ОБНАРУЖЕНИЕ АНОМАЛЬНЫХ СЕТЕВЫХ СОЕДИНЕНИЙ НА ОСНОВЕ ГИБРИДИЗАЦИИ МЕТОДОВ ВЫЧИСЛИТЕЛЬНОГО ИНТЕЛЛЕКТА

Выступающий: А.А. Браницкий
Руководители: д.т.н., проф. А.В. Тимофеев,
д.т.н., проф. И.В. Котенко

Федеральное государственное бюджетное учреждение науки Санкт-Петербургский институт информатики и автоматизации Российской академии наук

Санкт-Петербург, 2018

Актуальность темы исследования

- Совершенствование информационных технологий и рост компьютерных сетей порождают сетевой трафик, обладающий свойствами динамичности и несезонной изменчивости
- Большинство систем обнаружения атак (СОА) не способны выявлять модификации сетевых атак
- Применение адаптивных подходов позволяет избавиться от постоянного обновления базы сигнатур атак в СОА
- „Слепое“ использование адаптивных подходов для обнаружения широкого класса атак приводит к увеличению числа ложных срабатываний
- Слабая ориентация существующих СОА на классификацию сетевых атак приводит к формированию неоднозначных действий, направленных на предотвращение сетевых угроз
- Требуется разработать подход, позволяющий совместить основные преимущества сигнатурных и адаптивных механизмов, для решения задачи обнаружения аномальных сетевых соединений

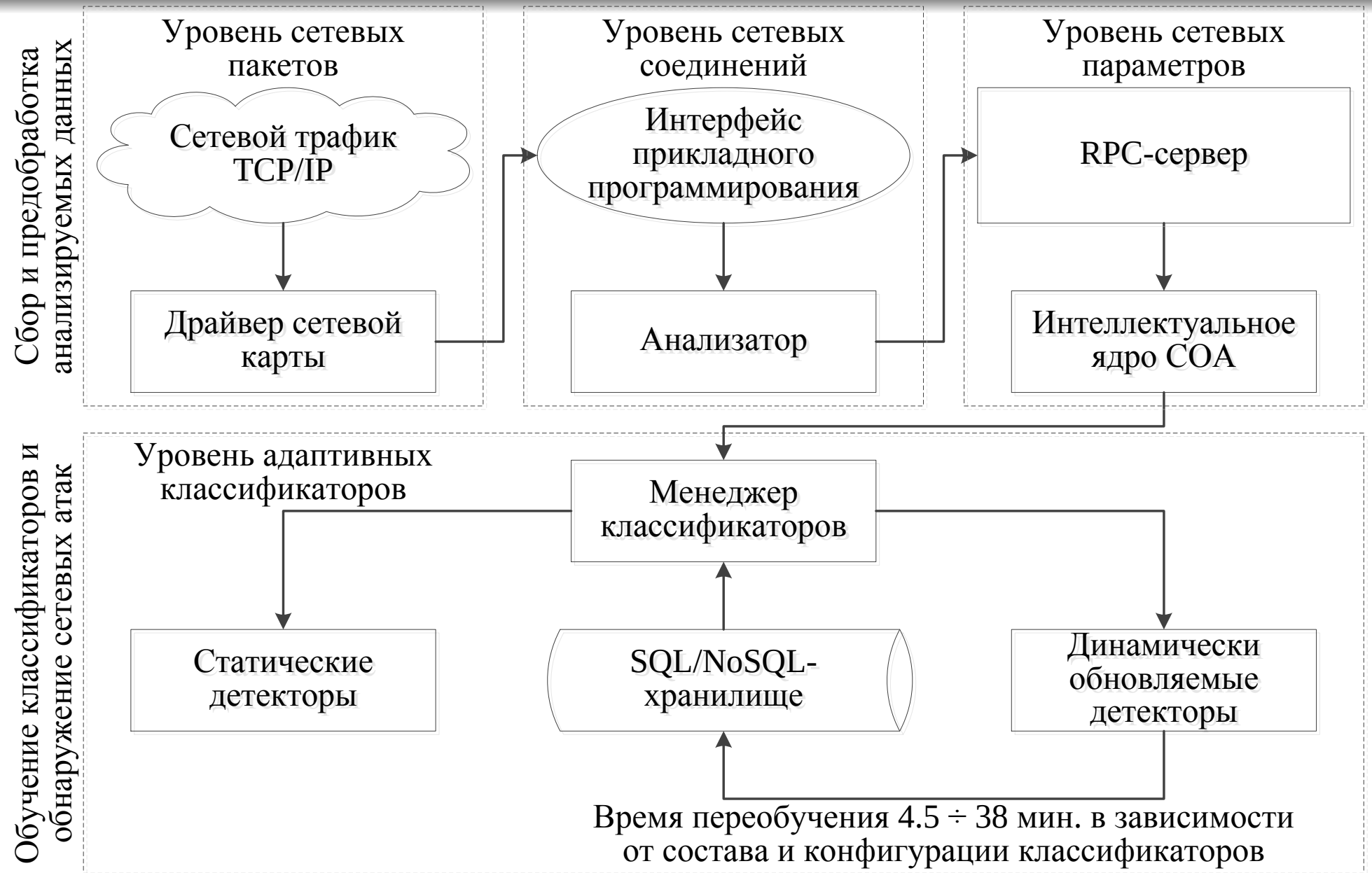
Цель и задачи исследования

Цель: повышение эффективности функционирования СОА при помощи подхода „гибридизация методов вычислительного интеллекта (ВИ)“.

Задачи:

1. анализ сигнатурных и эвристических методов обнаружения сетевых атак
2. разработка программных инструментов для тестирования сетевых СОА и оценка их возможностей
3. разработка модели искусственной иммунной системы на базе эволюционного подхода для классификации сетевых соединений
4. разработка алгоритма генетико-конкурентного обучения сети Кохонена для обнаружения аномальных сетевых соединений
5. разработка методики иерархической гибридизации бинарных классификаторов для обнаружения аномальных сетевых соединений
6. разработка архитектуры и программная реализация распределенной СОА, построенной на основе гибридизации методов ВИ и сигнатурного анализа
7. разработка программного стенда для генерации сетевых атак и экспериментальная оценка разработанной СОА

Предлагаемый подход



Подходы к обнаружению сетевых атак

Авторы (год)	**Система**	**Математический аппарат**
Ilgun K., Kemmerer R. A., Porras P. A. (1995)	USTAT	Конечные автоматы
Kumar S., Spafford E. H. (1995)	IDIOT	Раскрашенные сети Петри
Bombonato F. (2003)	Beholder IDS	Нейронные сети
Paxson V. (1995)	Bro	Язык описания сценариев
Roesch M. (1998)	Snort	Сопоставление с образцом
Lunt T. F., Tamaru A., Gillham F., [et al.] (1992)	IDES	Экспертно-логический вывод
Barbara D., Wu N., Jajodia S. (2001)	ADAM	Наивный байесовский классификатор
Hofmeyr S. A., Forrest S. (2000)	LISYS	Иммунные системы
Amini M., Rezaeenoor J., Hadavandi E. (2014)	—	Нейронные сети и нечеткая кластеризация
Brindasri S., Saravanan K. (2014)	—	Марковская цепь первого порядка

Постановка задачи исследования

Исходные данные:

- Множество классов сетевых соединений $\mathcal{C} = \{C_0, \ldots, C_m\}$
  - ✓ Класс нормальных сетевых соединений C_0
  - ✓ Классы аномальных сетевых соединений $C_1, \ldots, C_m$
- Набор маркированных векторов признаков сетевых соединений
$$\left(\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(LS)}} = \{(\boldsymbol{x_i}, \bar{c}_i)\}_{i=1}^{M}\right) \bigcup \left(\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(TS)}} = \{(\boldsymbol{z_i}, \bar{c}_i)\}_{i=1}^{M^*}\right)$$
- Множество конфигурационных параметров дерева классификаторов и отдельных классификаторов

Ограничения и допущения:

- Между узловыми сенсорами и коллектором COA организован высокоскоростной безопасный канал передачи данных
- Среда функционирования COA является безопасной с точки зрения вызова системных функций ОС
- Права суперпользователя для запуска компонентов COA предоставлены

Требуется выявлять аномальные сетевые соединения и минимизировать ошибку неверного соотнесения меток классов

$$\frac{1}{M^*} \cdot \#\left\{\boldsymbol{z_i} \,|G\left(\boldsymbol{z_i}\right) \neq \bar{c}_i\right\}_{i=1}^{M^*} \longrightarrow \min_{G}$$

Постановка задачи исследования

Требуется разработать:

1. модель искусственной иммунной системы на базе эволюционного подхода, которая является независимой от структуры внутреннего представления иммунных детекторов и учитывает свойства динамического непостоянства сетевого трафика
2. алгоритм генетико-конкурентного обучения сети Кохонена, который направлен на снижение числа эпох обучения за счет использования различных стратегий генетической оптимизации весов „мертвых“ нейронов
3. методику иерархической гибридизации бинарных классификаторов, которая, в отличие от остальных, предоставляет возможность объединения разнородных решателей без строгой привязки к агрегирующей их выходы композиции
4. архитектуру распределенной СОА, построенной на основе гибридизации методов ВИ и сигнатурного анализа

Результаты, выносимые на защиту

Модель искусственной иммунной системы на базе эволюционного подхода для классификации сетевых соединений

Алгоритм генетико-конкурентного обучения сети Кохонена для обнаружения аномальных сетевых соединений

Методика иерархической гибридизации бинарных классификаторов для обнаружения аномальных сетевых соединений

Архитектура и программная реализация распределенной СОА, построенной на основе гибридизации методов ВИ и сигнатурного анализа

Результаты

Результат 1

Модель искусственной иммунной системы на базе эволюционного подхода для классификации сетевых соединений

Постановка задачи

Исходные данные:

- $\mathcal{C} = \{C_0, \ldots, C_m\}$ — множество классов сетевых соединений
- $\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(LS)}} = \{(\boldsymbol{x_i}, \bar{c}_i)\}_{i=1}^{M}$ — маркированная обучающая выборка
- $\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(TS)}} = \{(\boldsymbol{z_i}, \bar{c}_i)\}_{i=1}^{M^*}$ — маркированная контрольная выборка

Ограничения и допущения:

- Модель должна динамически переобучаться на новых данных

Требуется: разработать модель $AISEA$ и алгоритм классификации G сетевых соединений на основе этой модели:

$$\frac{\#\left\{\boldsymbol{z_i} \,\middle|\, \boldsymbol{z_i} \in \mathcal{X}_{\mathcal{C}}^{(TS)} \wedge \{\bar{c}_i\} = G(\boldsymbol{z_i})\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i} \,\middle|\, \boldsymbol{z_i} \in \mathcal{X}_{\mathcal{C}}^{(TS)}\right\}_{i=1}^{M^*}} \longrightarrow \max_G$$

$$\frac{\#\left\{\boldsymbol{z_i} \,\middle|\, \boldsymbol{z_i} \in \mathcal{X}_{\mathcal{C}}^{(TS)} \wedge \{\bar{c}_i\} \in G(\boldsymbol{z_i}) \wedge \#G(\boldsymbol{z_i}) \geqslant 2\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i} \,\middle|\, \boldsymbol{z_i} \in \mathcal{X}_{\mathcal{C}}^{(TS)}\right\}_{i=1}^{M^*}} \longrightarrow \min_G$$

Искусственная иммунная система

$AISEA = \langle \mathcal{D}_\mathcal{T}, \mathcal{D}_\mathcal{M}, \mathcal{S}_\mathcal{A}, \mathcal{S}_\mathcal{N}, \mathcal{G}, R, \Psi \rangle$ — иммунная система

- $\mathcal{D}_\mathcal{T} \subset \mathcal{D}$ — набор временных иммунных детекторов
- $\mathcal{D}_\mathcal{M} \subset \mathcal{D}$ — набор иммунных детекторов памяти
- $\mathcal{S}_\mathcal{A} \subset \mathcal{S}$ — обучающий набор („чужие“ объекты)
- $\mathcal{S}_\mathcal{N} \subset \mathcal{S}$ — тестирующее множество („свои“ объекты)
- $\mathcal{D} = \mathcal{D}_\mathcal{T} \bigcup \mathcal{D}_\mathcal{M}$ — набор иммунных детекторов
- $\mathcal{G} = \{G_1, \ldots, G_K\}$ — стратегии генетической оптимизации иммунных детекторов
- $R : \mathcal{D} \times 2^{\mathcal{S}_\mathcal{A}} \times 2^{\mathcal{S}_\mathcal{N}} \times \mathcal{G} \to \mathcal{D}$ — правило обучения иммунных детекторов
- $\mathcal{S}$ — набор всевозможных входных объектов
- $\Psi : \mathcal{D} \times \mathcal{S} \to [0, +\infty)$ — функция вычисления аффинности между иммунным детектором $d \in D$ и тестовым объектом $s \in S$

$d = \langle representation, threshold, life\_time, state \rangle$ — иммунный детектор

- $representation \in \{BitString, RealVector, NeuralNetwork, PetriNet, \ldots\}$ — внутреннее представление иммунного детектора d
- $threshold \in [0, +\infty)$ — порог активации иммунного детектора d
- $life\_time \in [0, +\infty]$ — срок жизни иммунного детектора d
- $state \in \{immature, semimature, mature, memory\}$ — текущее состояние иммунного детектора d

Жизненный цикл иммунного детектора

$\mathcal{D}_{\mathcal{M}}$ и $threshold$

- Для обнаружения каждого класса атаки $C \in \mathcal{C}$ выделяется несколько иммунных детекторов, объединяющихся в класс детекторов $\mathcal{D}_C$ ($\bigcup_{C \in \mathcal{C}} \mathcal{D}_C = \mathcal{D}$).
- Каждый из детекторов $d \in \mathcal{D}_C$ использует бутстреп-обучение на разных случайных подвыборках множеств $\mathcal{S}_{\mathcal{A}}$ и $\mathcal{S}_{\mathcal{N}} - \mathcal{S}_{\mathcal{A}}^{(d)}$ и $\mathcal{S}_{\mathcal{N}}^{(d)}$.
- Под q-ой группой детекторов понимается набор детекторов $\mathcal{D}^{(q)}$ ($\bigcup_{q=1}^{m} \mathcal{D}^{(q)} = \mathcal{D}$, m — число классов атак), полностью покрывающих всё множество классов атак.
- Для каждого класса атаки $C \in \mathcal{C}$ определяется ровно один детектор памяти $d_m^{(C)}$ — детектор, который удовлетворяет требованию наибольшей степени приспособленности к распознаванию объектов класса C.

$$\mathcal{D}_{\mathcal{M}} = \bigcup_{C \in \mathcal{C}} \left\{ \operatorname{Arg\,max}_{d \in \mathcal{D}_C} \left(\sum_{s \in \mathcal{S}_{\mathcal{A}}^{(d)} \cap C} \Psi(d, s) \Big/ \# \left(\mathcal{S}_{\mathcal{A}}^{(d)} \cap C \right) \right) \right\}$$

$$threshold = h_d^- - \frac{h_d^- - h_d^+}{2} = \frac{h_d^- + h_d^+}{2} = \frac{\min_{s \in \mathcal{S}_{\mathcal{A}}^{(d)}} \Psi(d,s) + \max_{s \in \mathcal{S}_{\mathcal{N}}^{(d)}} \Psi(d,s)}{2}$$

Алгоритм классификации соединений

1. Вычисление для каждого иммунного детектора $d \in \mathcal{D}$ значения его активации $a_d = \Psi(d, s) - threshold$.
2. Мажоритарное голосование внутри каждого класса детекторов $\mathcal{D}_C$. Если $\left(A_C \leftrightharpoons \sum_{d \in \mathcal{D}_C} [a_d \geqslant 0]\right) > \left(B_C \leftrightharpoons \sum_{d \in \mathcal{D}_C} [a_d < 0]\right)$, то s распознается как „чужой“ объект. Если $A_C < B_C$, то s распознается как „свой“ объект. В случае наличия конфликтов, т.е. $A_C = B_C$, s классифицируется как „чужой“ объект, если $a_{d_m^{(C)}} \geqslant 0$, и s классифицируется как „свой“ объект, если $a_{d_m^{(C)}} < 0$, где $d_m^{(C)} \in \mathcal{D}_{\mathcal{M}} \bigcap \mathcal{D}_C$.
3. Формирование множества классов иммунных детекторов $\{\mathcal{D}_{C'}\}_{C' \in \mathcal{C}^*}$, которые распознают входной объект s как „чужой“ объект.
4. Определение класса объекта s. Если $\mathcal{C}^* = \varnothing$, то объект s относится к классу „своих“ объектов. Если $E_{\mathcal{C}^*} \leftrightharpoons \max_{C' \in \mathcal{C}^*} A_{C'}$ достигается в одной единственной точке, то класс объекта s — это $\arg E_{\mathcal{C}^*}$, иначе класс объекта s — это $\arg\max_{C' \in \{\arg E_{\mathcal{C}^*}\}} \sum_{d \in \mathcal{D}_{C'}} a_d \cdot [a_d \geqslant 0]$.

