

ГОСТ Р 56939-2024.  
Динамический анализ кода  
(на примере АК-ВС 3)

# ТРЕБОВАНИЯ ФСТЭК РОССИИ

## Приказ ФСТЭК России от 2 июня 2020 г. N 76

- п. 19 «Испытания по выявлению уязвимостей и недеklarированных возможностей средства»

## Методика ВУ и НДВ

- п. 4.2 «Статический анализ объекта оценки (САО)»
- п. 4.3.2 «Фаззинг-тестирование объекта оценки (ДАО.2)»
- п. 4.3.3 «Системное тестирование объекта оценки (ДАО.3)»

## ГОСТ Р 56939-2024 «Разработка безопасного ПО»

- п. 5.10 «Статический анализ исходного кода»
- п. 5.11 «Динамический анализ исходного кода»

# Общая информация

- **Динамический анализ кода программы:** Вид работ по инструментальному исследованию программы, основанный на анализе кода программы в режиме непосредственного исполнения (функционирования) кода.
- **Фаззинг-тестирование программы:** Вид работ по исследованию программы, основанный на передаче программе случайных или специально сформированных входных данных, отличных от данных, предусмотренных алгоритмом работы программы.

## Требования к реализации

- 5.11.2.1 Разработать регламент проведения динамического анализа кода ПО
- 5.11.2.2 Определить инструменты динамического анализа и фаззинг-тестирования, порядок их применения
- 5.11.2.3 Определить перечень модулей (компонентов) ПО, которые необходимо подвергнуть динамическому анализу, включая фаззинг-тестирование

## Требования к реализации

- 5.11.2.4 Определить сценарии проведения тестирования для каждого исследуемого модуля (компонента) ПО средствами динамического анализа, включая инструменты проведения фаззинг-тестирования
- 5.11.2.5 Проводить динамический анализ с использованием инструментов динамического анализа
- 5.11.2.7 Проводить фаззинг-тестирование

## Требования к реализации

- 5.11.2.6 Проводить повторный динамический анализ модулей (компонентов) ПО с целью контроля устранения ошибок
- 5.11.2.8 При проведении фаззинг-тестирования использовать тестовые коллекции входных данных, подлежащие дальнейшим мутациям, для каждого из подвергаемых фаззинг-тестированию модуля
- 5.11.2.9 Устранять выявленные в процессе динамического анализа, включая фаззинг-тестирование, ошибки в соответствии с принятыми процедурами устранения найденных средствами динамического анализа ошибок

# Что используем мы

«**AK-BC 3**» — это инструмент, который позволяет автоматизировать процесс:

- проведения **испытаний при сертификации** по требованиям **различных регуляторов**
- **автоматизация процесса внедрения мер по разработке безопасного ПО**



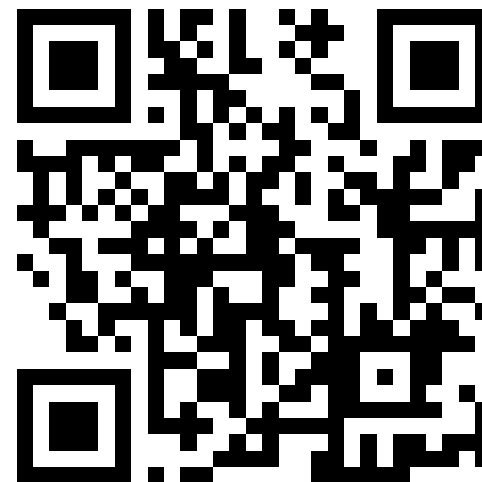
## Материалы по безопасной разработке



Современные методы динамического анализа кода. Фаззинг-тестирование и его классификация



Анализатор кода вычислительных систем АК-ВС 3. Комплексное решение для обеспечения безопасности кода в современных разработках



Как работают инструменты статического и динамического анализа кода. Опыт использования АК-ВС 3



## Пример тестового сценария динамического анализа

- Сценарий – ввод данных (IP подключения) для подключения к ftp серверу

Управление

Режим

☒ ОПД ? ☐ Запись ассемблерной трассы ?

Добавление меток данных

☒ Поиск строки в оперативной памяти (stringsearch)  
☐ Пометить данные, читаемые из файла (file\_taint)  
☐ Пометить данные, читаемые из последовательного порта (serial\_taint)  
☐ Установка меток на входные сетевые данные (tainted\_net) ?

