# КОМПЬЮТЕРНАЯ ИНЖЕНЕРИЯ И ТЕХНИЧЕСКАЯ ДИАГНОСТИКА

УДК 004.89

## НЕЙРОСЕТЕВОЙ ЭВРИСТИЧЕСКИЙ АНАЛИЗАТОР ВРЕДОНОСНЫХ ПРОГРАММ С ИММУННЫМ ОБУЧЕНИЕМ

*КОРАБЛЁВ Н.М., КУШНАРЁВ М.В., УЖВИЙ Д.П.*

Предлагается эвристический анализатор вредоносных программ, основным компонентом которого является искусственная нейронная сеть в виде многослойного персептрона с иммунным обучением. Для решения задачи обучения используется модель кодирования настраиваемых параметров в виде адаптивного структурированного мультиантитела, что позволяет уменьшить количество нейронов в скрытом слое и устранить, таким образом, избыточность нейронной сети. Проведено моделирование работы эвристического анализатора, показывающее эффективность предложенного подхода.

### 1. Введение

В настоящее время количество разнообразного вредоносного программного обеспечения растет, поэтому задачи обнаружения и предотвращения вторжений в компьютер не теряют свою актуальность. Существующие антивирусные продукты не могут обеспечить абсолютно надежную защиту компьютера. Это, в первую очередь, связано с тем, что применяемые в антивирусах принципы поиска не позволяют обнаруживать новые разновидности вредоносных программ до их изучения аналитиками и внесения дополнений и изменений в базы антивирусных программ [1, 2].

Основным способом выявления большинства компонентов вредоносных программ является сигнатурная проверка. Однако сигнатурный метод непригоден для защиты от новых и полиморфных вирусов, так как до тех пор, пока вирус не попал на анализ к экспертам, создать его сигнатуру невозможно. Существующие эвристические технологии, призванные помочь в определении новых модификаций вирусов, не дают должного уровня распознавания в связи с их слабой эффективностью при работе с зашифрованными объектами. Определять новые модификации таких вирусов с высокой долей вероятности еще до момента выхода соответствующего обновления антивирусных баз сегодня невозможно. Частично защитить компьютер от заражения новыми модификациями вирусов могут средства проактивной защиты антивирусных программ, однако и они имеют ограничения, так как настроены на распознавание общих типов вредоносного поведения.

К недостаткам существующих методов обнаружения вторжений также можно отнести уязвимость к новым атакам, низкую точность и скорость работы. Современные системы обнаружения вторжений плохо приспособлены к работе в реальном времени, в то время как возможность обрабатывать большой объем данных – это определяющий фактор использования таких систем. Указанные недостатки трудно устранить, используя только классические методы в области компьютерной безопасности. Поэтому появилась необходимость в разработке новых эвристических методов, которые смогли бы эффективно распознавать новые модификации вирусов. Основные требования, предъявляемые к таким методам, следующие:

1) высокий процент распознавания новых модификаций;

2) низкий коэффициент ложных срабатываний;

3) высокая скорость проверки одного файла;

4) небольшой размер сопутствующих обновлений;

5) небольшое потребление системных ресурсов;

6) возможность добавления в будущем новых методов обнаружения и анализа вирусов.

С появлением эвристических анализаторов (ЭА) нового поколения, использующих поведенческий анализ на основании данных, получаемых от эмуляторов, и их интеллектуального анализа, появилась возможность обнаруживать не только потенциально вредоносные коды без обращения к базам данных сигнатур, но и ранее не известные вирусы. Использование методов искусственного интеллекта позволяет создавать принципиально новые алгоритмы обнаружения вредоносных программ, повышающие уровень защищенности компьютерных систем. В этой связи в составе ЭА в настоящее время активно используются искусственные нейронные сети (ИНС), генетические алгоритмы (ГА), искусственные иммунные системы (ИИС), мультиагентные системы (МАС) [3-8], с помощью которых можно эффективно распознавать как старые, так и новые модификации вирусов, с минимально возможной загрузкой системы. Существующие эвристические технологии, использующие ИНС, ИИС, ГА и МАС, позволяют с довольно большой долей вероятности идентифицировать широкий класс вирусов. Однако полностью эффективных способов борьбы с угрозами на сегодняшний день не существует.