Отличия от других моделей

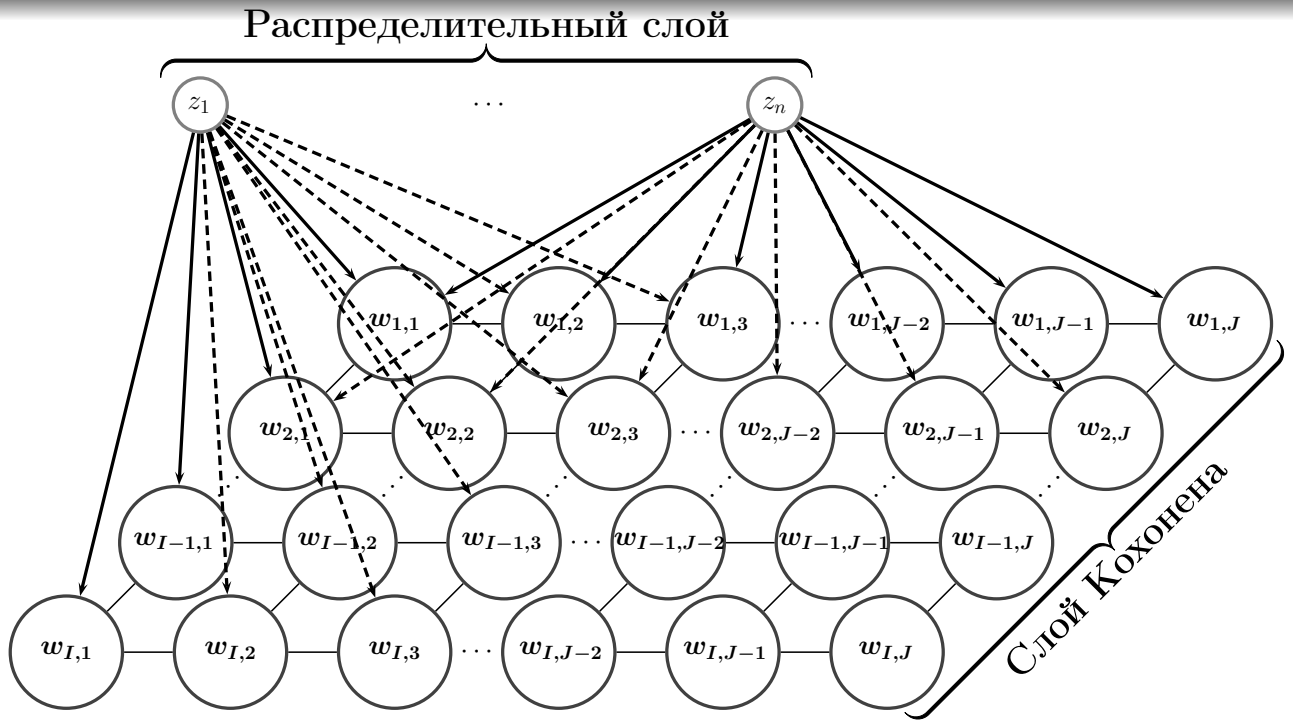
- Двухуровневый алгоритм обучения с автоматическим вычислением порога активации иммунных детекторов
- Механизм разрешения конфликтных случаев классификации за счет мажоритарного голосования, детекторов памяти и учета суммы взвешенных голосов от активировавшихся на данный стимул детекторов
- Независимость от внутренней структуры представления иммунных детекторов (универсальность модели)

Результаты

Результат 2

Алгоритм генетико-конкурентного обучения сети Кохонена для обнаружения аномальных сетевых соединений

Постановка задачи



Исходные данные:

- Инициализированные случайными малыми значениями веса сети Кохонена: $\left\{\boldsymbol{w_{ij}}\right\}_{\substack{1\leqslant i\leqslant I\\1\leqslant j\leqslant J}}$
- Набор векторов признаков сетевых соединений класса C: $\left\{\boldsymbol{x_i}\,|\boldsymbol{x_i}\in C\right\}_{i=1}^{M}$
- Набор стратегий генетической оптимизации весов: $\mathcal{G}=\{G_1,G_2,G_3\}$

Ограничения и допущения:

- $T_{modified}\leqslant T_{standard}$

Требуется: разработать алгоритм обучения сети Кохонена:

$$E(\boldsymbol{w_{11}},\ldots,\boldsymbol{w_{IJ}})=\tfrac{1}{2}\cdot\sum_{k=1}^{M}\left(D\left(\boldsymbol{w_{i_k^*j_k^*}},\boldsymbol{x_k}\right)\right)^2\longrightarrow\min_{\boldsymbol{w_{11}},\ldots,\boldsymbol{w_{IJ}}},$$

$$D(\boldsymbol{w},\boldsymbol{x})=\sqrt{(\boldsymbol{w}-\boldsymbol{x})^T\cdot(\boldsymbol{w}-\boldsymbol{x})},(i_k^*,j_k^*)=F_{IJ}(\boldsymbol{x_k})=\arg\min_{\substack{1\leqslant i\leqslant I\\1\leqslant j\leqslant J}}D\left(\boldsymbol{w_{ij}},\boldsymbol{x_k}\right)$$

Стандартный и модифицированный алгоритмы

Стандартный алгоритм

for t **in** $1..T$ **do**
#Ширина окрестности:
$\sigma(t) := \sigma_0 \cdot \exp\{-\ln(\sigma_0) \cdot t/T\}$
(*)
for $\boldsymbol{x}$ **in** $\{\boldsymbol{x_i} | \boldsymbol{x_i} \in C\}_{i=1}^{M}$ **do**
#Нейрон-победитель:
$(i^*, j^*) := F_{IJ}(\boldsymbol{x})$
#Текущая окрестность:
$V^* := \{(i,j) \,|\, d^*(i,j) < \sigma^2(t)\}$,
где $d^*(i,j) = (i-i^*)^2 + (j-j^*)^2$
for (i,j) **in** V^* **do**
#Обновление весов:
$\boldsymbol{w_{ij}} := \boldsymbol{w_{ij}} + \kappa(t) \cdot \varphi(i,j,t) \cdot (\boldsymbol{x} - \boldsymbol{w_{ij}})$,
где $\kappa(t) = \kappa_0 \cdot e^{-\eta \cdot t}$, $\varphi(i,j,t) = e^{-\frac{d^*(i,j)}{2 \cdot \sigma^2(t)}}$
od
(**)
od
(***)
od
(****)

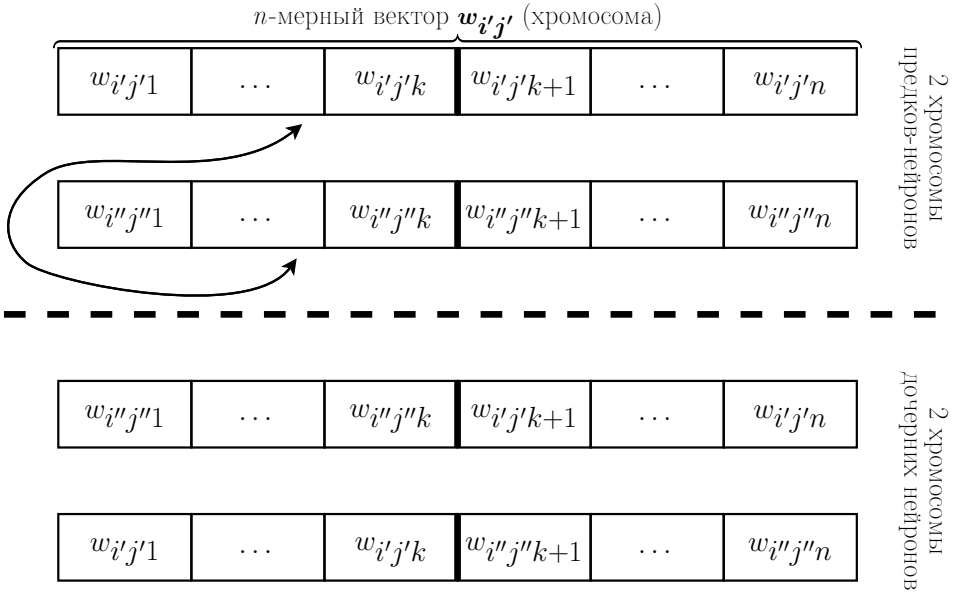
Модифицированный алгоритм

(*) #Набор активных нейронов:
$V^+ := \varnothing$
(**) #Расширение набора #активных нейронов:
$V^+ := V^+ \bigcup \{(i^*, j^*)\}$
#Интерактивная настройка:
применение стратегии G' к весам $\boldsymbol{w_{ij}}$, где $(i,j) \in V^\dagger = V^* \setminus \{(i^*, j^*)\}$,
$\Psi_{ijk} = \|\boldsymbol{x} - \boldsymbol{w_{ij}}\|^{-1}$ — ФП
(***) #Пакетная настройка:
применение стратегии G' к весам $\boldsymbol{w_{ij}}$, где $(i,j) \in V^\dagger = V^* \setminus V^+$,
$\Psi_{ij} = \left(\sum_{\boldsymbol{x} \in C} \|\boldsymbol{x} - \boldsymbol{w_{ij}}\| / M\right)^{-1}$ — ФП
(****) #Порог активации нейрона:
$h_{ij}^- := \min\limits_{\boldsymbol{x} \in C} \left\{ D^{-1}(\boldsymbol{w_{ij}}, \boldsymbol{x}) \,|\, (i,j) = F_{IJ}(\boldsymbol{x}) \right\}$
$h_{ij}^+ := \max\limits_{\boldsymbol{x} \in C_0} \left\{ D^{-1}(\boldsymbol{w_{ij}}, \boldsymbol{x}) \,|\, (i,j) = F_{IJ}(\boldsymbol{x}) \right\}$
$h_{ij} := (h_{ij}^- + h_{ij}^+)/2$

Генетические операторы

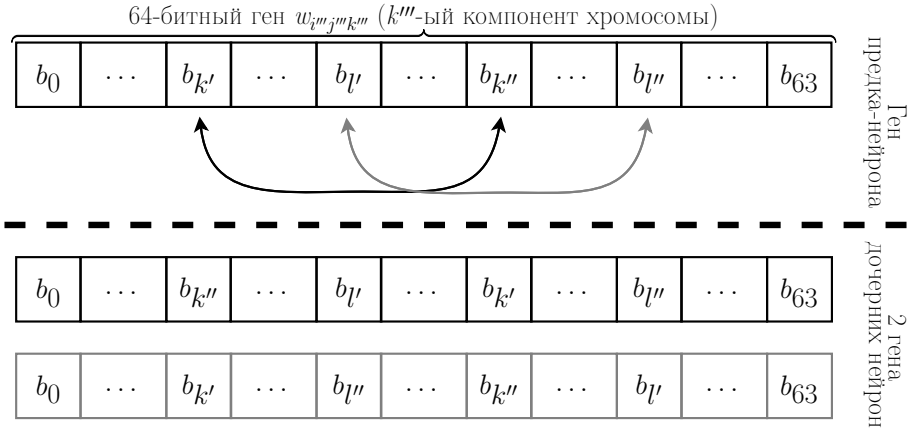
1 Кроссинговер

- ✓ Случайный выбор границы разделения хромосом
- ✓ Перестановка местами участков двух хромосом



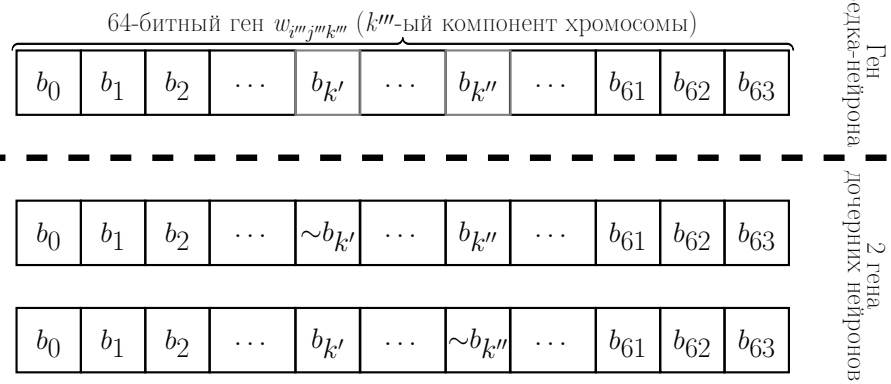
2 Мутация

- ✓ Случайный выбор бит внутри гена
- ✓ Перестановка местами двух бит



3 Инверсия

- ✓ Случайный выбор бита
- ✓ Инвертирование значения бита



Кроссинговер сохраняет целостность генов, но нарушает порядок их следования. Мутация и инверсия применяются только к части мантиссы 64-битного „вещественного гена“.

