >= 2 && buf_[0] == '\r' && buf_[1] == '\n') {
+                buf_.consume(2);
+                continue;
+            }
+
+            break;
```

Выводы

- Регулярное обновление используемых open source библиотек
- Регулярная проверка библиотек по открытым базам данных (bdu.fstec.ru, cve.mitre.org)
- Регулярная проверка кода статическими анализаторами
- Статический анализ не может найти все проблемы в коде
- Для выполнения всех проверок достаточно 2-3 специалистов

ideco

БЛАГОДАРЮ ЗА ВНИМАНИЕ

Андрей Орлов

Руководитель отдела сертификации

@Andrew0r

a.orlov@ideco.ru