Повысить эффективность обнаружения и анализа вредоносных программ помогут гибридные подходы [9], использующие различные технологии, которые позволяют решать сложные задачи и создавать программные системы, привносящие новое качество сервиса, высокую эффективность и ряд других преимуществ.

В работе предлагается нейросетевой ЭА вредоносных программ. Основным его компонентом является ИНС, обучение которой осуществляется с помощью ИИС. Для решения проблемы работы с полиморфным кодом предлагается взять в качестве входных данных ЭА протоколы, полученные в результате интерпретации вредоносных программ с помощью эмулятора.

## 2. Постановка задачи

Имеется множество исполняемых файлов (объектов) $X_f$, $f = \overline{1,F}$, которые могут содержать вредоносные коды. На этом множестве имеется разбиение на конечное число подмножеств (классов) $Y_k$, $k = \overline{1,K}$ и подробные карты их действий (протоколы). Разбиение определено не полностью – задан лишь некоторый набор обучающей информации $I_0(Y_1, Y_2, ..., Y_K)$ о классах $Y_k$. Протоколы могут содержать общие закономерности в поведении объектов, которые представляются в виде фрагментов программ, являющихся признаками объектов, совокупность которых определяет описание объектов (программ) $I(Y_1, Y_2, ..., Y_K)$. Предполагается, что для каждого исполняемого файла $X_f$, $f = \overline{1,F}$ существует вектор характерных поведенческих признаков $X_f = \lfloor x_{f,1}, x_{f,2}, ... x_{f,L_f} \rfloor$ из $L_f$ элементов, который может содержать вредоносные коды.

Задача детектирования и анализа вредоносных программ состоит в том, чтобы для исполняемых файлов $X_f$, $f = \overline{1,F}$ и набора классов $Y_k$, $k = \overline{1,K}$ по обучающей информации $I_0(Y_1, Y_2, ..., Y_K)$ и описаниям объектов $I(Y_1, Y_2, ..., Y_K)$ отнести исходные программы к определенному классу $Y_k$ путем выделения и сопоставления соответствующих признаков (фрагментов программ). Информация о вхождении объекта $X_f$, $f = \overline{1,F}$ в какой-либо класс представляется в виде информационного вектора

$$I(X_f) = \{I_1(X_f), I_2(X_f), ..., I_K(X_f)\},$$

где $I_k(X_f)$ несет информацию о принадлежности или не принадлежности объекта (программы) $X_f$, $f = \overline{1,F}$ к классу $Y_k$, $k = \overline{1,K}$.

Для решения задачи распознавания вредоносных программ необходимо определить рейтинг встречаемости для каждого найденного фрагмента программы, который покажет, в каком количестве объектов, из всего множества представленных, был найден данный фрагмент. При этом должна быть сформирована выборка с рейтингами признаков вредоносных программ.

## 3. Выделение характерных признаков

Для распознавания вредоносных программ необходимо выбрать перечень признаков, по которым это распознавание будет производиться. Для того чтобы эффективно распознать вредоносный код, необходимо составить библиотеку событий, которыми являются программные действия, связанные с системными вызовами, приводящими к изменениям в системе. Составление библиотеки событий делается на основе анализа большого количества вредоносных программ и выделения в них часто встречающихся характерных фрагментов. При этом основной задачей является собрать наиболее полный набор значимых признаков.

Работающая программа выполняет множество различных действий – изменяет значения регистров, флагов процессора, областей памяти и т.п. Но не все из них должны учитываться при распознавании вирусов. Можно взять большое количество вредоносных и невредоносных программ и подать их на семантический анализатор команд, подсчитав число встретившихся событий. Наиболее часто встречающиеся события во вредоносных программах и не встречающиеся в не вредоносных программах и будут характерными признаками.

Наибольшее количество алгоритмов скрытия кода от обнаружения ориентировано на вывод из строя анализаторов кода, поэтому следует отказаться от анализа кода в пользу поведенческого анализа, т.е. анализа взаимодействия программ с операционной системой (ОС) посредством системных вызовов. Большинство существующих систем поведенческого анализа базируются на использовании общих признаков (например, таких как копирование своего исполняемого файла в системные папки ОС, добавление своего исполняемого файла в объект автозагрузки и др.), присущих большому количеству вредоносных программ. Использование такого подхода оправдано тем, что, выделив и заложив такие признаки в поведенческий анализатор один раз, они должны покрыть большое количество существующих вредоносных программ, а также тех, которые появятся в будущем и будут использовать похожие алгоритмы. Недостатком в данном случае является то, что авторам вредоносных программ нетрудно будет обнаружить, на какие именно события настроен такой анализатор.