Подходы для генерации нейронов

$V := \varnothing$
while $V \neq V^\dagger$ **do**
  Выбрать очередной „мертвый“ нейрон $(i', j') \in V^\dagger$
  Выбрать случайно один из трех генетических операторов O^*
  if O^* — оператор скрещивания **then**
    Выбрать второй нейрон (i'', j'') на основе одного из трех подходов:
    1. Подход A_1 (случайный):
       $(i'', j'') \in V^* \setminus V \setminus \{(i', j')\}$
    2. Подход A_2 (геометрический):
       $(i'', j'') \in V^* \setminus V \setminus \{(i', j')\} \wedge d^*(i', j') = d^*(i'', j'')$
    3. Подход A_3 (на основе приспособленности):
       $(i'', j'') = \arg\max_{(i,j) \in V^* \setminus V \setminus \{(i', j')\}} \Psi_{ij(k)}$
    Применить оператор O^* к нейронам (i', j') и (i'', j'')
  else
    Применить дважды оператор O^* к нейрону (i', j')
  fi
  $V := V \bigcup \{(i', j')\}$
od

Стратегии генетической оптимизации

1. фиксированный выбор подхода
2. последовательный или случайный перебор подходов
3. выбор, основанный на механизме рулетки

0. $p_{A_i} := \frac{1}{3}$ — начальная вероятность выбора подхода A_i $(i=1,2,3)$
1. Генерация случайного вещественного числа $r \in [0,1]$
2. $[0,1) = r_{A_1} \vee r_{A_2} \vee r_{A_3} = [0, p_{A_1}) \vee [p_{A_1}, p_{A_2}) \vee [p_{A_1} + p_{A_2}, p_{A_3})$
3. Если $r \in r_{A_{i_1^*}}$, то $A_{i_1^*}$ — подход для генерации дочерних нейронов V_2^* и V_3^*
4. $\Psi_{V_l^*} = \frac{1}{\#V_l^*} \cdot \sum_{(i,j) \in V_l^*} \Psi_{ij(k)}$ — усредненное значение ФП $(l=1,2,3)$
5. Если $\Psi_{V_{l^*}^*} > \Psi_{V_1^*}$, то $V_{l^*}^*$ — текущее решение $(l^* = \arg\max\limits_{l=2,3} \{\Psi_{V_l^*}\})$,

$p_{A_{i_1^*}} := p_{A_{i_1^*}} + \Delta'_{A_{i_1^*}}$, $p_{A_{i_2^*}} := p_{A_{i_2^*}} - \Delta'_{A_{i_2^*}}$, $p_{A_{i_3^*}} := 1 - p_{A_{i_1^*}} - p_{A_{i_2^*}}$, где

$\Delta'_{A_{i_1^*}} = \min\left\{\frac{\Psi_{V_{l^*}^*} - \Psi_{V_1^*}}{\Psi_{V_{l^*}^*}}, 1 - p_{A_{i_1^*}}\right\}$, $\Delta'_{A_{i_2^*}} = \min\left\{\frac{1}{2} \cdot \Delta'_{A_{i_1^*}}, p_{A_{i_2^*}}\right\}$;

иначе V_1^* — текущее решение, $p_{A_{i_1^*}} := p_{A_{i_1^*}} - \Delta''_{A_{i_1^*}}$,

$p_{A_{i_2^*}} := p_{A_{i_2^*}} + \Delta''_{A_{i_2^*}}$, $p_{A_{i_3^*}} := 1 - p_{A_{i_1^*}} - p_{A_{i_2^*}}$, где

$\Delta''_{A_{i_1^*}} = \min\left\{\frac{\Psi_{V_1^*} - \Psi_{V_{l^*}^*}}{\Psi_{V_1^*}}, p_{A_{i_1^*}}\right\}$, $\Delta''_{A_{i_2^*}} = \min\left\{\frac{1}{2} \cdot \Delta''_{A_{i_1^*}}, 1 - p_{A_{i_2^*}}\right\}$

Отличия от других алгоритмов

- Генетическая оптимизация позволяет расширить множество активных нейронов сети Кохонена за счет дополнительной настройки „мертвых“ нейронов и их дальнейшего подключения к процессу распознавания сетевых атак
- Каждая стратегия манипулирует выбором подходов; в частности, стратегия „рулетка“ направлена на создание конкурентной борьбы именно между подходами (а не между особями) для генерации дочерних нейронов
- Дополнительная корректировка порога позволяет повысить способность обобщения у сети Кохонена за счет введения постфазы тестирования на нормальных экземплярах

Результаты

Результат 3

Методика иерархической гибридизации бинарных классификаторов для обнаружения аномальных сетевых соединений

Постановка задачи

Исходные данные:

- $\mathcal{C} = \{C_0, \ldots, C_m\}$ — множество классов сетевых соединений
- $F^{(1)}, \ldots, F^{(P)} : \mathbb{R}^n \to 2^{\{0,\ldots,m\}}$ — коллектив базовых классификаторов
- $\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(LS)}} = \{(\boldsymbol{x_i}, \bar{c}_i)\}_{i=1}^{M}$ — маркированная обучающая выборка
- $\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(TS)}} = \{(\boldsymbol{z_i}, \bar{c}_i)\}_{i=1}^{M^*}$ — маркированная контрольная выборка
- $F : \left(2^{\{0,\ldots,m\}}\right)^P \times \mathbb{R}^n \longrightarrow 2^{\{0,\ldots,m\}}$ — агрегирующая функция
- $G : \mathbb{R}^n \longrightarrow 2^{\{0,\ldots,m\}}$ — агрегирующая композиция
- $\zeta : \mathcal{C} \longrightarrow \{0, \ldots, m\}$ — вспомогательное отображение

Ограничения и допущения:

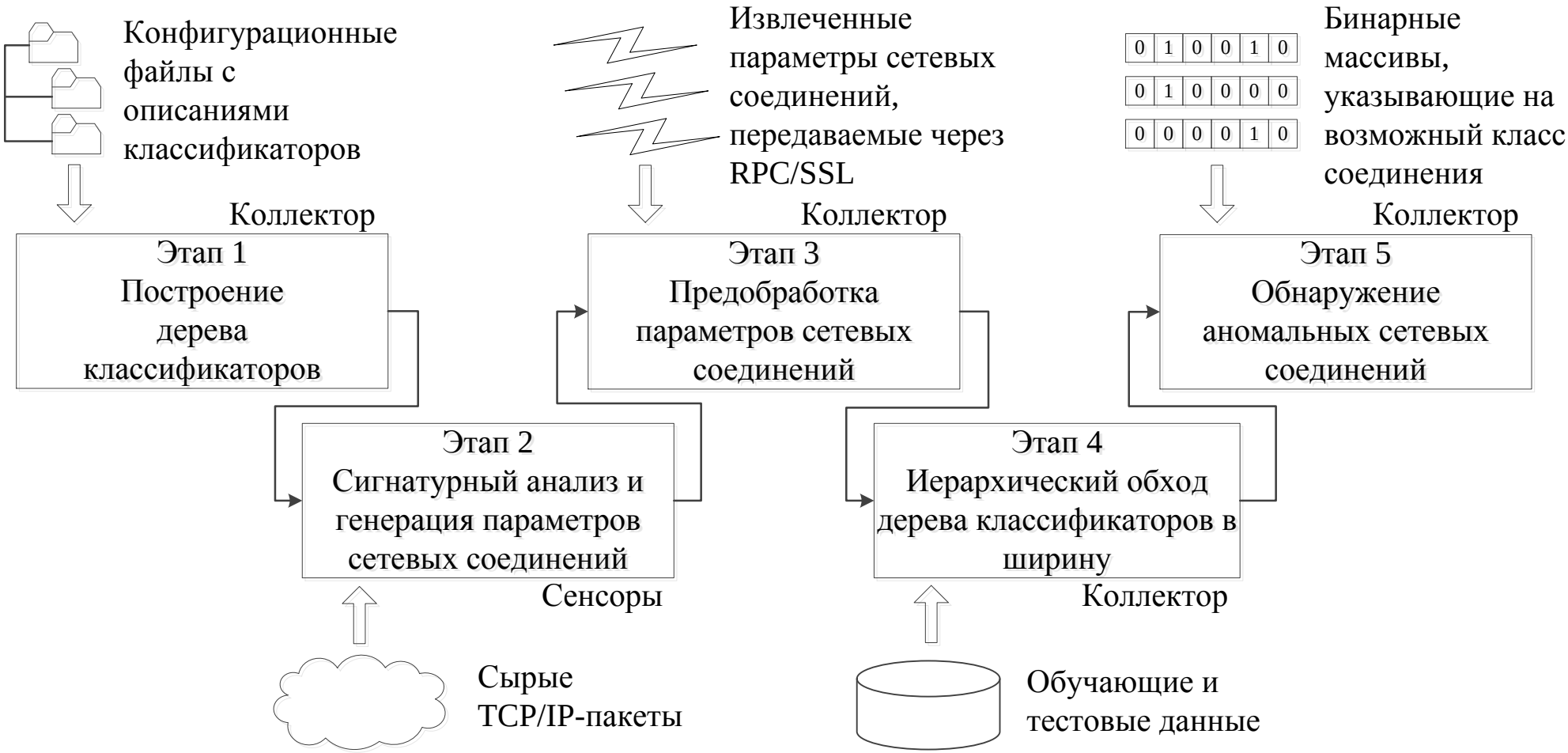
- Базовые классификаторы — это набор детекторов $F_{jk}^{(i)} : \mathbb{R}^n \to \{0,1\}$
- Настройка базовых классификаторов уже выполнена

Требуется: разработать методику:

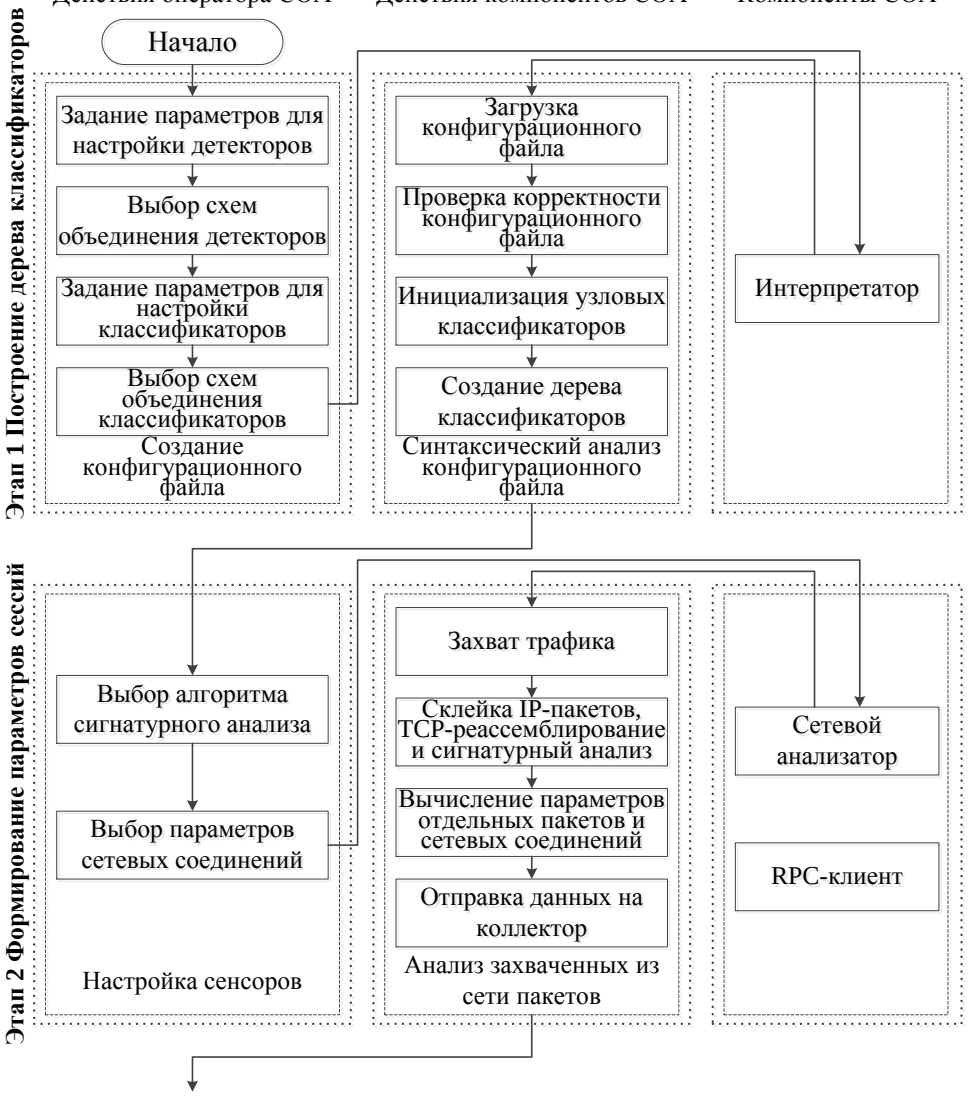
$$\Omega(G, \mathcal{X}_{\mathcal{C}}) = \Omega\left(F \circ \left[F^{(1)}, \ldots, F^{(P)}, \mathbf{id}\right], \mathcal{X}_{\mathcal{C}}\right) \leqslant \min_{j \in \{1,\ldots,P\}} \Omega\left(F^{(j)}, \mathcal{X}_{\mathcal{C}}\right),$$

$$\Omega(G, \mathcal{X}_{\mathcal{C}}) = \frac{1}{\#\mathcal{X}_{\mathcal{C}}} \cdot \#\left\{\boldsymbol{z} \,|(\exists C \in \mathcal{C}\ \boldsymbol{z} \in C) \wedge G(\boldsymbol{z}) \neq \zeta(C)\right\}_{\boldsymbol{z} \in \mathcal{X}_{\mathcal{C}}}$$

Основные этапы



Этапы 1–2



Исследуемые схемы комбинирования детекторов:

1. один-ко-всем (one-vs-all)
2. один-к-одному (one-vs-one)
3. классификационное бинарное дерево (CBT)
4. направленный ациклический граф (DAG)

Исследуемые агрегирующие композиции:

1. метод мажоритарного голосования (ММГ)

$$G^{(\mathbf{MV})}(\boldsymbol{z}) = \left\{ k \,\middle|\, \sum_{j=1}^{P} \left[k \in F^{(j)}(\boldsymbol{z}) \right] > \tfrac{1}{2} \cdot \sum_{i=0}^{m} \sum_{j=1}^{P} \left[i \in F^{(j)}(\boldsymbol{z}) \right] \right\}_{k=0}^{m}$$

2. метод взвешенного голосования (МВГ)

$$G^{(\mathbf{WV})}(\boldsymbol{z}) = \left\{ k \,\middle|\, \sum_{j=1}^{P} \omega_j \cdot \left[k \in F^{(j)}(\boldsymbol{z}) \right] = \max_{i \in \{0,\ldots,m\}} \sum_{j=1}^{P} \omega_j \cdot \left[i \in F^{(j)}(\boldsymbol{z}) \right] \right\}_{k=0}^{m}$$

$$\omega_j = \frac{\#\left\{ \boldsymbol{x_k} \,\middle|\, \bar{c}_k \in F^{(j)}(\boldsymbol{x_k}) \right\}_{k=1}^{M}}{\sum_{i=1}^{P} \#\left\{ \boldsymbol{x_k} \,\middle|\, \bar{c}_k \in F^{(i)}(\boldsymbol{x_k}) \right\}_{k=1}^{M}}$$

3. метод многоярусной укладки (ММУ)