Строки для поиска (в кавычках или ASCII). [Подробнее о формате](#)

"simple string"  
41:53:43:49:49:20:73:74:72

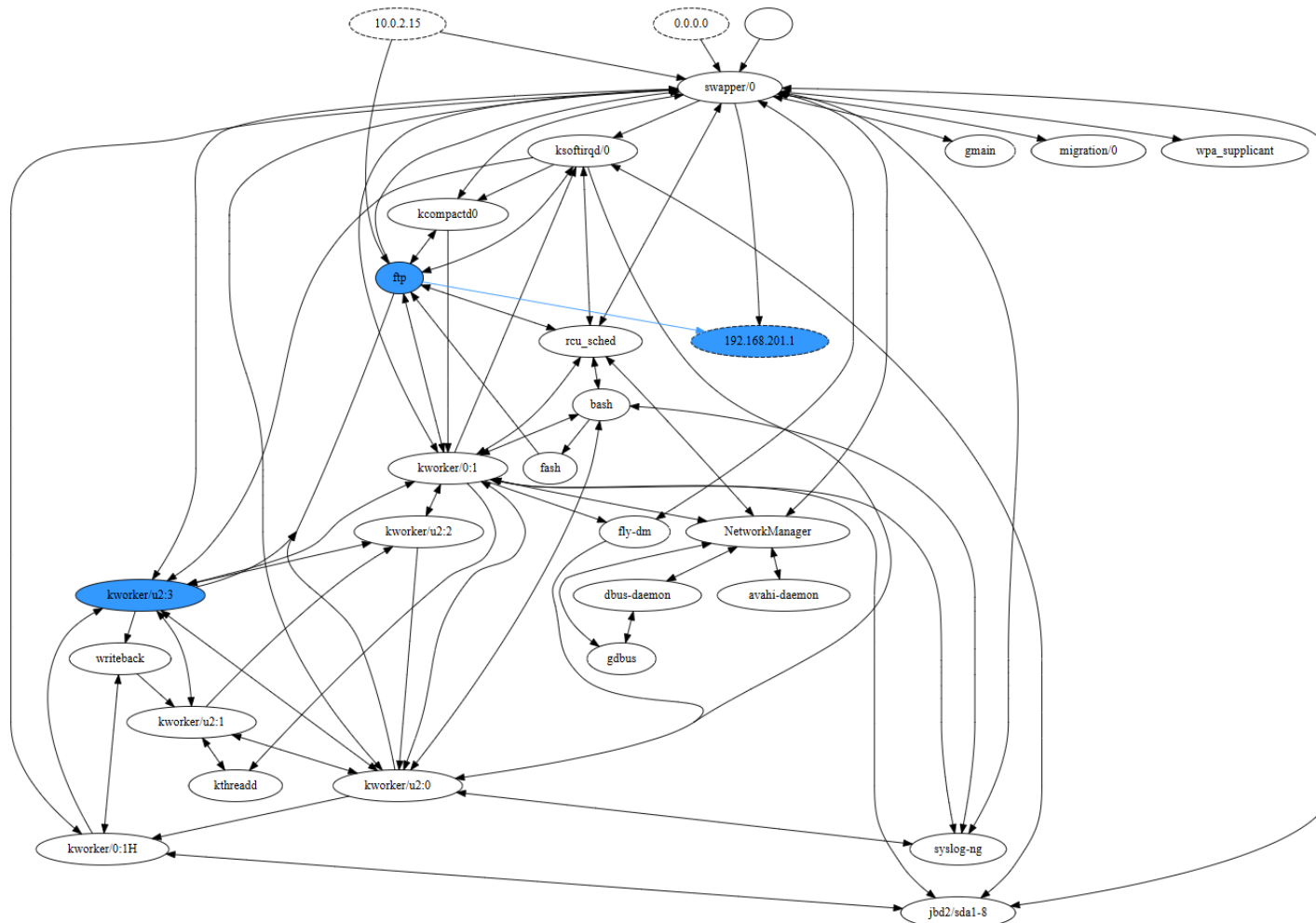
Размер информации о стеке вызова для печати в лог файл ?