Эффективным с точки зрения точности распознавания вредоносных программ является подход, ориентированный на распознавание не общих для большинства семейств событий, а уникальных, характерных только для одного отдельно взятого семейства вирусов. Для получения эффективного и достоверного результата таких событий должно быть несколько. Чем больше их количество и чем более они уникальны, тем выше будет процент правильных срабатываний. Наиболее приоритетными при выборе должны быть следующие события:

– связанные с системным реестром;

– связанные с файловой системой;

– связанные с работой сети;

– связанные с манипулированием окнами;

– связанные с манипулированием процессами;

– связанные с установкой перехватчиков системных событий.

Однако некоторые действия, выполняемые вредоносными программами, могут также выполняться и невредоносными программами. Для того чтобы свести риск ложных срабатываний к минимуму, следует применить интеллектуальный анализ встречаемости различных событий во вредоносных и невредоносных объектах. Анализ заключается в разделении всех возможных комбинаций событий на две группы. К первой группе будут принадлежать вирусы искомого семейства, ко второй – вирусы других семейств, а также невредоносные программы. Данный подход при эффективной работе поведенческого анализатора, настроенного на распознавание особенностей поведения, характерного для конкретного типа вредоносных программ, позволяет определять новые модификации вирусов, близких по функционалу, без необходимости срочного обновления антивирусных баз.

Поскольку распознаванию подлежит поток тесно связанных друг с другом команд, который может нанести вред информации пользователя, то запуск вредоносного кода на компьютере пользователя является недопустимым. Для этого создается искусственное окружение с помощью специальных программных эмуляторов, в которых моделируется часть аппаратных и программных функций компьютера для того, чтобы зафиксировать все выполняемые исследуемой программой действия и собрать данные для дальнейшего их анализа. В этом случае программа выполняется не на физическом процессоре, а на интерпретаторе, который также по мере возможности эмулирует внешнюю среду: устройства, память, системные вызовы и др. При этом, если эмулируемая программа начнет выполнять различные деструктивные действия, информация на реальной системе не пострадает, так как будет недоступна из эмулятора.

**4. Модель эвристического анализатора**

Для решения задачи распознавания вредоносных программ в составе ЭА, который выполняет вероятностное распознавание на основе взвешенной оценки определенного количества признаков, используется ИНС. Схематически модель такого ЭА состоит из следующих блоков (рис. 1) [8]:

*1. Блок мониторинга*. Функцией данного блока является мониторинг поведения вредоносных и невредоносных объектов в целях получения протокола их работы (последовательностей вызова API функций и переданных им аргументов).

*2. Блок сравнения*. Данный блок принимает протоколы работы нескольких программ от блока мониторинга и сравнивает их. Результатом работы будет множество одинаковых фрагментов (признаков) в протоколах разных программ одного семейства.

*3. Библиотека признаков*. Данный блок хранит в себе все признаки, выявленные блоком сравнения, и ведет статистику их встречаемости. На основе этой статистики каждому признаку присваивается рейтинг, характеризующий встречаемость данного признака. Таким образом, фрагмент, который был найден в протоколах всех программ, будет иметь наибольший рейтинг, а фрагмент, который найден в наименьшем количестве программ – наименьший.

*4. Блок принятия решений*. Функция данного компонента – принятие решения о принадлежности или непринадлежности рассматриваемой программы к некоторому семейству вредоносных программ.

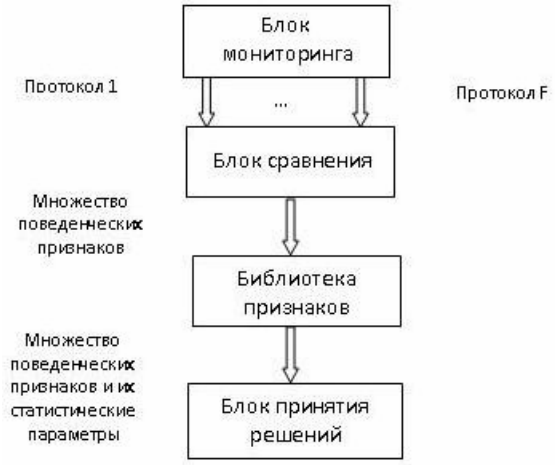


Рис. 1. Модель эвристического анализатора

*Блок мониторинга* представляет собой систему, моделирующую искусственное окружение для работы программ. К данному компоненту предъявляются следующие требования: 1) поддержка выполнения команд центрального процессора (ЦП); 2) поддержка основных и дополнительных, часто используемых API функций; 3) создание идентичных окружений, для всех запускаемых на нем программ.