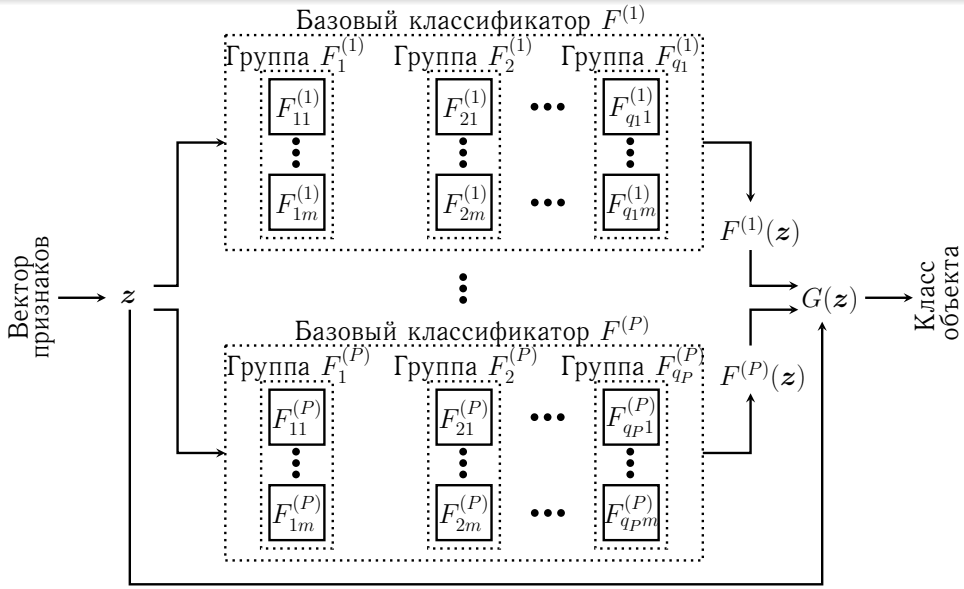
$$G^{(\mathbf{ST})}(\boldsymbol{z}) = F^{(\mathbf{ST})}\left(F^{(1)}(\boldsymbol{z}), \ldots, F^{(P)}(\boldsymbol{z}), \boldsymbol{z} \right)$$

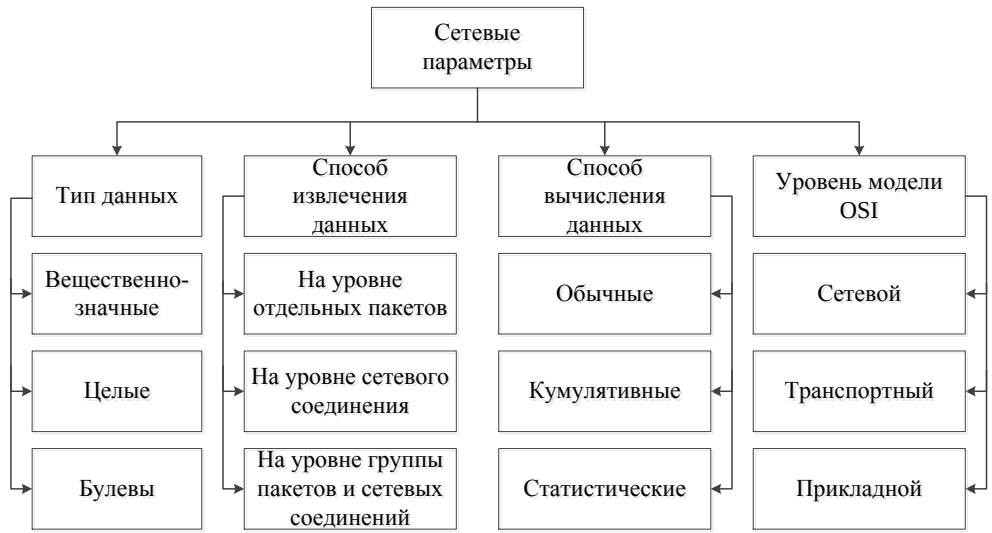
4. метод Фикса—Ходжеса (МФХ)

$$G^{(\mathbf{FH})}(\boldsymbol{z}) = \bigcup_{l=1}^{P} \left(F^{(l)}(\boldsymbol{z}) \,\middle|\, \underbrace{\sum_{j=1}^{\widetilde{M}} \left[F^{(l)}(\boldsymbol{x_{i_j}}) = \left\{ \bar{c}_{i_j} \right\} \right]}_{D_{\widetilde{M}}(l)} = \max_{k \in \{1,\ldots,P\}} D_{\widetilde{M}}(k) \right)$$

$$\left\{ i_1, \ldots, i_{\widetilde{M}} \right\} = \arg \min_{\substack{\widetilde{I} \subseteq \{1,\ldots,M\} \\ (\#\widetilde{I} = \widetilde{M})}} \sum_{i \in \widetilde{I}} \rho\left(\boldsymbol{x_i}, \boldsymbol{z} \right)$$

Дерево классификаторов и сетевые параметры





- Каждый классификатор $F^{(i)}$ $(i = 1, \ldots, P)$ содержит q_i групп $F_j^{(i)}$ $(j = 1, \ldots, q_i)$
- Каждая группа объединяет m или больше детекторов $F_{jk}^{(i)}$ $(k = 1, \ldots, m)$
- Группа детекторов $F_j^{(i)}$ обучается на различных случайных бутстреп-подвыборках из $\Upsilon_{\mathcal{X}_{\mathcal{C}}^{(LS)}}$
- Объединение групп $F_j^{(i)}$ в классификатор $F^{(i)}$ осуществляется на основе гибридного правила, представляющего собой смесь голосования большинством и голосования max-wins:

$$F^{(i)}(\boldsymbol{z}) = \left\{ \bar{c} \,\middle|\, \underbrace{\sum_{j=1}^{q_i} \left[\bar{c} \in F_j^{(i)}(\boldsymbol{z}) \right]}_{\Xi_i(\bar{c})} > \tfrac{1}{2} \cdot q_i \wedge \Xi_i(\bar{c}) = \max_{\bar{c}' \in \{0,\ldots,m\}} \Xi_i(\bar{c}') \right\}_{\bar{c}=0}^{m}$$

Адаптированный метод скользящей средней для вычисления статистических показателей (интенсивности пакетов):

- $\Delta_0^{(L)} = [0, L] = \bigcup_{i=0}^{k-1} \Delta_{\delta \cdot i}^{(L')}$ — наблюдаемый временной интервал $(0 < L' \leqslant L,\ 0 < \delta \leqslant L')$
- $k = 1 + \lfloor \frac{L - L'}{\delta} \rfloor$ — число скользящих интервалов
- ω_i — слепок параметров в течение $\Delta_{\delta \cdot i}^{(L')}$
- $\bar{\omega} = \frac{1}{k} \cdot \sum_{i=0}^{k-1} \omega_i$ — средняя величина (интенсивность)

Этапы 3–5

Показатели эффективности СОА:

1. корректности обнаружения сетевых атак

$$TPR=\frac{TP}{TP+FN}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\bar{c}_i\neq 0\wedge 0\notin G(\boldsymbol{z_i})\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\bar{c}_i\neq 0\right\}_{i=1}^{M^*}}$$

2. ложных срабатываний:

$$FPR=\frac{FP}{FP+TN}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\bar{c}_i=0\wedge 0\notin G(\boldsymbol{z_i})\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\bar{c}_i=0\right\}_{i=1}^{M^*}}$$

3. корректности классификации соединений:

$$CCR=\frac{CC}{TP+FN+FP+TN}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\{\bar{c}_i\}=G(\boldsymbol{z_i})\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\right\}_{i=1}^{M^*}}$$

4. некорректной классификации:

$$ICR=\frac{IC}{TP+FN+FP+TN}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\wedge\bar{c}_i\notin G(\boldsymbol{z_i})\right\}_{i=1}^{M^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\mathcal{X}_{\mathcal{C}}^{(TS)}\right\}_{i=1}^{M^*}}$$

5. обобщающей способности при обнаружении:

$$GPR=\frac{\overline{TP^*}}{\overline{TP^*}+\overline{FN^*}}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)}\setminus\mathcal{X}_{\mathcal{C}}^{(LS)}\wedge\bar{c}_i\neq 0\wedge 0\notin G(\boldsymbol{z_i})\right\}_{i=1}^{\bar{M}^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)}\setminus\mathcal{X}_{\mathcal{C}}^{(LS)}\wedge\bar{c}_i\neq 0\right\}_{i=1}^{\bar{M}^*}}$$

6. переобученности при обнаружении:

$$OPR=\overline{TPR}-GPR=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(LS)}\wedge\bar{c}_i\neq 0\wedge 0\notin G(\boldsymbol{z_i})\right\}_{i=1}^{\bar{M}}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(LS)}\wedge\bar{c}_i\neq 0\right\}_{i=1}^{\bar{M}}}-GPR$$

7. обобщающей способности при классификации:

$$GCR=\frac{\overline{CC^*}}{\overline{TP^*}+\overline{FN^*}+\overline{FP^*}+\overline{TN^*}}=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)}\setminus\mathcal{X}_{\mathcal{C}}^{(LS)}\wedge\{\bar{c}_i\}=G(\boldsymbol{z_i})\right\}_{i=1}^{\bar{M}^*}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)}\setminus\mathcal{X}_{\mathcal{C}}^{(LS)}\right\}_{i=1}^{\bar{M}^*}}$$

8. переобученности при классификации:

$$OCR=\overline{CCR}-GCR=\frac{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(LS)}\wedge\{\bar{c}_i\}=G(\boldsymbol{z_i})\right\}_{i=1}^{\bar{M}}}{\#\left\{\boldsymbol{z_i}\middle|\boldsymbol{z_i}\in\bar{\mathcal{X}}_{\mathcal{C}}^{(LS)}\right\}_{i=1}^{\bar{M}}}-GCR$$

Отличия от других методик

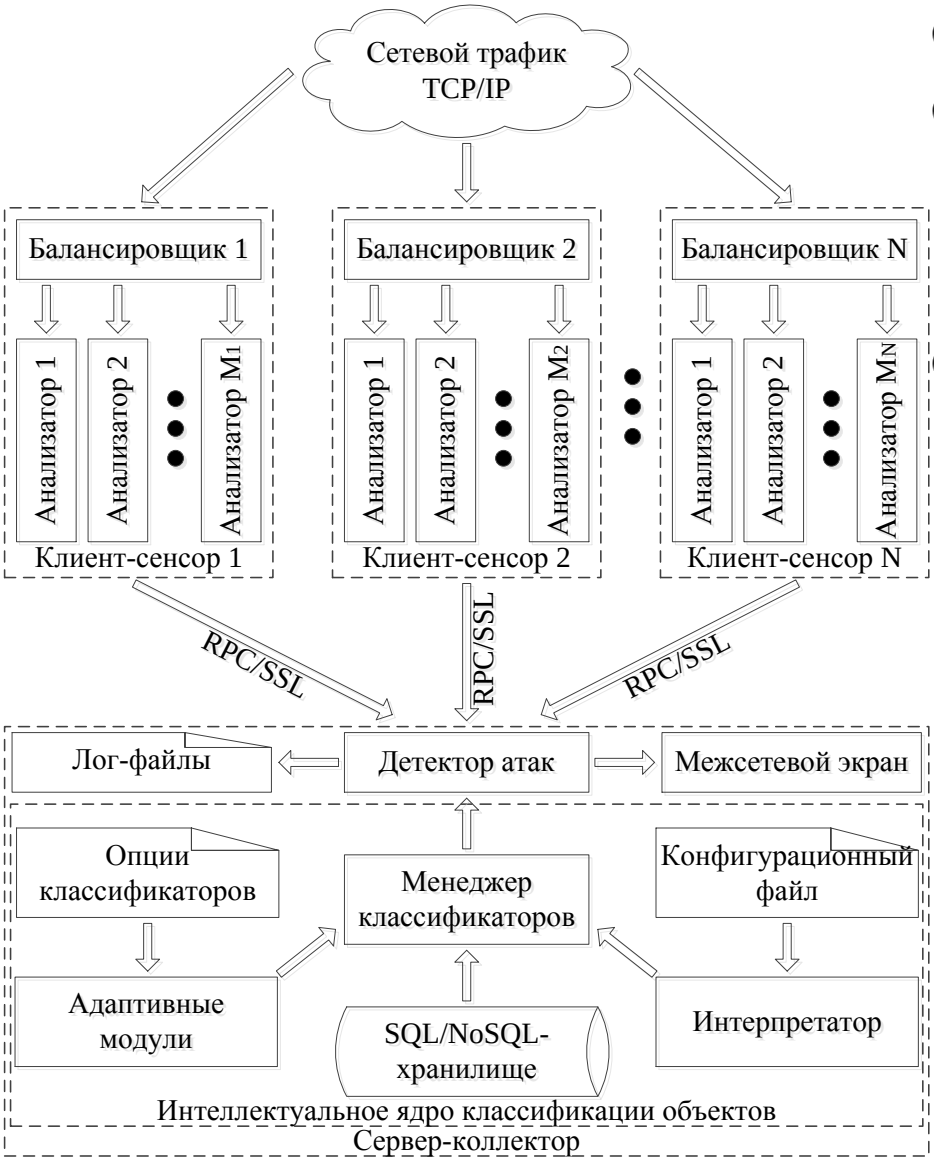
- Задание произвольной вложенности классификаторов друг в друга с указанием правил формирования входных данных
- „Ленивое“ подключение классификаторов благодаря наличию алгоритма каскадного обучения узлов
- Гибкое объединение детекторов при помощи различных низкоуровневых схем их комбинирования и агрегирующих композиций

Результаты

Результат 4

Архитектура и программная реализация распределенной СОА, построенной на основе гибридизации методов ВИ и сигнатурного анализа

Архитектура распределенной СОА



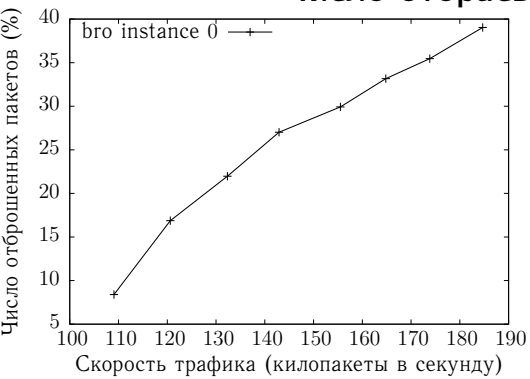
- Клиент-серверная архитектура
- Сенсоры
  - ✓ **Балансировщик трафика**
  - ✓ **Анализаторы трафика**
- Коллектор
  - ✓ **Интерпретатор**
  - ✓ **Адаптивные модули (so-библиотеки)**
  - ✓ **Менеджер классификаторов**
  - ✓ Детектор атак
  - ✓ Межсетевой экран (iptables)
  - ✓ SQL/NoSQL-хранилище (MySQL, MongoDB, CSV)
- Канал на основе RPC/SSL