16

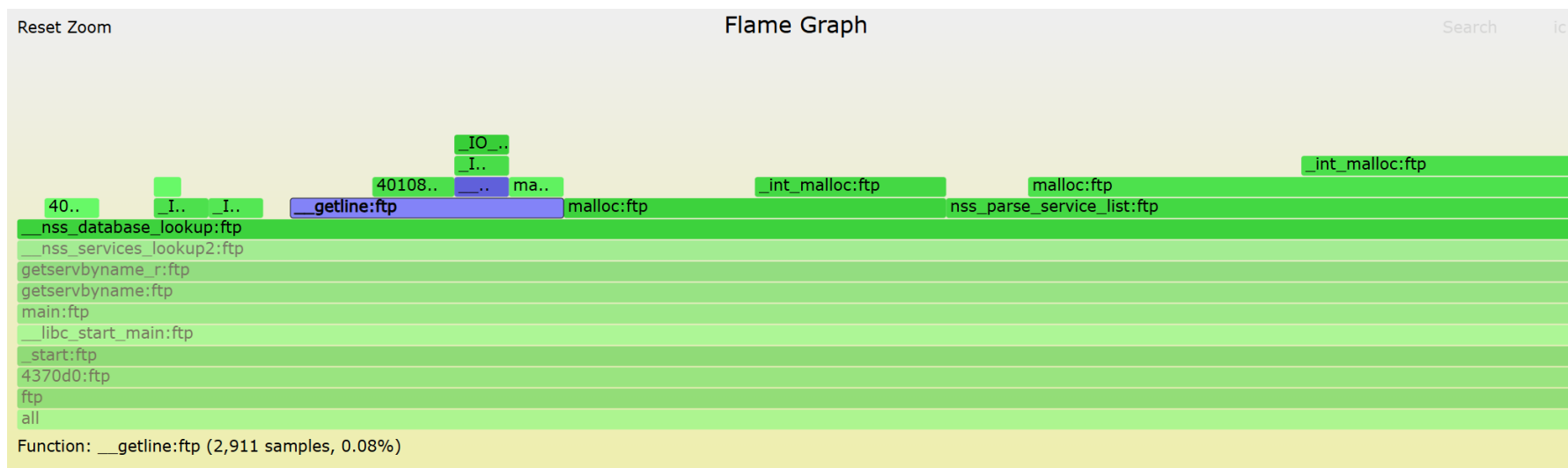
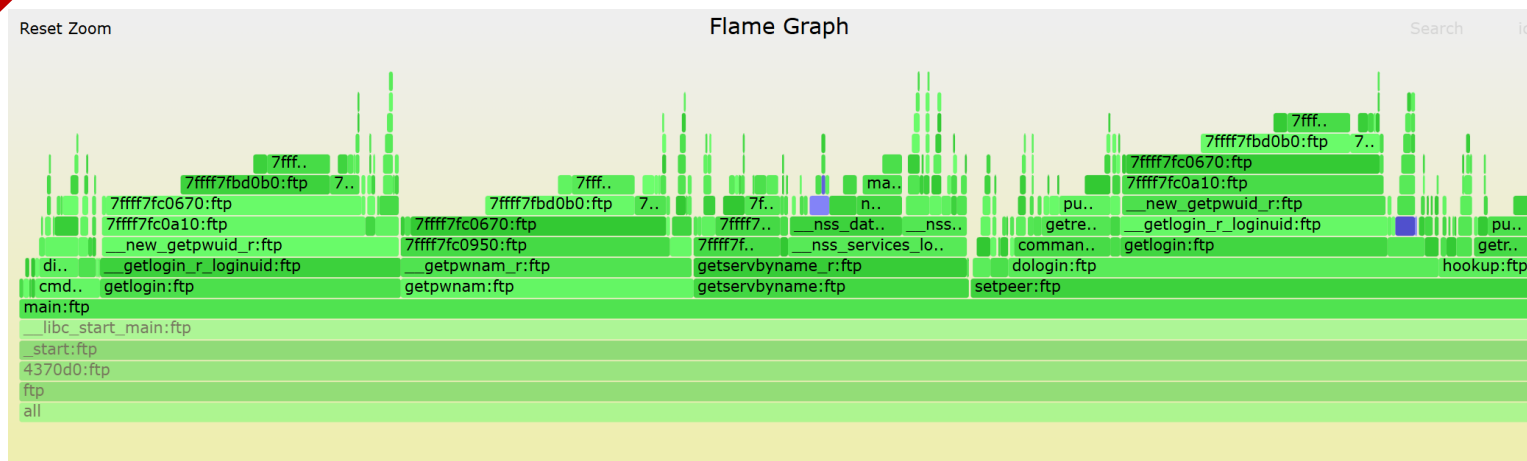
# Пример тестового сценария динамического анализа