Основной задачей этого блока является сбор следующих данных: 1) имена вызываемых в процессе работы программы функций WinAPI; 2) передаваемые исследуемой программой аргументы при вызове функций; 3) последовательность вызова функций;

На вход блока поступает объектный файл, который запускается на выполнение в искусственно созданной среде. Во время выполнения программы обработчиками вызываемых ею функций ведется сбор данных и формируется протокол. По завершению выполнения программы сформированный протокол выдается на выход блока для последующей обработки. В качестве модуля мониторинга использовался эмулятор iMUL, разработанный Лабораторией Касперского. Для сбора данных используются специальные программы перехвата вызовов API функций. Перехват API функций осуществляется путем внедрения в память функций инструкции JMP, которая передает управление обработчику, собирающему информацию.

*Блок сравнения* реализует ряд алгоритмов, направленных на выявление общих для исследуемых программ особенностей поведения. На вход блока посту-

пают протоколы работы исполняемых файлов, полученные от эмулятора. На выходе необходимо получить множество общих фрагментов для всех протоколов, которые были поданы на вход. Обработка происходит в два этапа:

1) фильтрация входящих протоколов и очистка их от данных, которые не несут информацию о событиях, связанных с действиями над файловой системой, системным реестром, процессами или работой в интернет;

2) отфильтрованные протоколы попарно сравниваются для нахождения в них идентичных фрагментов, независимо от их расположения.

Первый этап обработки реализуется специально разработанной вспомогательной программой «Threader». Программа также группирует протокол, разбивая его на несколько частей, в каждой из которых содержится информация о событиях, выполняемых отдельными потоками.

Второй этап реализуется специально разработанной программой «Matcher», с помощью которой производится сравнение протоколов. По завершении своей работы программа приводит список найденных совпадающих последовательностей, их смещения в массиве и размеры.

*Библиотека признаков* представляет собой хранилище данных, в котором содержатся фрагменты протоколов в удобном для обработки формате. Фрагменты представляют собой массивы структур, каждая из которых предназначена для хранения ключевой информации об API функции – тип функции и значения наиболее важных с точки зрения анализа аргументов.

Для каждого хранимого признака $x_f$, $f = \overline{1, L_f}$ подсчитывается рейтинг встречаемости:

$$R_f = X_f / F, \qquad (1)$$

где $X_f$ – количество найденных объектов с данным признаком; $F$ – общее количество объектов.

*Блок принятия решений* является основным компонентом ЭА, состоящим из нескольких параллельно работающих нейронных сетей (НС), каждая из которых предназначена для распознавания вредоносных программ определённого семейства и способна выполнять классификацию входных векторов на две группы. В работе использован многослойный персептрон с иммунным обучением, имеющий входной слой, один промежуточный слой и выходной слой, который содержит один нейрон. В промежуточном слое нейронов в качестве функций активации используются сигмоидальные функции:

$$z_m = f(u_m) = \frac{1}{1 + e^{-\lambda_m u_m}}, \; u_m = \sum_{n=1}^{N} w_{n,m} \cdot x_n + w_{o,m}, \qquad (2)$$

где $z_m$, $m = \overline{1, M}$ – выходной сигнал $m$-го нейрона промежуточного слоя, состоящего из $M$ нейронов, котрые имеют $N$ входов; $x_n$, $n = \overline{1, N}$ – $n$-я компонента входного вектора признаков, подаваемого на входной слой НС; $w_{n,m}$ – весовой коэффициент $n$-го входного признака $x_n$, поступающего на вход $m$-го нейрона промежуточного слоя; $w_{o,m}$ – значение смещения; $\lambda_m$ – коэффициент, определяющий крутизну функции активации $f(u_m)$.