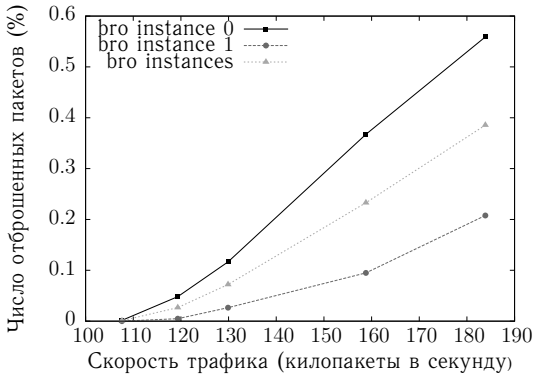
Балансировщик трафика

1 выходной интерфейс

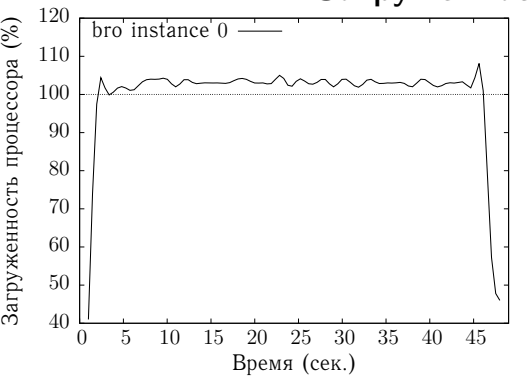
2 выходных интерфейса

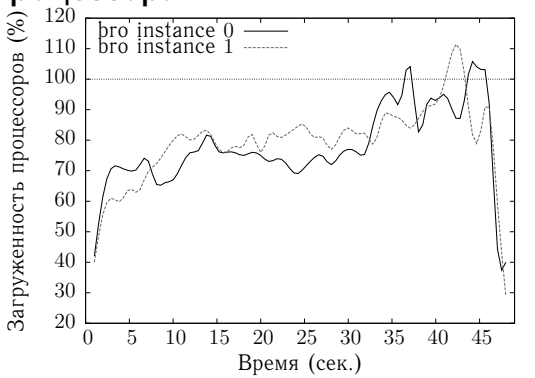
Число отбрасываемых пакетов



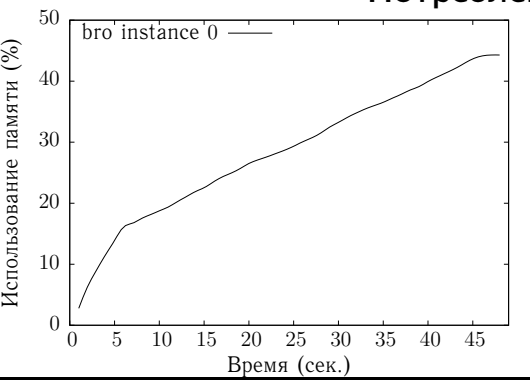


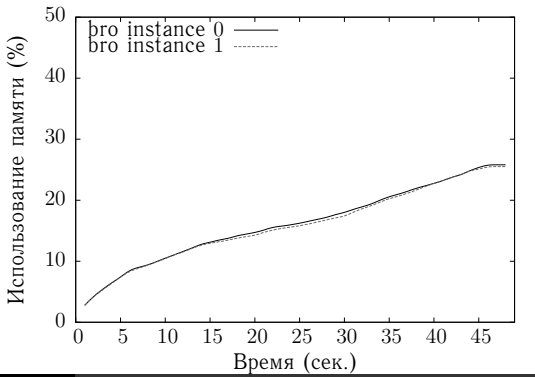
Загруженность процессора

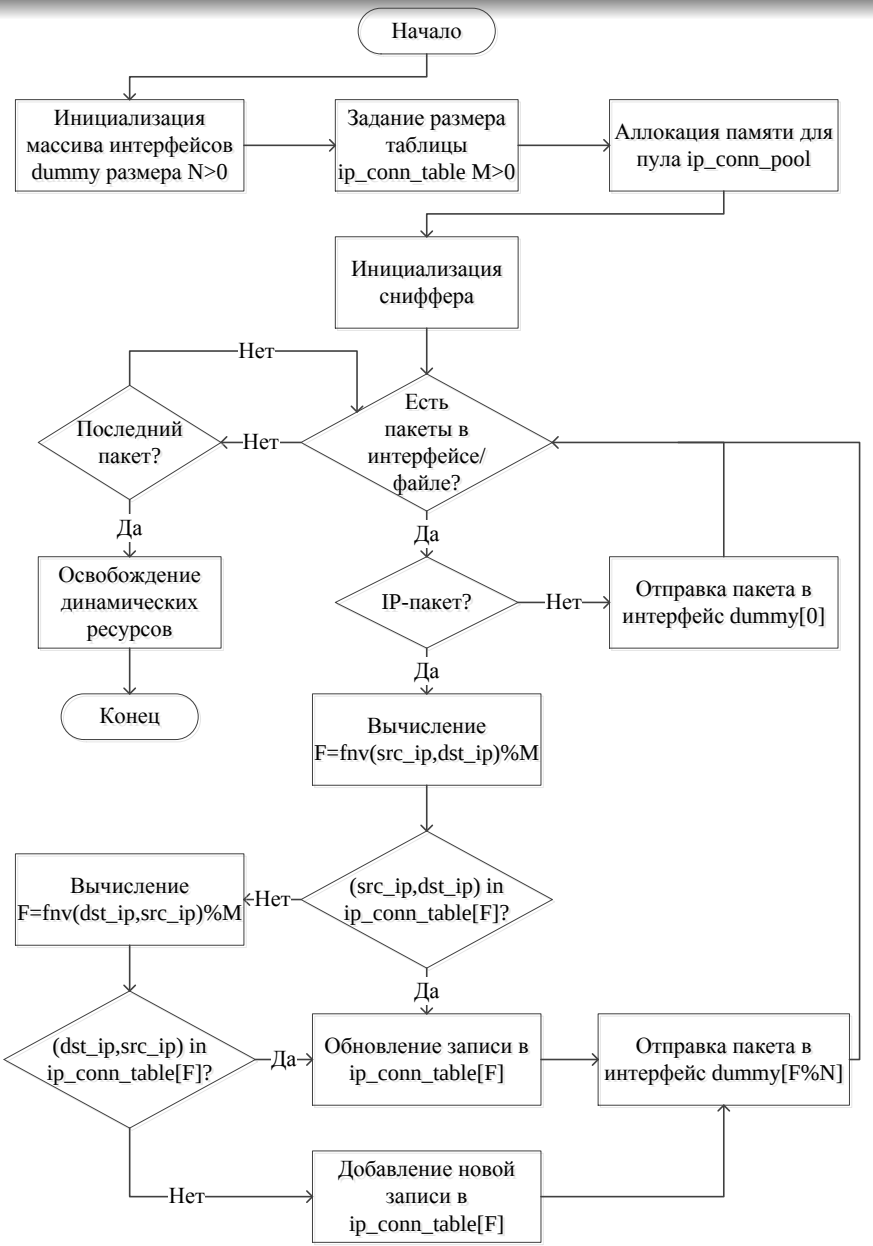




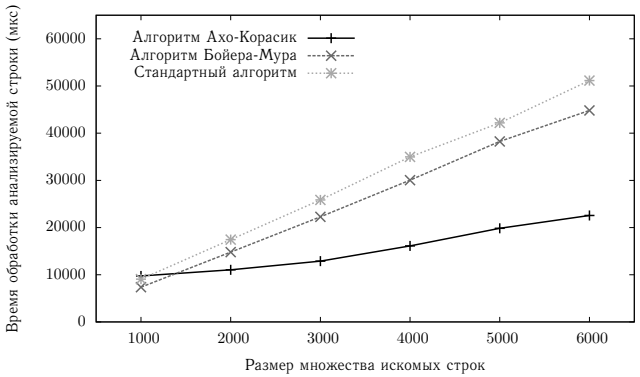
Потребление памяти

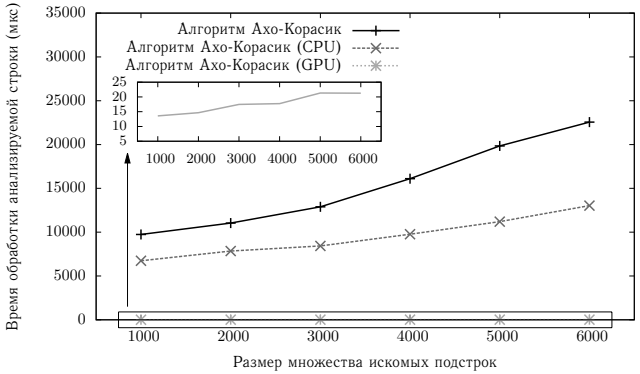


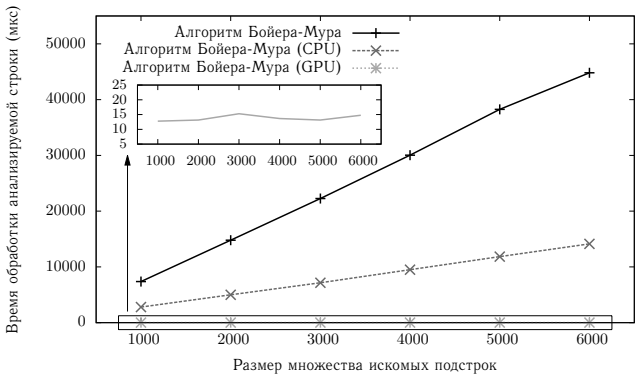




Анализатор трафика







- Поддержка событийно-ориентированного механизма обработки сетевых соединений (32 события)
- Реализация алгоритмов IP-дефрагментации и TCP-реассемблирования, свойственных сетевому стеку ОС Linux (RFC 791, 3128, 793, 7413, 1858, 1071, ...)
- Параллельные модификации алгоритмов поиска шаблонных подстрок в сигнатурных правилах СОА (технологии OpenMP, CUDA)
- 106 сетевых параметров (продолжительность соединения, служба, интенсивность отправки пакетов, число активных соединений, признак изменения масштабирования TCP-окна, состояние соединения, ...)
- Алгоритм преаллокации памяти для хранения fnv-хешированных записей о соединениях
- Наличие API для обнаружения атак со скрытием и со вставкой (Ptacek & Newsham)

Интерпретатор и адаптивные модули

```
classifier_tree: {
  vars: {
    in_dim: {
      "33" }
    out_dim: {
      "6" } }
  classifier: {
    name: {
      "meta_classifier" }
    identifier: {
      "7" }
    so_library: {
      "libmeta_classifier.so" }
    prf_file: {
      "meta_classifier_results.prf" }
    sls_file: {
      "meta_classifier_struct.sls" }
    opt_file: {
      "meta_classifier_options.opt" }
    input_dim: {
      "7" }
    output_dim: {
      "$out_dim" }
    input_data: {
      "concat(sum(idx#4,idx#5),idx#3)" }
    output_data: {
      type: {
        "array" } }
    nested_classifiers: {
      classifier:{
        ... } } } }
```

- Интерпретатор
  - ✓ КС грамматика
  - ✓ Левосторонняя рекурсия
  - ✓ Построение дерева классификаторов
  - ✓ Поддержка операторов условного ветвления, векторной конкатенации, ...
- Адаптивные модули
  - ✓ 20 плагинов
  - ✓ 5 АЯВУ (C, C++, Perl, Python, R)
  - ✓ Маршалинг основных типов данных C в скриптовые объекты и обратно
  - ✓ Динамическое встраивание плагинов в ядро COA без необходимости предлинковки

Менеджер классификаторов

- загрузка классификаторов из плагинов
- вызов функций, определенных в плагинах
- иерархический обход дерева классификаторов

Алгоритм каскадного обучения

1 функция f_1 конвертирования дерева классификаторов cls_tree в однонаправленный список dir_list
2 начало блока
3 dir_list :=создать пустой список
4 добавить корень $root\_node$ дерева cls_tree в качестве головного элемента в список dir_list
5 $list\_elem$:=получить головной элемент списка dir_list ($root\_node$)
6 пока $list\_elem$ не равен $NULL$ выполнять
7 если узел $list\_elem$ не является терминальным в дереве cls_tree тогда
8 для каждого дочернего узла $nested\_node$ элемента $list\_elem$ выполнять
9 добавить узел $nested\_node$ в качестве хвостового элемента в список dir_list
10 $list\_elem$:=получить следующий за $list\_elem$ элемент из списка dir_list
11 возвратить dir_list
12 функция f_2 каскадного обучения узла $node$ и его зависимостей в дереве классификаторов cls_tree
13 начало блока
14 если узел $node$ помечен как необученный тогда
15 если узел $node$ нетерминальный тогда
16 для каждого узла dep_node из списка зависимостей узла $node$ выполнять
17 вызвать функцию f_2 для узла dep_node
18 in_data :=сформировать входные векторы для узла $node$ согласно его дереву выражений
19 выполнить обучение классификатора, размещенного в узле $node$, на данных in_data
20 пометить узел $node$ как обученный
21 функция f_3 обхода дерева классификаторов cls_tree в ширину при помощи списка dir_list
22 начало блока
23 $node$:=получить головной элемент списка dir_list
24 пока $node$ не равен $NULL$ выполнять
25 вызвать функцию f_2 для узла $node$
26 $node$:=получить следующий за $node$ элемент из списка dir_list
27 cls_tree :=вызвать интерпретатор для заданного пользователем конфигурационного файла
28 dir_list :=вызвать функцию f_1 для дерева классификаторовcls_tree
29 вызвать функцию f_3 для дерева классификаторов cls_tree и списка dir_list

Отличия от других архитектур

- Реализация оригинальной методики иерархической гибридизации бинарных классификаторов для обнаружения аномальных сетевых соединений
- Возможность „горячей“ вставки нового исполняемого кода за счет загрузки плагинов, представленных в виде so-библиотек и встраиваемых динамически в ядро COA
- Наличие интерпретатора с возможностью написания собственных сценариев для задания структуры и правил функционирования адаптивных классификаторов и отдельных детекторов

Эксперименты (оценка СОА)

- Вычислительная адаптивность
- Способность обобщения
- Устойчивость к наличию шумов
- Быстродействие системы