- "Address","Datum","Labels"
- instr\_count=17720282
- 0xbcbdb3005, ascii = q, Labels: 10, hex = 0x71
- 0xbcbdb3006, ascii = w, Labels: 10, hex = 0x77
- 0xbcbdb3007, ascii = e, Labels: 10, hex = 0x65
- 0xbcbdb3008, ascii = 1, Labels: 10, hex = 0x31
- 0xbcbdb3009, ascii = 2, Labels: 10, hex = 0x32
- 0xbcbdb300a, ascii = 3, Labels: 10, hex = 0x33
- 0xbcbdb300b, ascii = Q, Labels: 10, hex = 0x51
- 0xbcbdb300c, ascii = W, Labels: 10, hex = 0x57
- 0xbcbdb300d, ascii = E, Labels: 10, hex = 0x45
- 0xc5148006, ascii = [.] , Labels: 10, hex = 0x0
- 0xc5148007, ascii = [.] , Labels: 10, hex = 0x0
- 0xc5148008, ascii = [.] , Labels: 10, hex = 0x0
- 0xc5148009, ascii = [.] , Labels: 10, hex = 0x0
- 0xc514800a, ascii = [.] , Labels: 10, hex = 0x0
- 0xc514800b, ascii = [.] , Labels: 10, hex = 0x0
- 0xc514800c, ascii = [.] , Labels: 10, hex = 0x0
- 0xc514800d, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e0, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e1, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e2, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e3, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e4, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e5, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e6, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e7, ascii = [.] , Labels: 10, hex = 0x0
- 0xe28551e8, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac5746, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac5747, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac5748, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac5749, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac574a, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac574b, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac574c, ascii = [.] , Labels: 10, hex = 0x0
- 0xe5ac574d, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf750, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf751, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf752, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf753, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf754, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf755, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf756, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf757, ascii = [.] , Labels: 10, hex = 0x0
- 0xecabf758, ascii = [.] , Labels: 10, hex = 0x0
- 0x1228551e2, ascii = e, Labels: 10, hex = 0x65
- 0x1228551e3, ascii = 1, Labels: 10, hex = 0x31
- 0x1228551e4, ascii = 2, Labels: 10, hex = 0x32
- 0x1228551e5, ascii = 3, Labels: 10, hex = 0x33
- 0x1228551e6, ascii = Q, Labels: 10, hex = 0x51
- 0x1228551e7, ascii = W, Labels: 10, hex = 0x57
- 0x1228551e8, ascii = E, Labels: 10, hex = 0x45
- 0x12cabf750, ascii = q, Labels: 10, hex = 0x71
- 0x12cabf751, ascii = w, Labels: 10, hex = 0x77
- 0x12cabf752, ascii = e, Labels: 10, hex = 0x65
- 0x12cabf753, ascii = 1, Labels: 10, hex = 0x31
- 0x12cabf754, ascii = 2, Labels: 10, hex = 0x32
- 0x12cabf755, ascii = 3, Labels: 10, hex = 0x33
- 0x12cabf756, ascii = Q, Labels: 10, hex = 0x51
- 0x12cabf757, ascii = W, Labels: 10, hex = 0x57

# Пример тестового сценария динамического анализа



## Пример тестового сценария динамического анализа



## Динамический анализ. Плюсы

- Практически не зависит от исследуемого программного обеспечения
- Позволяет проводить анализ помеченных данных
- Позволяет проводить анализ ассемблерной трассы выполнения
- Само тестирование и анализ не требует большого количества времени
- Нет ограничений по реализуемым сценариям

## Динамический анализ. Минусы

- Требует отдельной сборки для проведения анализа
- Требует подготовки (разработки) эффективный сценариев
- Высокие требования к вычислительным ресурсам