Нейрон выходного слоя имеет пороговую функцию активации $\varphi$ и используется для вынесения вердикта о принадлежности анализируемой программы к семейству $Y_k$ вредоносных программ или непринадлежности:

$$y_k = \varphi\left(\sum_{m=1}^{M} v_m \cdot z_m + v_o\right) = \begin{cases} 1, & \text{если } y_k > 0, \\ 0, & \text{если } y_k \le 0. \end{cases} \qquad (3)$$

где $v_m$ – весовые коэффициенты, $v_o$ – смещение.

Количество входов НС выбирается равным количеству найденных при сравнении повторяющихся фрагментов (признаков). Из необходимости учитывать не только присутствие определенного признака, но и его проявляемость, на входы НС подаются рейтинги встречаемости признаков $R_f$. Обученная НС может оценивать признаки объекта и давать заключение – принадлежит объект к классу вредоносных или нет. Найденные в рассматриваемом образце признаки сверяются с библиотекой, откуда извлекаются значения рейтингов, которые затем поступают на входы НС. Если не все признаки в данной группе найдены, то рейтинги ненайденных признаков устанавливаются равными 0. На основе взвешенной оценки присутствующих признаков и их проявляемости НС выносит общее решение о вредоносности объекта.

ЭА работает в двух режимах: обучение и распознавание. В режиме обучения происходит настройка ЭА на распознавание поведений вредоносных программ, при этом выполняются следующие действия:

1. Множество исполняемых файлов, сходных по функционалу и принадлежащих к одному семейству, исследуется с помощью эмуляции и составляются подробные карты их действий (протоколы).

2. Протоколы сравниваются между собой для того, чтобы выявить общие закономерности в поведении объектов.

3. Найденные закономерности представляются в виде фрагментов протоколов и сохраняются в библиотеке. Подсчитывается рейтинг встречаемости для каждого найденного фрагмента. Этот рейтинг показывает, в каком количестве объектов, из всего множества представленных, был найден данный фрагмент. Происходит формирование выборки с рейтингами признаков вредоносных программ для обучения НС на положительные вердикты.

4. Выборка невредоносных программ исследуется с помощью эмуляции аналогичным образом. После этого в протоколах работы данных программ производится поиск фрагментов поведения вредоносных программ, предварительно сохраненных в библиотеке. Для всех невредоносных программ также формируется выборка с рейтингами признаков вредоносных программ, которая будет использована для обучения НС на отрицательные вердикты.

5. Производится создание НС и ее обучение на ранее подготовленных наборах рейтингов встречаемости. При этом выполняется кластеризация входного множества данных на два подмножества, первое из которых будет соответствовать вредоносным программам исследуемого семейства, а второе – невредоносным программам или же вредоносным программам из других семейств.

Будем решать задачу обучения всех параметров НС в режиме off-line с использованием ИИС [10], основной идеей которой является представление решаемой задачи в виде антигена, а возможные ее решения – в виде антител. В виде популяции антигенов $Ag$ выступает обучающая выборка размерности $S$ относительно входных $x_n$, $n = \overline{1,N}$ и выходной $Y$ переменных. В качестве антител используются векторы настраиваемых параметров. В одном антителе кодируются все настраиваемые параметры сети $w_{n,m}$, $w_{o,m}$, $v_m$, $v_o$ и $\lambda_m$, $n = \overline{1,N}$ $m = \overline{1,M}$. Используется вещественное кодирование антител, при котором каждый параметр вектора антитела описывается отдельным действительным числом. Для решения задачи используется модель кодирования настраиваемых параметров в виде адаптивного структурированного мультиантитела [11, 12], состоящего из двух частей, каждая из которых может обрабатываться независимо друг от друга (рис.2).

| $w_{1,1},\ldots,w_{1,M};\ldots,w_{N,1},\ldots,$ $w_{N,M};w_{0,1},\ldots,w_{0,M};\lambda_1,\ldots,\lambda_M$ | $v_1$ | … | $v_M$ | $v_0$ |
|---|---|---|---|---|
| $ab_0$ | $ab_1$ | … | $ab_M$ | $ab_{M+1}$ |
| Часть 1 | Часть 2 | | | |

Рис.2. Структура мультиантитела $mAb$

В части 1 мультиантитела закодированы весовые коэффициенты $w_{n,m}$, значения смещений $w_{o,m}$ и коэффициенты $\lambda_m$; в части 2 закодированы коэффициенты $v_m$ и смещение $v_o$.