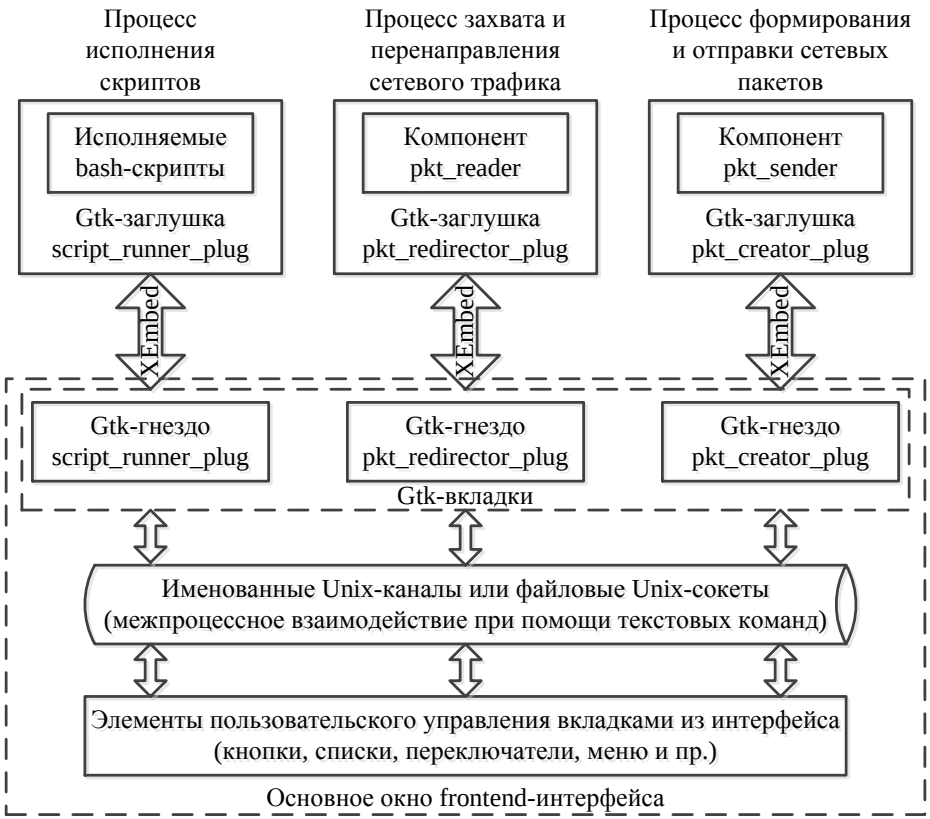
Эксперименты

Эксперимент 1

Вычислительная адаптивность

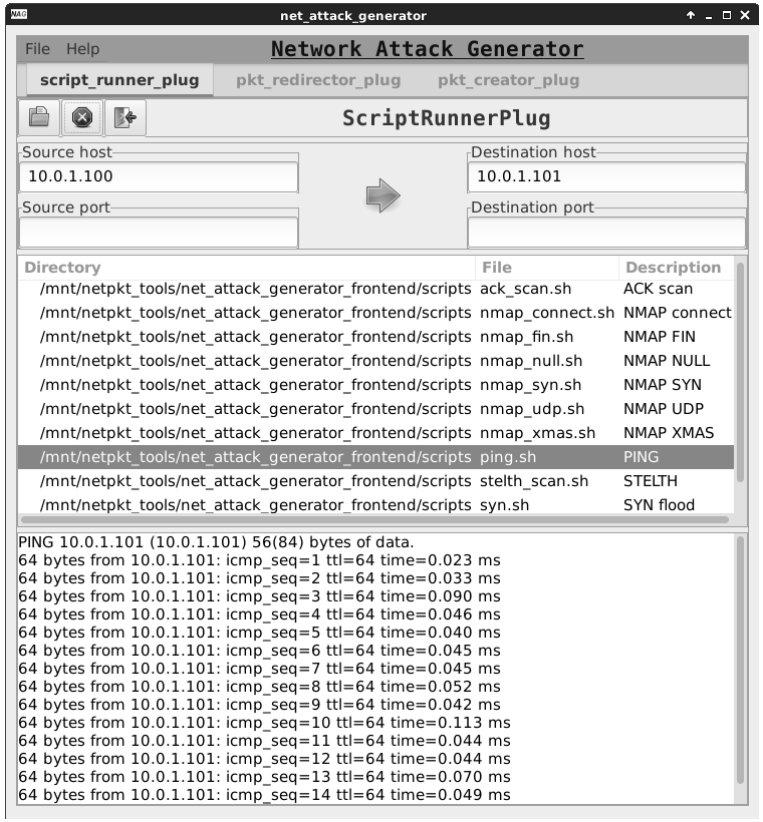
Генератор атак и набор данных

Архитектура генератора атак



- Взаимодействие процессов с основным окном на системном уровне через протокол XEmbed
- Управление процессами на пользовательском уровне через именованные Unix-каналы или файловые Unix-сокеты

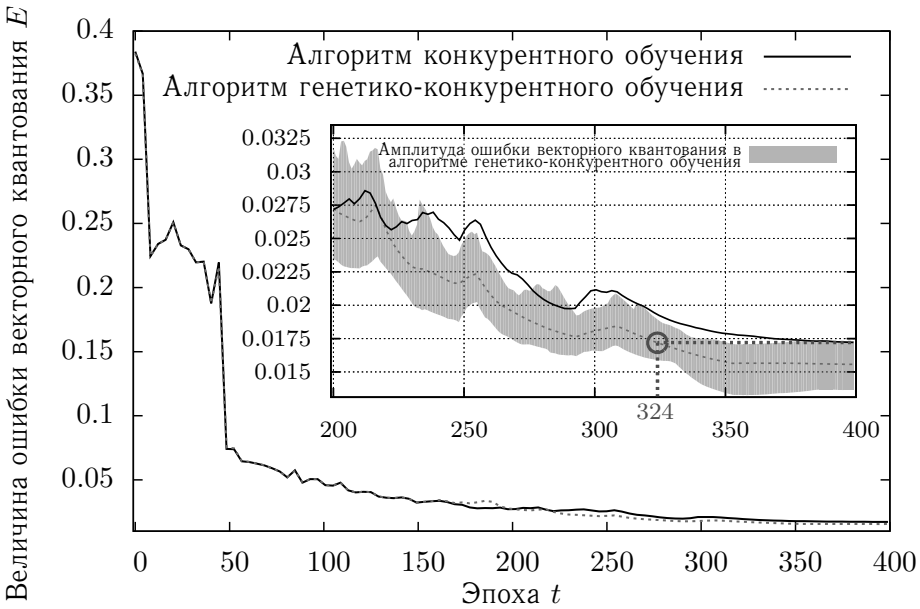
Интерфейс генератора атак

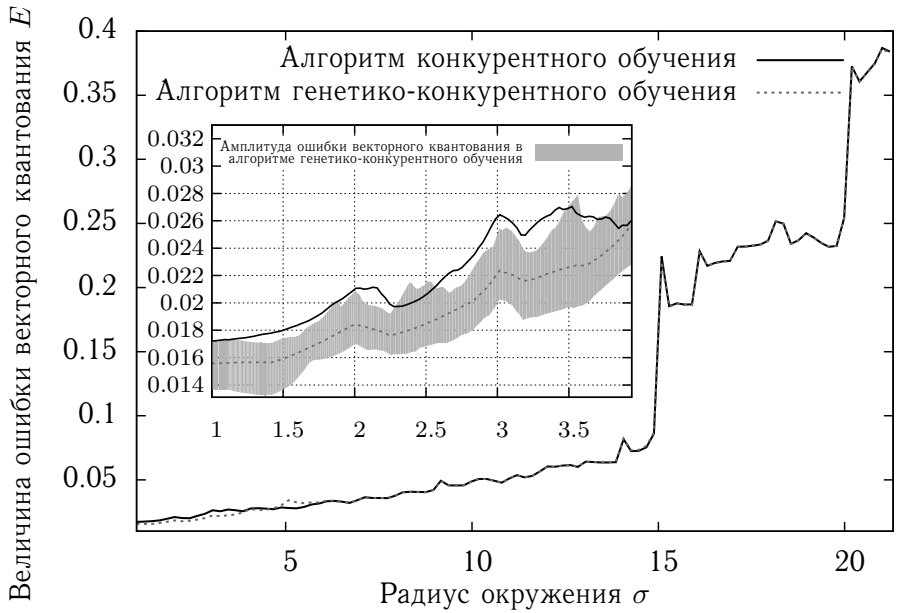


- Отказоустойчивое исполнение загруженных внутри вкладок процессов
- Возможность независимого запуска и прерывания скриптов (nmap, hping3, ...)
- Набор данных с нормальными сетевыми соединениями — MAWILab Data Set 2015/07/01

Вычислительная адаптивность (класс normal)

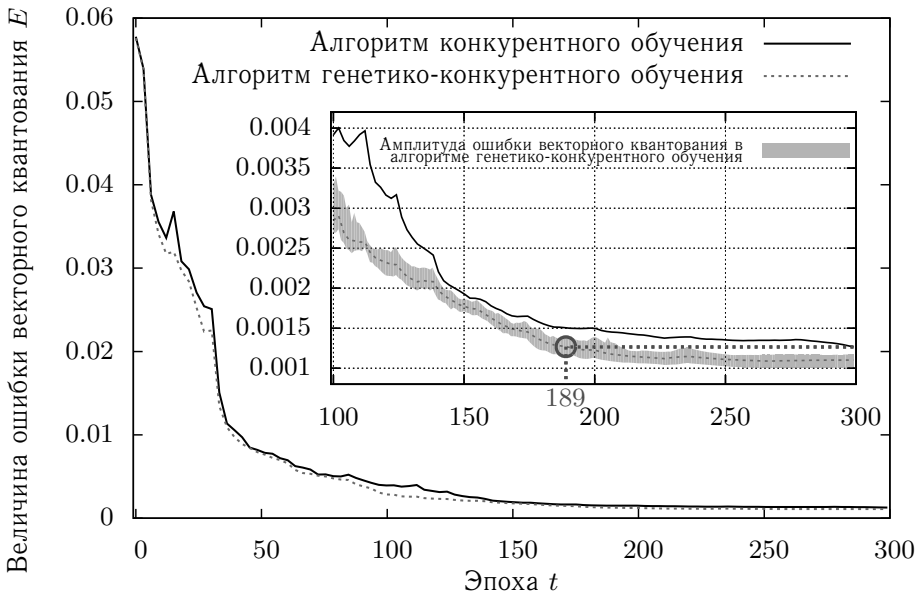
- Начальный параметр скорости обучения: $\kappa_0 = 0.7$
- Мощность обучающей выборки: $M = 900$
- Размерность входного вектора: $n = 20$
- Размер выходной решетки: $I \times J = 15 \times 15$
- Число эпох обучения: $T = 400$
- Начальный радиус окружения: $\sigma_0 = 21.21$
- **Сокращение числа эпох обучения на 74**

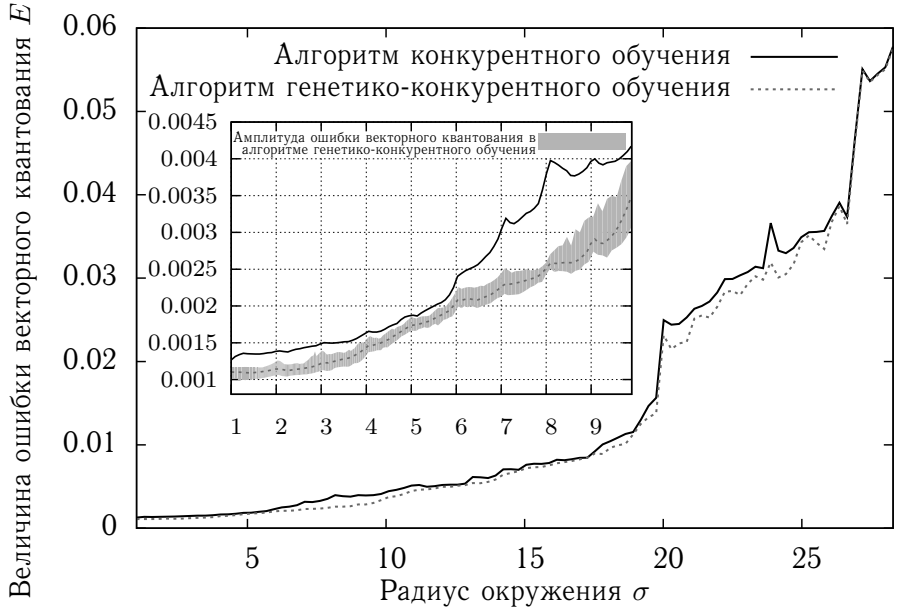




Вычислительная адаптивность (класс scan)

- Начальный параметр скорости обучения: $\kappa_0 = 0.3$
- Мощность обучающей выборки: $M = 1000$
- Размерность входного вектора: $n = 10$
- Размер выходной решетки: $I \times J = 20 \times 20$
- Число эпох обучения: $T = 300$
- Начальный радиус окружения: $\sigma_0 = 28.28$
- **Сокращение числа эпох обучения на 110**





Эксперименты

Эксперимент 2

Способность обобщения

Пороги активации

$$h_{ij} = \begin{cases} h_{ij}^- - \beta \cdot \left(h_{ij}^- - h_{ij}^+ \right), \text{ если } \forall \boldsymbol{x_k} \in C_0 \bigcap \mathcal{X}_{\mathcal{C}}^{(LS)} \; d_{ijk}^{-1} < h_{ij}^- \\ h_{ij}^-, \text{ если } \exists \boldsymbol{x_k} \in C_0 \bigcap \mathcal{X}_{\mathcal{C}}^{(LS)} \; d_{ijk}^{-1} \geqslant h_{ij}^-, \end{cases}$$

$$h_{ij}^- = \min_{\boldsymbol{x_k} \in C_l \bigcap \mathcal{X}_{\mathcal{C}}^{(LS)}} \left\{ d_{ijk}^{-1} \,|(i,j) = F_{IJ}(\boldsymbol{x_k}) \right\},$$

$$h_{ij}^+ = \max_{\boldsymbol{x_k} \in C_0 \bigcap \mathcal{X}_{\mathcal{C}}^{(LS)}} \left\{ d_{ijk}^{-1} | (i,j) = F_{IJ}(\boldsymbol{x_k}) \right\},$$

$$d_{ijk} = D(\boldsymbol{x_k}, \boldsymbol{w_{ij}}) = \sqrt{\sum_{l=1}^{n} (x_{kl} - w_{ijl})^2}, F_{IJ}(\boldsymbol{x_k}) = \arg \min_{\substack{1 \leqslant i \leqslant I \\ 1 \leqslant j \leqslant J}} d_{ijk},$$

C_0 — класс нормальных сетевых соединений,

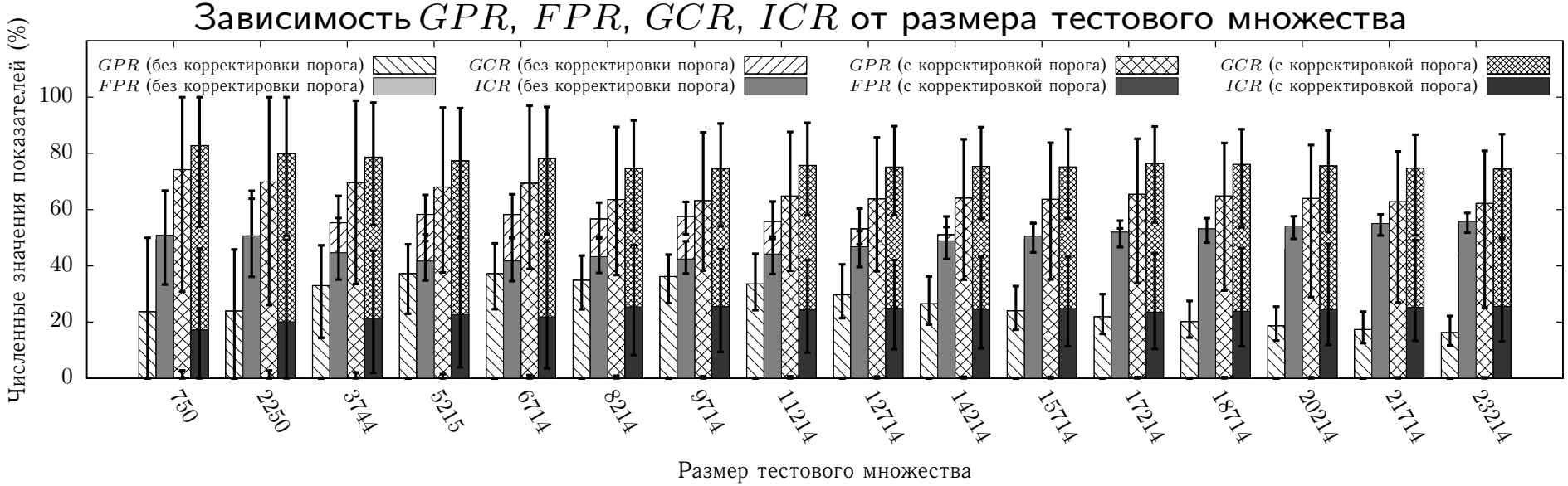
C_l — l-ый класс аномальных сетевых соединений$(l = 1, 2)$.