# Пример тестового сценария фаззинг-тестирования

Название	/opt/skvs_fuzzman /fuzzman_server_analyzing/app/tp
Анализ начал	17.09.2025, 00:24:46
Последнее обновление	17.09.2025, 00:25:46
Циклов выполнено	0
Исполнений (вызовов execve())	56799
Вызовов в секунду	944.04
Всего тестов	197
Эффективных (для анализа) тестов	19
Максимальная глубина	3
Номер текущего теста	158
Осталось выполнить эффективных тестов	16
Осталось выполнить всего тестов	153
Число нестабильных тестов	0
Стабильность	100.00%
Покрытия	7.52%
Уникальных аварий	0
Уникальных зависаний	1
С последнего нахождения пути прошло	55сек
С последней аварии прошло	—
С последнего зависания прошло	56сек
Исполнений с последней аварией	56799
Таймаут исполнения	20
Самое медленное исполнение	0мс

Название	/opt/skvs_fuzzman /fuzzman_server_analyzing/app/tp
Анализ начал	17.09.2025, 00:24:46
Последнее обновление	17.09.2025, 00:25:46
Циклов выполнено	0
Исполнений (вызовов execve())	45287
Вызовов в секунду	752.99
Всего тестов	191
Эффективных (для анализа) тестов	18
Максимальная глубина	4
Номер текущего теста	169
Осталось выполнить эффективных тестов	11
Осталось выполнить всего тестов	142
Число нестабильных тестов	0
Стабильность	100.00%
Покрытия	7.21%
Уникальных аварий	0
Уникальных зависаний	8
С последнего нахождения пути прошло	53сек
С последней аварии прошло	—
С последнего зависания прошло	53сек
Исполнений с последней аварией	45287
Таймаут исполнения	20
Самое медленное исполнение	0мс

Название	/opt/skvs_fuzzman /fuzzman_server_analyzing /app/tp
Анализ начал	17.09.2025, 00:24:46
до остановки Последнее обновления	17.09.2025, 11:26:45
Циклов выполнено	1
Исполнений (вызовов execve())	12887186
Вызовов в секунду	324.46
Всего тестов	1606
Эффективных (для анализа) тестов	80
Максимальная глубина	14
Номер текущего теста	518
Осталось выполнить эффективных тестов	0
Осталось выполнить всего тестов	210
Число нестабильных тестов	27
Стабильность	66.67%
Покрытия	25.52%
Уникальных аварий	133
Уникальных зависаний	31
С последнего нахождения пути до остановки прошло	26мин 16сек
С последней аварии до остановки прошло	2ч 8мин 14сек
С последнего зависания до остановки прошло	2ч 32мин 47сек
Исполнений с последней аварией	2294716
Таймаут исполнения	20
Самое медленное исполнение	0мс

Название	/opt/skvs_fuzzman /fuzzman_server_analyzing /app/tp
Анализ начал	17.09.2025, 00:24:46
до остановки Последнее обновления	17.09.2025, 11:26:45
Циклов выполнено	2
Исполнений (вызовов execve())	13090979
Вызовов в секунду	329.59
Всего тестов	1638
Эффективных (для анализа) тестов	69
Максимальная глубина	16
Номер текущего теста	246
Осталось выполнить эффективных тестов	0
Осталось выполнить всего тестов	141
Число нестабильных тестов	20
Стабильность	90.04%
Покрытия	25.52%
Уникальных аварий	125
Уникальных зависаний	47
С последнего нахождения пути до остановки прошло	35мин 16сек
С последней аварии до остановки прошло	56мин 44сек
С последнего зависания до остановки прошло	8ч 44мин 26сек
Исполнений с последней аварией	997638
Таймаут исполнения	20
Самое медленное исполнение	0мс

## Фаззинг-тестирование. Плюсы

- После запуска не требует вмешательства
- За счет мутаций позволяет рассмотреть разнообразные входные данные
- Отсутствие ложноположительных срабатываний
- Позволяет обнаруживать уязвимости нулевого дня



## Фаззинг-тестирование. Минусы

- Достижение удовлетворительных результатов может занять длительное время
- Требуется подготовки таких входных данных, использование которых позволит получить наибольшее изначальное покрытие
- Высокие требования к вычислительным ресурсам
- Зависит от инструментов разработки исследуемого ПО

**СПАСИБО ЗА ВНИМАНИЕ!**

Чуманов Никита Олегович,

Ведущий эксперт Центра сертификации АО «НПО «Эшелон» [n.chumanov@npo-echelon.ru](mailto:n.chumanov@npo-echelon.ru)

Ежов Артём Александрович,

Ведущий эксперт Центра сертификации АО «НПО «Эшелон» [a.ezhov@npo-echelon.ru](mailto:a.ezhov@npo-echelon.ru)