В качестве вычислительной модели ИИС используются принципы клонального отбора и сетевого взаимодействия [10]. Согласно принципу клонального отбора антитело, распознавшее антиген, клонируется, и полученные клоны подвергаются мутации. Если в результате мутации аффинность клона улучшается, то соответствующее антитело заменяется своим клоном, т.е. осуществляется клональный отбор. В соответствии с принципом сетевого взаимодействия антитела взаимодействуют не только с антигенами, но и с другими антителами, в результате чего обеспечивается эффект взаимной стимуляции и суппрессии, что приводит к сокращению числа нейронов в промежуточном слое.

Алгоритм обучения НС представляет собой такую последовательность шагов:

1. Инициализация популяции мультиантител $mAb$. Инициализация начальной популяции мультиантител выполняется случайным образом.

2. Вычисление аффинности $Aff_{mAb-Ag}$ для каждого мультиантитела. Для вычисления аффинности мультиантитела необходимо подставить параметры, закодированные в мультиантителе, в НС. На вход НС подаются входные признаки $x_n$ и вычисляется значение выходной переменной $y_s$. Аффинность мультиантитела $mAb$ с антигеном $Ag$ вычисляется в виде:

$$Aff_{mAb-Ag} = (1 + d_{mAb-Ag})^{-1}, \qquad (4)$$

где $d_{mAb-Ag}$ – расстояние Хэмминга между полученным значением выхода НС $y_s, s = \overline{1,S}$ и желаемым $y$ для всех $S$ антигенов популяции $Ag$:

$$d_{mAb-Ag} = \sum_{s=1}^{S} y_s, \text{ где } y_s = \begin{cases} 1, & \text{если } y_s \neq y, \\ 0, & \text{если } y_s = y. \end{cases} \qquad (5)$$

3. Клонирование мультиантител пропорционально их аффинности и формирование популяции клонов $Cl$. Параметрами оператора клонирования являются количество антител для клонирования $n$ и кратность клонирования антитела $N_{Cl}$. В иммунном алгоритме обучения НС будем использовать фиксированное значение параметра $n$. Кратность клонирования мультиантитела $N_{Cl}$ будем регулировать в процессе работы иммунного алгоритма в зависимости от аффинности мультиантитела по соотношению:

$$N_{Cl} = \alpha * N_{Cl\_min} + (1 - \alpha) * N_{Cl\_max}, \qquad (6)$$

где $\alpha = \dfrac{Aff_{best} - Aff_{mAb-Ag}}{Aff_{best}}$, $N_{Cl\_min}$ и $N_{Cl\_max}$ – минимальная и максимальная кратность клонирования мультиантитела; $Aff_{best}$ – лучшее значение аффинности в текущем поколении.

4. Мутация клонов обратно пропорционально аффинности мультиантитела, формирование популяции мутированных клонов $MC$. Мутацию выбранных параметров мультиантитела $mAb$ будем выполнять путем добавления гауссовского шума:

$$mAb_{i+1} = mAb_i + N(0, \sigma_i), \quad (7)$$

где $\sigma_i$ – дисперсия случайной величины, которая ассоциируется с каждым параметром мультиантитела. Для изменения $\sigma_i$ используется соотношение:

$$\sigma_{i+1} = \sigma_i \frac{Aff_{best} - Aff_{mAb-Ag}}{Aff_{best} - Aff_{worst}}, \quad (8)$$

где $Aff_{worst}$ – худшее значение аффинности в текущем поколении.

5. Вычисление аффинности популяции MC в соответствии с (4). Если в результате мутации аффинность улучшилась, замена соответствующих мультиантител в популяции $mAb$.

6. Вычисление аффинности антител внутри части 2 мультиантитела. Суппрессия антител, аффинность которых больше заданного порога $\delta_{net}$.

Вычисление аффинности выполняется по выражению:

$$Aff_{ab_i-ab} = (1 + d_{ab_i-ab})^{-1}, \quad (9)$$

где $d_{ab_i-ab}$ – расстояние между $i$-м антителом и остальными антителами части 2 мультиантитела:

$$d_{ab_i-ab} = \left\| ab_i - ab_j \right\| = \sqrt{\sum_{j=1}^{M} (v_i - v_j)^2}, i = \overline{1, M}. \quad (10)$$

Выполнение суппрессии путем удаления антител $ab_i$ с аффинностью, большей заданного порога $\delta_{net}$, позволяет уменьшить количество нейронов в скрытом слое и устранить, таким образом, избыточность.