- $\beta = \frac{1}{2}$
- $\beta = \arg \max_{\beta = 0, 0.01, \ldots, 1} GPR - FPR + GCR - ICR$

Способность обобщения ($\beta = 1/2$)

- $\kappa_0 = 0.1$
- $M = 4500$
- $n = 25$
- $T = 500$
- $I \times J$:10–15$\times$10–15
- 6 детекторов
- $\bar{M}^*$=750, . . . ,23214

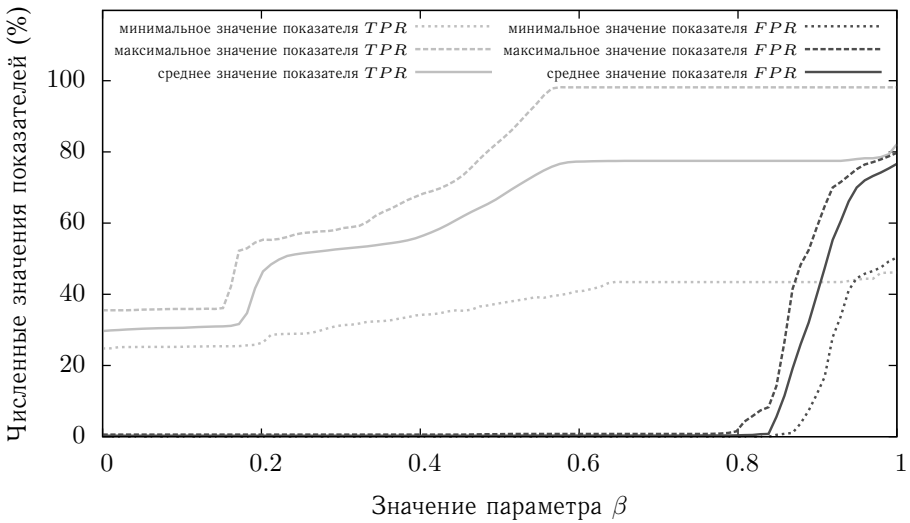
Порог	Средние значения показателей (%)			
	GPR	FPR	GCR	ICR
Размер тестового множества: M^*=750				
h_{ij}^-	23.684	0	49.123	50.877
h_{ij}	74.208	0.048	82.789	17.211
Размер тестового множества: $\bar{M}^*$=23214				
h_{ij}^-	16.243	0	44.206	55.794
h_{ij}	62.222	0.007	74.409	25.591

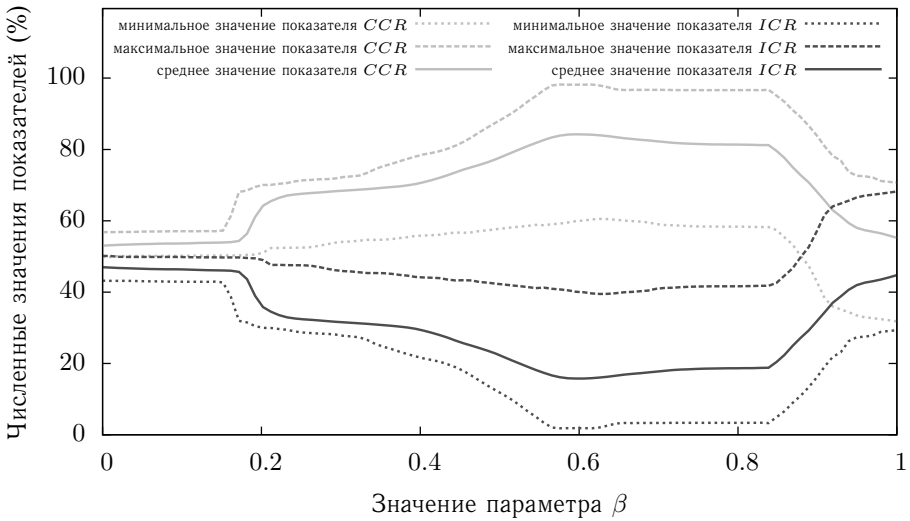


Способность обобщения ($\beta = 3/5$)

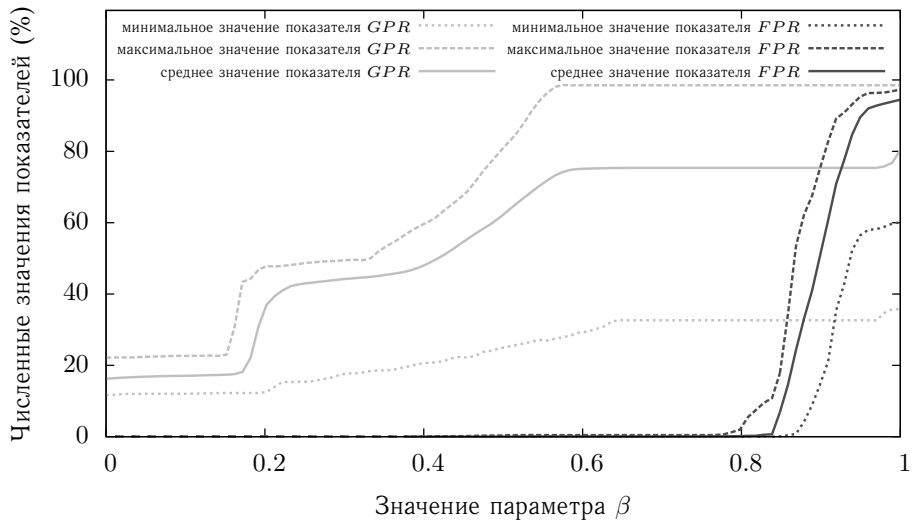
$$\beta_{opt} = \arg \max_{\beta=0,0.01,\ldots,1} GPR - FPR + GCR - ICR = 0.6$$

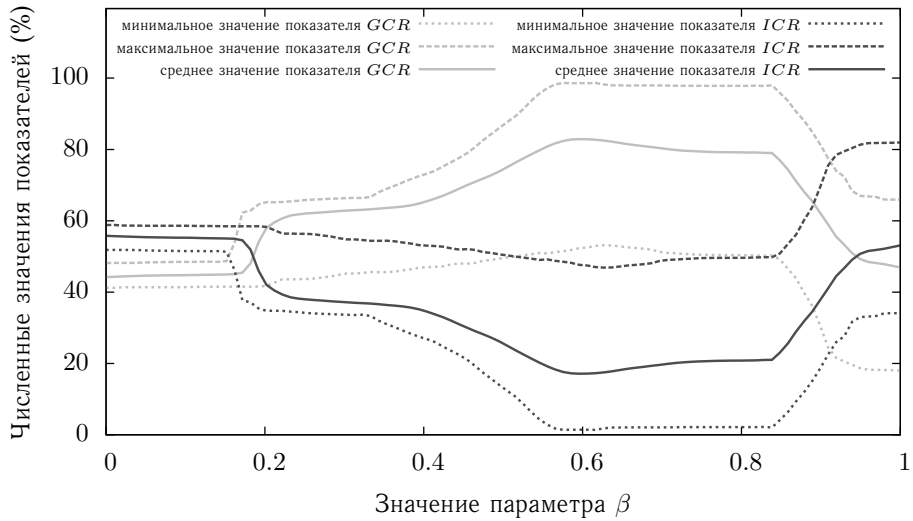
Зависимость TPR, FPR, CCR, ICR от β





Зависимость GPR, FPR, GCR, ICR от β





Эксперименты

Эксперимент 3

Устойчивость к наличию шумов

Набор данных

5-блочная кросс-валидация

- Разбиение множества $\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)}$:

$$\bar{\mathcal{X}}_{\mathcal{C}}^{(TS)} = \bigvee_{k=1}^{5} \overset{(k)}{\bar{\mathcal{X}}}{}_{\mathcal{C}}^{(TS)}, \quad \#\overset{(1)}{\bar{\mathcal{X}}}{}_{\{C_l\}}^{(TS)} \approx \ldots \approx \overset{(5)}{\bar{\mathcal{X}}}{}_{\{C_l\}}^{(TS)}$$

$(l=0,\ldots,6)$

- Отношение размера обучающей выборки к контрольной: $3:2$
- Число процессов обучения и тестирования базовых классификаторов: $3 \times C_5^3 = 30$ раз
- Вычисление показателей на уникальных элементах, не встречавшихся при обучении:

$$\min_{p=1,2,3} \frac{1}{10} \cdot \sum_{1 \leqslant d < e \leqslant 5} GPR_{(de)_p},$$

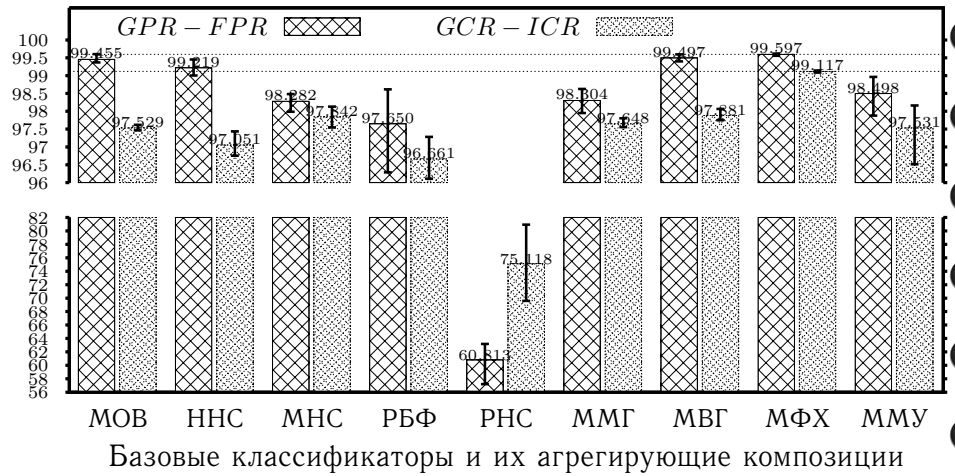
$$\max_{p=1,2,3} \frac{1}{10} \cdot \sum_{1 \leqslant d < e \leqslant 5} GPR_{(de)_p},$$

$$\frac{1}{3} \cdot \sum_{p=1}^{3} \frac{1}{10} \cdot \sum_{1 \leqslant d < e \leqslant 5} GPR_{(de)_p}$$

- Набор данных DARPA 1998
- 101113 записей
- 53733 уникальных записи
- 5 базовых классификаторов: машины опорных векторов (МОВ), нейронечеткие сети (ННС), многослойные нейронные сети (МНС), нейронные сети с радиальными базисными функциями (РБФ), рекуррентные нейронные сети Джордана (РНС)
- 2 ошибочных классификатора: случайный классификатор (СК) и ложный классификатор (ЛК)
- 6 классов аномальных сетевых соединений (probing и DoS)
- 4 типа низкоуровневых схем объединения детекторов
- 4 типа агрегирующих композиций

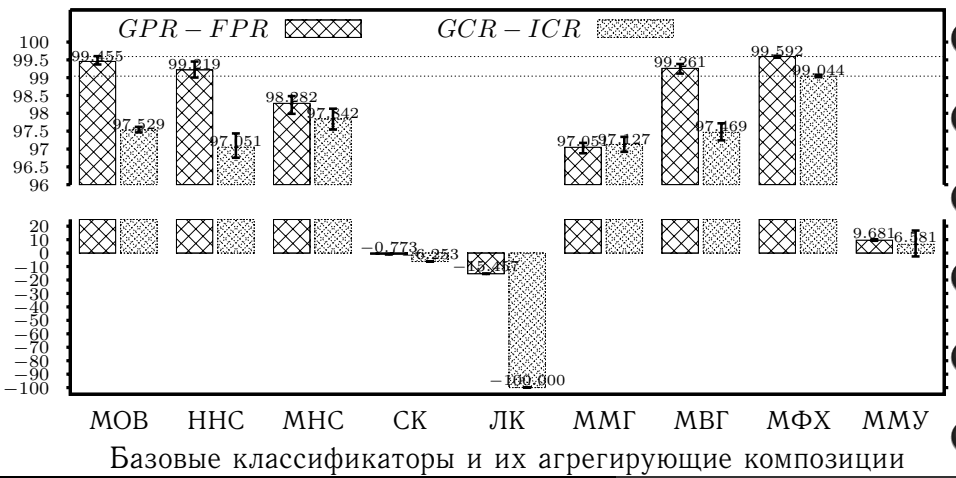
Устойчивость к наличию шумов (one-vs-all)

Схема комбинирования детекторов one-vs-all (90)



- $GPR \Uparrow$ на 0.021% (МФХ vs ННС)
- $FPR \Downarrow$ на 0.072% (ММГ vs МОВ)
- $GCR \Uparrow$ на 0.707% (МФХ vs МНС)
- $ICR \Downarrow$ на 0.569% (МФХ vs МНС)
- $GPR–FPR \Uparrow$ на 0.142% (МФХ vs МОВ)
- $GCR–ICR \Uparrow$ на 1.276% (МФХ vs МНС)

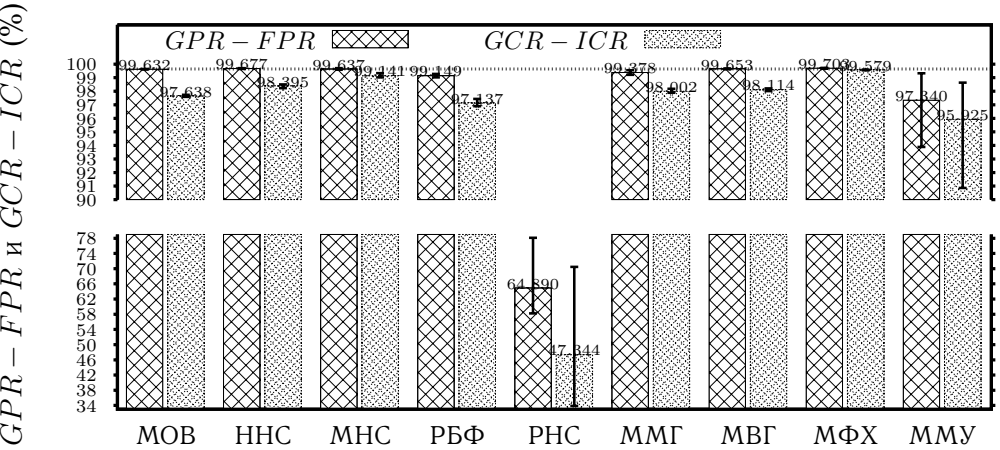
Схема комбинирования детекторов one-vs-all (с СК и ЛК)