7. Проверка критерия останова. В качестве критерия останова используется либо достижение заданного порога аффинности, либо достижение заданного количества поколений работы алгоритма. Результатом работы алгоритма будет мультиантитело с лучшей по популяции аффинностью, содержащее настроенные параметры НС и определяющее ее структуру.

Шаги 1-5 алгоритма соответствуют принципу клонального отбора. Алгоритм на данных этапах работает с обеими частями мультиантитела. Шаг 6 соответствует принципу сетевого взаимодействия. Если ранее мультиантитело обрабатывалось как обычное антитело, то на данном шаге работа выполняется только с частью 2 мультиантитела, которая состоит из отдельных антител, представляющих собой параметры $v_m$, $m = \overline{0, M}$.

При распознавании в режиме off-line уже обученная НС классифицирует объект как вредоносный или невредоносный по представленному ей набору найденных в объекте признаков и их рейтингов, при этом выполняются следующие действия:

1. Исполняемый файл исследуется на эмуляторе.

2. В протоколе исследуемого файла производится поиск фрагментов поведения вредоносных программ из библиотеки. Из библиотеки извлекаются рейтинги для найденных фрагментов. Рейтинги подаются на входы НС, которая выносит вердикт о принадлежности исследуемого объекта к классу вредоносных программ.

**5. Моделирование работы ЭА**

Экспериментальные исследования проводились с использованием специального программного обеспечения, которое эмулирует работу ЦП, API функций ОС и ее внутренних структур, и подходит для задачи мониторинга поведения программ и сбора необходимых данных. Для моделирования работы ЭА был выбран следующий набор инструментов:

1. Моделирование блока мониторинга проводилось с использованием эмулятора системы Microsoft Windows – iMUL. Данное программное обеспечение эмулирует работу ЦП, API функций ОС и ее внутренних структур и подходит для задачи мониторинга поведения программ и сбора необходимых данных.

2. Моделирование блока сравнения признаков проводилось с помощью специально разработанных для этого утилит Threader и Matcher. Данные утилиты предназначены для преобразования и исследования информации, получаемой от эмулятора iMUL. Threader является консольным приложением, входные данные для его работы – протокол эмулятора в текстовом виде. Выходом для этой программы являются текстовые файлы в ее рабочей папке с именами, которые соответствуют идентификаторам потоков. Matcher – так же является консольным приложением, выходные данные для его работы – файлы, созданные программой Threader.

3. Моделирование работы НС и ИИС проводилось при помощи программного пакета MATLAB R2013a, технологическая платформа которой способна моделировать работу НС с необходимой архитектурой, поддерживает обучение с учителем с использованием ИИС и подходит для моделирования блока принятия решений.

Первый этап работы предполагает запуск вредоносных объектов на эмуляторе и получение протоколов. Для эксперимента были взяты следующие вредоносные программы семейства троянских программ, предназначенных для похищения паролей, включая следующие модификации:

Trojan-PSW.Win32.Tepfer.kcoi;

Trojan-PSW.Win32.Tepfer.kcta;

Trojan-PSW.Win32. Tepfer.kcti;

Trojan.Win32.KillFiles.brrq;

Trojan.Win32.Bublik.axmx;

Trojan-Ransom.Win32.Foreign.dbdo.

Данные модификации были запущены на эмуляторе и для каждой из них был получен протокол. Полученные протоколы были проанализированы с помощью программы Threader, которая вычла из них служебную информацию о системных вызовах, не изменяющих систему. Отфильтрованные протоколы сравнивались попарно каждый с каждым при помощи специально разработанной утилиты Matcher. Результатом работы данной программы явилось множество общих для всех входных протоколов фрагментов (характерных поведенческих признаков). Программа выявила 13 признаков для семейства троянских программ, для которых был производен подсчет их рейтингов встречаемости.

Была построена НС в виде трехслойного персептрона для детектирования вредоносных программ семейства троянских программ, для которой сформирована обучающая выборка из полученных рейтингов встречаемости признаков во вредоносных и невредоносных объектах. Обучение НС выполнено с использованием ИИС.

Была произведена классификация новых образцов программ с помощью созданной НС. В качестве тестовых программ для распознавания были представлены две вредоносные программы: 1) Trojan.OlympicGames и 2) Win32.Worm.Prolaco.S, и две невредоносные программы: 3) Mirabilis ICQ и 4) Opera WebBrowser. Представленные программы были распознаны верно. Следовательно, разработанная НС с иммунным обучением успешно решает возложенную на нее задачу и способна распознавать новые модификации вредоносных программ.

## 6. Выводы

Решена актуальная задача выявления и распознавания как существующих, так и новых модификаций вредоносных программ на основе использования нейросетевого ЭА с иммунным обучением. Предложена модель ЭА, основными компонентами которой являются блок мониторинга, блок сравнения, библиотека признаков и блок принятия решений, который является основным компонентом ЭА, состоящим из НС, каждая из которых предназначена для распознавания вредоносных программ определённого семейства. Для обучения НС использована ИИС, в которой модель кодирования настраиваемых параметров представлена в виде адаптивного структурированного мультиантитела.

Проведены экспериментальные исследования на примере вредоносных программ семейства троянских программ, предназначенных для похищения паролей. Результаты моделирования показали, что предложенный нейросетевой ЭА с иммунным обучением способен распознавать новые модификации вредоносных программ.

**Литература: 1.** *Шибаева Т.А., Щеглов А.Ю., Оголюк А.А.* Защита от внедрения и запуска вредоносных программ / / Вопросы защиты информации. 2011. № 2. С. 26-30. **2.** *Новиков Е.А., Краснопевцев А.А.* Сравнительный анализ методов обнаружения вторжений // Безопасность информационных технологий. 2012. № 1. С. 47-50. **3.** *Гаврилов А.В.* Применение постоянно модифицирующихся нейронных сетей для защиты программного обеспечения // Нейрокомпьютеры, разработка, применение. 2008. № 1-2. С. 90-101. **4.** Нейросетевая технология обнаружения сетевых атак на информационные ресурсы / *Ю.Г. Емельянова, А.А. Талалаев, И.П. Тищенко, В.П. Фраленко* // Программные системы: теория и приложения. 2011. Т. 2, № 3. С. 3-15. **5.** *Bezobrazov S., Golovko V.* Artificial immune system approach for malware detection: neural networks applying for immune detectors construction // Inernational journal of «Computing». 2008. Vol. 7, no. 2. P. 44-50. **6.** *Гаврилов А.В., Тихомиров А.В.* Применение иммунных систем в целях защиты корпоративной информации от нецелевого использования // Известия Южного федерального университета. Технические науки. 2010. Т. 108. № 7. С. 154-163. **7.** *Zekri M., Souici-Meslati L.* Artificial Immune System for Intrusion Detection // Evolutionary Computation. 2011. V. 13, № 2. P. 145-153. **8.** *Кораблев Н.М., Кушнарев М.В.* Модель эвристического анализатора вредоносных программ на основе искусственной иммунной сети // Системи обробки інформації. 2013. Вип. 8 (115). С. 216-222. **9.** *Войцехович Л.Ю. Головко В.А., Курош Мадани*. Применение мультиагентной системы с нейросетевым классификатором для выявления атак в трафике TCP/IP // Нейроинформатика. 2011. Часть 1. С. 190-201.**10.** *Dasgupta D., Nino L.F.* Immunological computation, theory and applications. CRC Press, 2009. 298 p. **11.** *Korablev N., Sorokina I.* Immune Approach for Neuro-Fuzzy Systems Learning Using Multiantibody Model // Springer Lecture Notes in Computer Science. 2011. Vol. 6825. P. 395–405. **12.** *Кораблев Н.М., Сорокина И.В., Русецкий А.И.* Иммунный алгоритм обучения адаптивных нечетких нейронных сетей // Системи управління, навігації та зв'язку. 2008. Вип. 4 (8). С. 62–67.

Поступила в редколлегию 12.04.2014

**Рецензент:** д-р техн. наук, проф. Руденко О.Г.

**Кораблев Николай Михайлович,** д-р техн. наук, профессор кафедры ЭВМ ХНУРЭ. Научные интересы: интеллектуальная обработка информации. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: (057) 7021-354.

**Кушнарев Максим Владимирович,** ассистент кафедры ЭВМ ХНУРЭ. Научные интересы: обнаружение и анализ вредоносных программ с использованием методов искусственного интеллекта. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: (057) 7021-354.

**Ужвий Денис Петрович,** магистрант кафедры ЭВМ ХНУРЭ. Научные интересы: искусственные нейронные сети. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: (057) 7021-354.