- $GPR \Uparrow$ на 0.017% (МФХ vs ННС)
- $FPR \Downarrow$ на 0.005% (МФХ vs МОВ)
- $GCR \Uparrow$ на 0.67% (МФХ vs МНС)
- $ICR \Downarrow$ на 0.533% (МФХ vs МНС)
- $GPR–FPR \Uparrow$ на 0.137% (МФХ vs МОВ)
- $GCR–ICR \Uparrow$ на 1.203% (МФХ vs МНС)

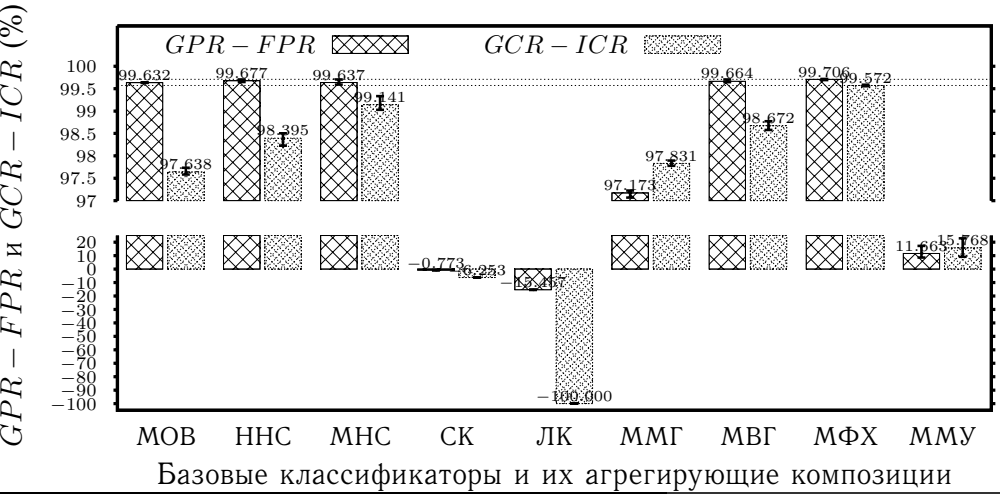
Устойчивость к наличию шумов (one-vs-one)

Схема комбинирования детекторов one-vs-one (315)



- $GPR \Uparrow$ на 0.003% (МФХ vs ННС)
- $FPR \Downarrow$ на 0.023% (МФХ vs ННС)
- $GCR \Uparrow$ на 0.217% (МФХ vs МНС)
- $ICR \Downarrow$ на 0.221% (МФХ vs МНС)
- $GPR–FPR \Uparrow$ на 0.025% (МФХ vs ННС)
- $GCR–ICR \Uparrow$ на 0.438% (МФХ vs МНС)

Схема комбинирования детекторов one-vs-one (с СК и ЛК)



- $GPR \Uparrow$ на 0.003% (МФХ vs ННС)
- $FPR \Downarrow$ на 0.056% (ММГ vs ННС)
- $GCR \Uparrow$ на 0.213% (МФХ vs МНС)
- $ICR \Downarrow$ на 0.218% (МФХ vs МНС)
- $GPR–FPR \Uparrow$ на 0.029% (МФХ vs ННС)
- $GCR–ICR \Uparrow$ на 0.431% (МФХ vs МНС)

Эксперименты

Эксперимент 4

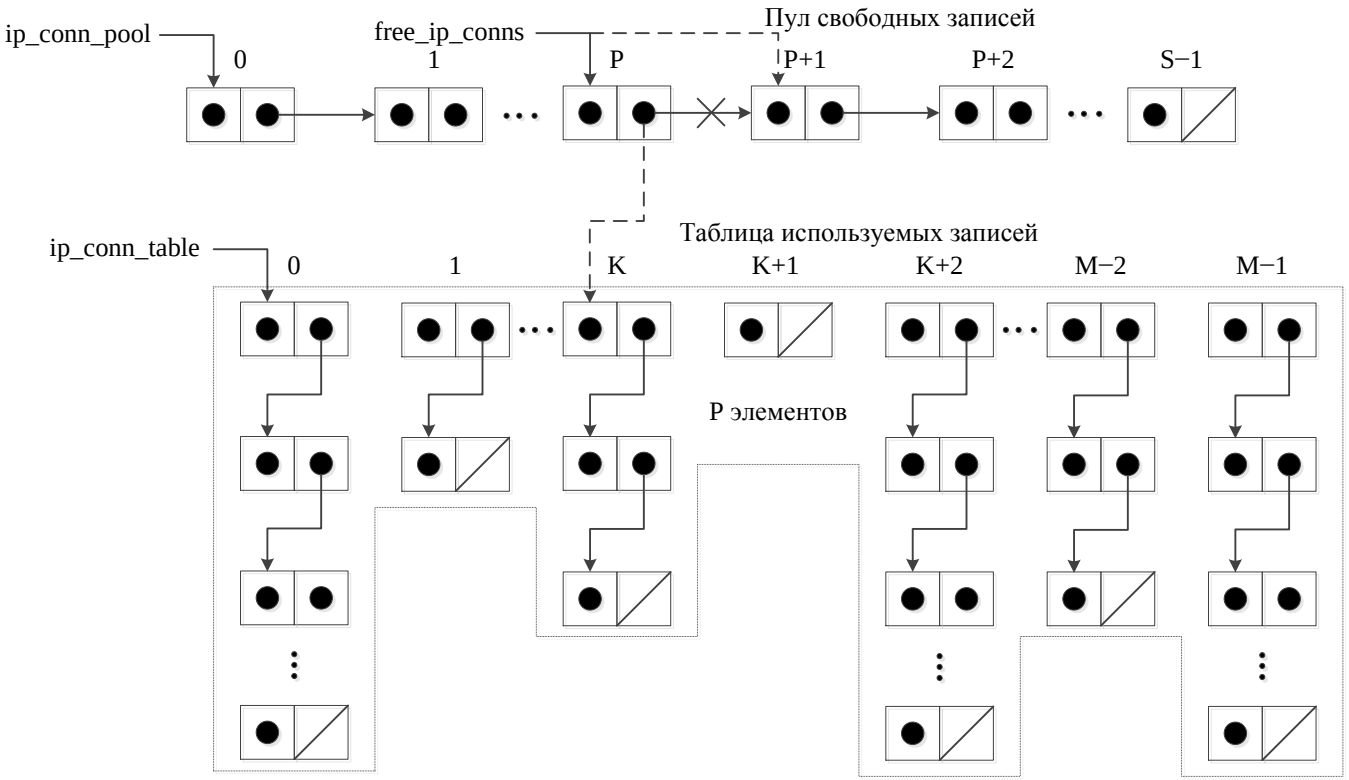
Быстродействие системы

Сравниваемые характеристики СОА

- Запуск каждой СОА — 100 раз
- Размер тестового pcap-файла — 10^7 пакетных записей, 709.047 MB
- Сравниваемые характеристики:
  - ✓ $T^{(real)}$ — общее время функционирования
  - ✓ $T^{(proc)}$ — время обработки пакетов
  - ✓ $V^{(pkts)}$ — скорость обработки килопакетов данных
  - ✓ $V^{(data)}$ — скорость обработки килобайтов данных
  - ✓ $L^{(CPU)}$ — загрузка CPU
  - ✓ $C^{(virt)}$ — размер потребляемой виртуальной памяти
  - ✓ $C^{(resd)}$ — размер потребляемой резидентной памяти
  - ✓ $C^{(real)}$ — размер потребляемой реальной памяти
- Сравниваемые СОА:
  - ✓ Snort 2.9.8.2
  - ✓ Suricata 3.0.1
  - ✓ Bro 2.4.1

netpcap_sensor

Механизм выделения и хранения памяти



Настройки сенсора:

- Размер списка ip_conn_pool: $MAX\_IP\_POOL\_SIZE{=}S{=}300000$
- Размер таблицы ip_conn_table: $MAX\_IP\_TABLE\_SIZE{=}M{=}100000$
- Хранение 5-секундной истории захваченных пакетов
- Хеширование 13-байтового ключа сетевого соединения

Быстродействие и ресурсопотребление

Характеристика		COA				
		netpcap_sensor	Snort	Suricata (single)	Suricata (autofp)	Bro
$T^{(real)}$ (sec.)	min	29.881	63.749	48.264	86.659	177.691
	max	31.094	72.340	60.049	120.151	195.778
	avg	30.657	65.682	53.121	102.743	184.730
$T^{(proc)}$ (sec.)	min	29.304	49.112	41.800	80.138	154.167
	max	30.512	55.975	53.569	113.668	175.034
	avg	30.071	51.340	46.618	96.192	160.801
$V^{(pkts)}$ (Kpkts/sec.)	min	320.063	174.465	182.300	85.914	55.793
	max	333.257	198.844	233.627	121.860	63.345
	avg	324.756	190.215	209.483	101.522	60.731
$V^{(data)}$ (Kbytes/sec.)	min	18 675.303	10 179.843	10 637.004	5012.965	3255.455
	max	19 445.207	11 602.359	13 631.906	7110.405	3696.090
	avg	18 949.140	11 098.823	12 223.090	5923.697	3543.591
$L^{(CPU)}$ (%)	min	52.100	0.000	0.000	0.000	6.600
	max	100.000	100.000	179.500	265.700	212.600
	avg	83.622	88.766	141.909	193.261	115.972
$C^{(virt)}$ (Mbytes)	min	176.098	77.785	0.000	0.000	0.000
	max	256.121	800.371	675.496	1026.121	1528.699
	avg	233.353	705.731	617.404	914.086	972.210
$C^{(resd)}$ (Mbytes)	min	144.262	14.664	0.000	0.000	0.000
	max	224.469	527.496	332.332	360.129	991.828
	avg	201.585	422.200	303.798	318.679	634.394
$C^{(real)}$ (Mbytes)	min	141.203	10.758	0.000	0.000	0.000
	max	221.234	522.137	325.188	352.902	982.750
	avg	198.452	416.870	296.685	311.556	625.444

Заключение по результатам

Основные результаты:

1. Разработана **модель искусственной иммунной системы на базе эволюционного подхода для классификации сетевых соединений**. Модель характеризуется наличием двухуровневой процедуры обучения и тестирования и позволяет учитывать динамическую природу сетевого трафика посредством переобучения иммунных детекторов как с течением времени, так и в ответ на выявленные в сетевом трафике аномалии.
2. Разработан **алгоритм генетико-конкурентного обучения сети Кохонена для обнаружения аномальных сетевых соединений**. Отличительной особенностью алгоритма является наличие процесса имитации эволюции „мертвых“ нейронов, который достигается за счет использования нескольких стратегий генетической оптимизации весовых коэффициентов сети Кохонена.
3. Разработана **методика иерархической гибридизации бинарных классификаторов для обнаружения аномальных сетевых соединений**. Методика предоставляет возможность строить многоуровневые схемы с произвольной вложенностью классификаторов друг в друга и их „ленивым“ подключением в процессе анализа входного вектора.
4. Разработана **архитектура распределенной СОА, построенной на основе гибридизации методов ВИ и сигнатурного анализа, и выполнена ее программная реализация**. СОА обеспечивает возможность „горячей“ вставки исполняемого кода за счет загрузки плагинов, представленных в виде so-библиотек и встраиваемых динамически в ядро СОА.

Заключение по экспериментам

Результаты проведенных экспериментов доказывают, что разработанная СОА удовлетворяет требованиям, предъявляемым к вычислительно интеллектуальным системам (Bezdek J.C., Fogel D.B., Eberhart R.C.), а именно она:

1. является адаптивной в терминах ВИ
2. обладает способностью обобщения
3. устойчива к наличию шумов
4. характеризуется высокой скоростью функционирования

Достижение поставленной цели

Научный результат	**Целевое назначение**	**Выполнение**
Результат 1	•Повышение обоснованности классификации соединений	•Выполнено
Результат 2	•Повышение своевременности процесса обучения	•Выполнено
Результат 3	•Повышение обоснованности классификации соединений •Повышение своевременности процесса обучения	•Выполнено •Выполнено
Результат 4	•Повышение своевременности процесса анализа пакетов •Снижение объема затраченных системных ресурсов	•Выполнено •Выполнено

Поставленная цель, заключающаяся в повышении эффективности функционирования СОА, **достигнута**.

Публикации (ВАК)

- *Браницкий А. А., Котенко И. В.* — Обнаружение сетевых атак на основе комплексирования нейронных, иммунных и нейронечетких классификаторов. — // Информационно-управляющие системы. — 2015. — 4 (77). — С. 69—77.
- *Браницкий А. А., Котенко И. В.* — Построение нейросетевой и иммуноклеточной системы обнаружения вторжений. — // Проблемы информационной безопасности. Компьютерные системы. — 2015. — № 4. — С. 23—27.
- *Браницкий А. А., Котенко И. В.* — Анализ и классификация методов обнаружения сетевых атак. — // Труды СПИИРАН. — 2016. — 2 (45). — С. 207—244.
- *Браницкий А. А.* — Иерархическая гибридизация бинарных классификаторов для выявления аномальных сетевых соединений. — // Труды СПИИРАН. — 2017. — 3 (52). — С. 204—233.
- *Браницкий А. А., Котенко И. В.* — Открытые программные средства для обнаружения и предотвращения сетевых атак. — // Защита Информации. Инсайд. — 2017. — 2 (74). — С. 40—47.
- *Браницкий А. А., Котенко И. В.* — Открытые программные средства для обнаружения и предотвращения сетевых атак (окончание). — // Защита Информации. Инсайд. — 2017. — 3 (75). — С. 58—66.

Публикации (прочие)

Тимофеев А. В., *Браницкий А. А.* — Исследование и моделирование нейросетевого метода обнаружения и классификации сетевых атак. — // International Journal Information Technologies & Knowledge. — 2012. — Т. 6, № 3. — С. 257—265.

Branitskiy A., *Kotenko I.* — Network attack detection based on combination of neural, immune and neuro-fuzzy classifiers. — // In Proceedings of the 18th IEEE International Conference on Computational Science and Engineering (IEEE CSE2015). — IEEE. Окт. 2015. — 152—159 (**WoS, Scopus, Best paper award**).

Branitskiy A., *Kotenko I.* — Hybridization of computational intelligence methods for attack detection in computer networks. — // Journal of Computational Science. — 2017. — Т. 23. — 145—156 (**WoS, Scopus, Q2**).

Branitskiy A., *Kotenko I.* — Network Anomaly Detection Based on an Ensemble of Adaptive Binary Classifiers. — // In Proceedings of International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security. — Springer. 2017. — 143—157 (**WoS, Scopus**).

Свидетельства о регистрации программ для ЭВМ

Браницкий А. А., *Котенко И. В.* — Адаптивная система обнаружения атак на основе гибридизации методов вычислительного интеллекта. — Свидетельство о государственной регистрации программы для ЭВМ № 2015662189. Зарегистрировано в Реестре программ для ЭВМ 18.11.2015.

Браницкий А. А., *Котенко И. В.* — Компонент классификации аномальных сетевых соединений на основе искусственных иммунных систем. — Свидетельство о государственной регистрации программы для ЭВМ № 2016663476. Зарегистрировано в Реестре программ для ЭВМ 08.12.2016.

Браницкий А. А., *Котенко И. В.*, *Саенко И. Б.* — Frontend-интерфейс генератора сетевых атак. — Свидетельство о государственной регистрации программы для ЭВМ № 2017660184. Зарегистрировано в Реестре программ для ЭВМ 19.09.2017.